Printed Version of
VNA Help File
Supports
A.15.40.xx/A.15.50.x

Keysight
VNA Series
Network Analyzers
Documentation Warranty

THE MATERIAL CONTAINED IN THIS DOCUMENT IS PROVIDED "AS IS," AND IS SUBJECT TO BEING CHANGED, WITHOUT NOTICE, IN FUTURE EDITIONS. FURTHER, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, KEYSIGHT DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED WITH REGARD TO THIS MANUAL AND ANY INFORMATION CONTAINED HEREIN, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. KEYSIGHT SHALL NOT BE LIABLE FOR ERRORS OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, USE, OR PERFORMANCE OF THIS DOCUMENT OR ANY INFORMATION CONTAINED HEREIN. SHOULD KEYSIGHT AND THE USER HAVE A SEPARATE WRITTEN AGREEMENT WITH WARRANTY TERMS COVERING THE MATERIAL IN THIS DOCUMENT THAT CONFLICT WITH THESE TERMS, THE WARRANTY TERMS IN THE SEPARATE AGREEMENT WILL CONTROL.
U.S. Government Rights

U.S. Government Rights. The Software is “commercial computer software,” as defined by Federal Acquisition Regulation (“FAR”) 2.101. Pursuant to FAR 12.212 and 27.405-3 and Department of Defense FAR Supplement (“DFARS”) 227.7202, the U.S. government acquires commercial computer software under the same terms by which the software is customarily provided to the public. Accordingly, Keysight provides the Software to U.S. government customers under its standard commercial license, which is embodied in its End User License Agreement (EULA), a copy of which can be found at http://www.keysight.com/find/sweula. The license set forth in the EULA represents the exclusive authority by which the U.S. government may use, modify, distribute, or disclose the Software. The EULA and the license set forth therein, does not require or permit, among other things, that Keysight: (1) Furnish technical information related to commercial computer software or commercial computer software documentation that is not customarily provided to the public; or (2) Relinquish to, or otherwise provide, the government rights in excess of these rights customarily provided to the public to use, modify, reproduce, release, perform, display, or disclose commercial computer software or commercial computer software documentation. No additional government requirements beyond those set forth in the EULA shall apply, except to the extent that those terms, rights, or licenses are explicitly required from all providers of commercial computer software pursuant to the FAR and the DFARS and are set forth specifically in writing elsewhere in the EULA. Keysight shall be under no obligation to update, revise or otherwise modify the Software. With respect to any technical data as defined by FAR 2.101, pursuant to FAR 12.211 and 27.404.2 and DFARS 227.7102, the U.S. government acquires no greater than Limited Rights as defined in FAR 27.401 or DFAR 227.7103-5 (c), as applicable in any technical data.
This page left blank intentionally.
## TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home</td>
<td>71</td>
</tr>
<tr>
<td>What's New</td>
<td>72</td>
</tr>
<tr>
<td>Administrative Tasks</td>
<td></td>
</tr>
<tr>
<td>Microsoft EULA</td>
<td>91</td>
</tr>
<tr>
<td>VNA User Accounts and Passwords</td>
<td>95</td>
</tr>
<tr>
<td>Computer Properties</td>
<td>98</td>
</tr>
<tr>
<td>Error-check and Disk Defragmenter</td>
<td>104</td>
</tr>
<tr>
<td>Operating System Recovery</td>
<td>106</td>
</tr>
<tr>
<td>Windows Considerations</td>
<td>107</td>
</tr>
<tr>
<td>Windows File Locations</td>
<td>109</td>
</tr>
<tr>
<td>Quick Start</td>
<td></td>
</tr>
<tr>
<td>Simulator</td>
<td>110</td>
</tr>
<tr>
<td>Connector Care</td>
<td>115</td>
</tr>
<tr>
<td>Front Panel Tour</td>
<td>127</td>
</tr>
<tr>
<td>Rear Panel Tour</td>
<td>144</td>
</tr>
<tr>
<td>Screen Display Tour</td>
<td>150</td>
</tr>
<tr>
<td>Powering the VNA ON and OFF</td>
<td>159</td>
</tr>
<tr>
<td>Traces, Channels, Windows, and Sheets</td>
<td>162</td>
</tr>
<tr>
<td>QuickStart Dialog</td>
<td>186</td>
</tr>
<tr>
<td>Basic Measurement Sequence</td>
<td>193</td>
</tr>
<tr>
<td>Frequency Blanking</td>
<td>194</td>
</tr>
<tr>
<td>Internal Second Source</td>
<td>197</td>
</tr>
<tr>
<td>Connectivity Guide</td>
<td>199</td>
</tr>
<tr>
<td>Preferences</td>
<td>200</td>
</tr>
<tr>
<td>VNC</td>
<td>209</td>
</tr>
<tr>
<td>LXI Compliance</td>
<td>211</td>
</tr>
<tr>
<td>Dialog Transparency</td>
<td>214</td>
</tr>
<tr>
<td>Keyboard Shortcuts</td>
<td>216</td>
</tr>
<tr>
<td>Using Help</td>
<td>218</td>
</tr>
<tr>
<td>Help About</td>
<td>229</td>
</tr>
<tr>
<td>1. Set Up a Measurement</td>
<td></td>
</tr>
<tr>
<td>Preset the VNA</td>
<td>231</td>
</tr>
<tr>
<td>Measurement Classes</td>
<td>234</td>
</tr>
<tr>
<td>Measurement Parameters</td>
<td>237</td>
</tr>
</tbody>
</table>
Frequency Range 250
Power Level 262
Receiver Leveling 275
Sweep Settings 284
Trigger Setup 298
External Triggering 303
Reference 311
Trigger Model Animation 314
Data Format 315
Scale 324
Path Configurator 333
Phase Control 337
Phase Coherent Measurements 347
Customize Your Analyzer Screen 349
Copy Channels 372
DC Control 375
ADC Measurements 378
Pulsed Measurement Setup 380
Pulse Generators 380
Global Source 399
Device Expert (DE) 402
Undo/Redo 419

2. Optimize a Measurement 423
   Dynamic Range 424
   Number of Data Points 427
   Phase Accuracy 429
   Phase Coherent Measurements 347
   Electrically Long Devices 433
   Reflection Accuracy 436
   Measurement Stability 439
   Noise Reduction Techniques 441
   Crosstalk 451
   Effects of Accessories 452
   Fastest Sweep 453
   Multiple State Measurements 455
   Fastest Data Transfer 459
   Using Macros 460

3. Calibrate a Measurement
Connectors Tab 788
Standards Tab 791
SOLT Tab 800
TRL Tab 803
Multiline TRL Calibration 807

4. Analyze Data
   Locate Data Using Markers 819
   Math & Memory Operations 861
   8510 Mode 869
   Equation Editor 871
   Equation Editor Import Functions 887
   External DC Meter Data Conversion 896
   Parameter Conversion 900
   Use Limits to Test Devices 903
   Use Ripple Limit Test 913
   Use Bandwidth Limit Test 920

5. Output Data
   Save and Recall Data 923
   Drive Mapping 946
   Print 947

Programming 953

COM

   Commands
      Objects
         ActiveParametersApp Object 956
         The Analyzer Object Model 958
         Application Object 961
         AuxTrigger Object 968
         BalancedMeasurement Object 970
         BalancedStimulus Object 972
         BalancedTopology Object 974
         CalFactorSegments Collection 978
         CalFactorSegmentsPMAR Collection 979
         CalibrateAllChannels Object 980
         Calibrator Object 983
         CalKit Object 989
         CalManager Object 991
         CalSet Object 995
<table>
<thead>
<tr>
<th>Class</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalSets Collection</td>
<td>1000</td>
</tr>
<tr>
<td>CalStandard Object</td>
<td>1002</td>
</tr>
<tr>
<td>Capabilities Object</td>
<td>1005</td>
</tr>
<tr>
<td>Channel Object</td>
<td>1010</td>
</tr>
<tr>
<td>Channels Collection</td>
<td>1019</td>
</tr>
<tr>
<td>ComColors Object</td>
<td>1021</td>
</tr>
<tr>
<td>ComTraceColors Object</td>
<td>1023</td>
</tr>
<tr>
<td>Converter:Object</td>
<td>1025</td>
</tr>
<tr>
<td>ConverterEmbeddedLO Object</td>
<td>1031</td>
</tr>
<tr>
<td>CorrectionMethods Object</td>
<td>1033</td>
</tr>
<tr>
<td>DCStimulus Object</td>
<td>1035</td>
</tr>
<tr>
<td>DIQ Object</td>
<td>1037</td>
</tr>
<tr>
<td>Display Object</td>
<td>1040</td>
</tr>
<tr>
<td>E5091Testset Collection</td>
<td>1041</td>
</tr>
<tr>
<td>E5091Testset Object</td>
<td>1043</td>
</tr>
<tr>
<td>ECalModule Object</td>
<td>1045</td>
</tr>
<tr>
<td>ECalModules Collection</td>
<td>1047</td>
</tr>
<tr>
<td>ECalUserCharacterizer Object</td>
<td>1049</td>
</tr>
<tr>
<td>EmbeddedLO Object</td>
<td>1052</td>
</tr>
<tr>
<td>EmbeddedLODiagnostic Object</td>
<td>1054</td>
</tr>
<tr>
<td>ENRFile Object</td>
<td>1056</td>
</tr>
<tr>
<td>ExternalDCDevice Object</td>
<td>1058</td>
</tr>
<tr>
<td>ExternalDevice Object</td>
<td>1061</td>
</tr>
<tr>
<td>ExternalDevices Collection</td>
<td>1064</td>
</tr>
<tr>
<td>ExternalPulseGenerator Object</td>
<td>1066</td>
</tr>
<tr>
<td>ExternalSMUDevice Object</td>
<td>1068</td>
</tr>
<tr>
<td>ExternalSource Object</td>
<td>1070</td>
</tr>
<tr>
<td>ExternalTestsets Collection</td>
<td>1072</td>
</tr>
<tr>
<td>FIFO Object</td>
<td>1074</td>
</tr>
<tr>
<td>Fixturing Object</td>
<td>1076</td>
</tr>
<tr>
<td>FOM Collection</td>
<td>1081</td>
</tr>
<tr>
<td>FOMRange Object</td>
<td>1083</td>
</tr>
<tr>
<td>GainCompression Meas Object</td>
<td>1085</td>
</tr>
<tr>
<td>GainCompression Object</td>
<td>1087</td>
</tr>
<tr>
<td>GainCompressionCal Object</td>
<td>1091</td>
</tr>
<tr>
<td>Gating Object</td>
<td>1093</td>
</tr>
<tr>
<td>GlobalPowerLimit Object</td>
<td>1095</td>
</tr>
<tr>
<td>GroupDelayAperture Object</td>
<td>1097</td>
</tr>
</tbody>
</table>
AnalysisEnable Property
AnalysisIsDiscreteFreq Property
AnalysisXAxis Property
Application
Arrange Windows
Attenuator Mode
Attenuator
AutoBandwidth Property
AutoCWSweepTime Property
AutoDetection Property
AutoIFBandWidth Property
AutoIFBWAjustment
AutoOptimizePRF Property
AutoOrient
AutoOrientTuner Property
AutoPortExtConfig
AutoPortExtDCOffset
AutoPortExtLoss
AutoPortExtSearchStart
AutoPortExtSearchStop
AutoPortExtState
AutoPulseTiming Property
AutoSelectPulseGen Property
AuxiliaryTriggerCount
AuxTriggerScopeIsGlobal
AvailableMeasurementClasses Property
AverageMode
Averaging Count
Averaging Factor
Averaging ON/OFF
AvoidSpurs
B_BalPortNegative Property
B_BalPortPositive Property
Background Property
BackOff Property
BackOffGain Property
BackOffPin Property
BackOffPout Property
BalancedMode 1359
BalancedPortTrueState Property 1360
BalPort1PhaseOffset 1361
BalPort1StartPhase 1362
BalPort1StopPhase 1363
BalPort2PhaseOffset 1364
BalPort2PowerOffset 1365
BalPort2StartPhase 1366
BalPort2StopPhase 1367
BalSMeasurement Property 1368
BandDensityACPRState Property 1369
BandDensityBW Property 1370
BandDensityEQSPan Property 1371
BandDensityNoiseState Property 1372
BandDensityNPRState Property 1373
BandDensityPowerBW Property 1374
BandDensityPowerState Property 1375
BandDensityToneBW Property 1376
BandDensityToneSpacing Property 1377
BandDensityToneState Property 1378
BandDensityValue Property 1379
BandNoisedBmpHz Property 1380
BandnoiseSpan Property 1381
BandnoiseState Property 1382
BandPowerdBm Property 1383
BandpowerSpan Property 1384
BandpowerState Property 1385
BandwidthSearch Property 1386
BandwidthShape (SA) Property 1387
Bandwidth Target 1388
Bandwidth Tracking 1389
BandwidthNarrowMax Property 1391
BandwidthNarrowMin Property 1392
BandwidthWideMax Property 1393
BandwidthWideMin Property 1394
BB_BalPort1Negative 1395
BB_BalPort1Positive 1396
BB_BalPort2Negative 1397
CalSet Property
Center
Center (Meas)
Center Frequency
CenterFrequencyStepSize Property
CenterFrequencyStepSizeMode Property
ChanActive Property
Channel Number
Channels Property
CharacterizationNumber
CharacterizeMixerOnly
CharFileName
CharMixerReverse
CitiContents
CitiFormat
CmnModeZConvPortImag
CmnModeZConvPortReal
CmnModeZConvPortZ0
CmnModeZConvState
CompatibleCalKits
CompositeNormalizationMode
CompositeNormalizedCSOPower
CompositeNormalizedCTBPower
Compression Property
CompressionAlgorithm
CompressionBackoff
CompressionDeltaX
CompressionDeltaY
CompressionInterpolation
CompressionLevel
CompressionMax Property
CompressionLevel (mkr)
CompressionPin
CompressionPout
CompressionSaturation Property
ConfigurationFile Property
Configurations Property
ConfirmPreset Property
<table>
<thead>
<tr>
<th>Property Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConnectorType</td>
<td>1480</td>
</tr>
<tr>
<td>ConnectorType ECal</td>
<td>1482</td>
</tr>
<tr>
<td>ControlLines</td>
<td>1483</td>
</tr>
<tr>
<td>CorrectionSubsettingState Property</td>
<td>1485</td>
</tr>
<tr>
<td>Count</td>
<td>1486</td>
</tr>
<tr>
<td>Couple Ports</td>
<td>1488</td>
</tr>
<tr>
<td>CoupleChannelParams</td>
<td>1489</td>
</tr>
<tr>
<td>Coupled</td>
<td>1490</td>
</tr>
<tr>
<td>Coupled Markers</td>
<td>1491</td>
</tr>
<tr>
<td>CoupledMarkersMethod Property</td>
<td>1492</td>
</tr>
<tr>
<td>CoupledParameters - Gate</td>
<td>1493</td>
</tr>
<tr>
<td>CoupledParameters - Transform</td>
<td>1494</td>
</tr>
<tr>
<td>CouplePhasePortSettings Property</td>
<td>1496</td>
</tr>
<tr>
<td>CoupleTonePower</td>
<td>1497</td>
</tr>
<tr>
<td>CpuRevision Property</td>
<td>1498</td>
</tr>
<tr>
<td>CSONumDistortionProducts</td>
<td>1499</td>
</tr>
<tr>
<td>CSOOffset</td>
<td>1500</td>
</tr>
<tr>
<td>CTBOffset</td>
<td>1501</td>
</tr>
<tr>
<td>CTBXMODNumCarriers</td>
<td>1502</td>
</tr>
<tr>
<td>CurrentLimit</td>
<td>1503</td>
</tr>
<tr>
<td>CustomBalNegativePort Property</td>
<td>1504</td>
</tr>
<tr>
<td>CustomBalPositivePort Property</td>
<td>1505</td>
</tr>
<tr>
<td>CustomCalConfiguration Property</td>
<td>1506</td>
</tr>
<tr>
<td>CustomChannelConfiguration</td>
<td>1507</td>
</tr>
<tr>
<td>CustomMeasurementConfigurationProperty</td>
<td>1508</td>
</tr>
<tr>
<td>CustomPhysicalPortsSequence Property</td>
<td>1509</td>
</tr>
<tr>
<td>CustomPortTypeSequence Property</td>
<td>1510</td>
</tr>
<tr>
<td>CustomSEPysicalPort Property</td>
<td>1511</td>
</tr>
<tr>
<td>CustomTopologyPortCount Property</td>
<td>1512</td>
</tr>
<tr>
<td>CW Frequency</td>
<td>1513</td>
</tr>
<tr>
<td>CWFrequency CS Property</td>
<td>1514</td>
</tr>
<tr>
<td>Data</td>
<td>1515</td>
</tr>
<tr>
<td>DataAndLimits Property</td>
<td>1516</td>
</tr>
<tr>
<td>DataAsBytes Property</td>
<td>1517</td>
</tr>
<tr>
<td>DataAsFloat32 Property</td>
<td>1518</td>
</tr>
<tr>
<td>DataAsInt16 Property</td>
<td>1519</td>
</tr>
<tr>
<td>DataAsInt32 Property</td>
<td>1520</td>
</tr>
<tr>
<td>DataBinCount Property</td>
<td>1521</td>
</tr>
</tbody>
</table>
DataByteCount Property
DataByteSize Property
DataByteSizeLOW Property
DataByteSizeHIGH Property
DataBytesPerBin Property
DataCount
DataExportMarkersEnabled Property
DataExportWindowingFactor Property
DataFirstRFBin Property
DataFormat Property
DataInCompactForm
DataLevelThreshold Property
DataLevelThresholdEnabled Property
DateTime Property
DCCorrection Property
DCOffset Property
DCOrder Property
DCScale Property
DCSourcePointCount Property
DCSourceSweepFirstDimension Property
DCSourceSweepState
DCState Property
DCType Property
DefinedRoles Property
Delay (Segment Sweep) Property
Delay
Delay pulse
Delay_trigger
DelayCalculationMethod Property
DelayIncrement
DelayOption Property
DeltaFrequency
DeltaFrequencyStart
DeltaFrequencyStop
DeltaMarker
Description
DescriptiveText
DetectorFunction Property
<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeviceInputPort (FCA)</td>
<td>1561</td>
</tr>
<tr>
<td>DeviceInputPort</td>
<td>1562</td>
</tr>
<tr>
<td>DeviceNames Property</td>
<td>1563</td>
</tr>
<tr>
<td>DeviceOutputPort (FCA)</td>
<td>1564</td>
</tr>
<tr>
<td>DeviceLinearPowerLevel</td>
<td>1565</td>
</tr>
<tr>
<td>DeviceOutputPort</td>
<td>1566</td>
</tr>
<tr>
<td>DeviceType Property</td>
<td>1567</td>
</tr>
<tr>
<td>DiffPortMatch_C</td>
<td>1568</td>
</tr>
<tr>
<td>DiffPortMatch_G</td>
<td>1569</td>
</tr>
<tr>
<td>DiffPortMatch_L</td>
<td>1570</td>
</tr>
<tr>
<td>DiffPortMatch_R</td>
<td>1571</td>
</tr>
<tr>
<td>DiffPortMatchMode</td>
<td>1572</td>
</tr>
<tr>
<td>DiffPortMatchState</td>
<td>1573</td>
</tr>
<tr>
<td>DiffPortMatchUserFilename</td>
<td>1574</td>
</tr>
<tr>
<td>DiffZConvPortImag</td>
<td>1575</td>
</tr>
<tr>
<td>DiffZConvPortReal</td>
<td>1576</td>
</tr>
<tr>
<td>DiffZConvPortZ0</td>
<td>1577</td>
</tr>
<tr>
<td>DiffZConvState</td>
<td>1578</td>
</tr>
<tr>
<td>DimensionCatalog Property</td>
<td>1579</td>
</tr>
<tr>
<td>DimensionCount Property</td>
<td>1580</td>
</tr>
<tr>
<td>DimensionPointCount Property</td>
<td>1581</td>
</tr>
<tr>
<td>DimensionRepeatCount Property</td>
<td>1582</td>
</tr>
<tr>
<td>DirectoryPath Property</td>
<td>1583</td>
</tr>
<tr>
<td>DiscreteFrequencies Property</td>
<td>1584</td>
</tr>
<tr>
<td>Display Format</td>
<td>1585</td>
</tr>
<tr>
<td>DisplayAutomationErrors</td>
<td>1587</td>
</tr>
<tr>
<td>DisplayDomain Property</td>
<td>1588</td>
</tr>
<tr>
<td>DisplayGlobalPassFail</td>
<td>1589</td>
</tr>
<tr>
<td>DisplayInputPower Property</td>
<td>1590</td>
</tr>
<tr>
<td>DisplayInterpolationState Property</td>
<td>1591</td>
</tr>
<tr>
<td>DisplayRange</td>
<td>1592</td>
</tr>
<tr>
<td>Distance</td>
<td>1593</td>
</tr>
<tr>
<td>DistanceMarkerMode</td>
<td>1594</td>
</tr>
<tr>
<td>DistanceMarkerUnit</td>
<td>1595</td>
</tr>
<tr>
<td>Divisor</td>
<td>1596</td>
</tr>
<tr>
<td>Do1PortEcal</td>
<td>1597</td>
</tr>
<tr>
<td>Do2PortEcal</td>
<td>1598</td>
</tr>
<tr>
<td>Domain</td>
<td>1599</td>
</tr>
</tbody>
</table>
EnableDetectorBypass Property
EnableDitherFFTGridOrigin Property
EnableForceADCRrecordSize Property
EnableForceLOToFrequency Property
EnableImageRejectTraces Property
EnableLOPowerCal
EnableModulationControl Property
EnableOffsetDelays Property
EnablePhase Property
EnablePowerCompensation Property
EnableSnPDataExtrapolation Property
EnableRandomizedLO Property
EnableSourceUnleveledEvents
EndOfSweepOperation
ENRFile
ENRID
ENRSN
EqualTonePower Property
EquationEditorFastProcessing Property
Error Correction
ErrorCorrection(Channel)
ErrorCorrectionIndicator Property
ErrorQuery Property
ErrorTermUncertainty Property
ExitCmd Property
ExportReceiverCount Property
ExportReceiverList Property
ExportReceiverSetList Property
ExtDCDeviceCorrectionScale Property
ExtendedProperties Property
External ALC
ExternalDeviceDeActivatePolicy Property
ExternalTriggerConnectionBehavior
ExternalTriggerDelay
ExtractionToneMode Property
F1Frequency
F2Frequency
FailedTraces Property
Frequency Offset Frequency 1727
Frequency Offset Multiplier 1728
FrequencyOffsetRangeForCalComputations Property 1729
Frequency Offset Override To CW 1730
Frequency Offset State 1731
FrequencyStep Property 1732
FrequencyType Property 1733
FullyCorrectedPorts Property 1734
Gain Property 1735
GainLinear Property 1736
GainMax Property 1737
GainSaturation Property 1738
GeneratedCalsets Property 1739
FrequencySpan 1740
Gate Shape 1741
Gate Type 1742
GPIBAddress 1743
GPIB Mode 1744
GPIBPortCount 1745
Grid Property 1746
GridLineType Property 1747
HandshakeEnable 1748
Has2ndOrderTrace Property 1749
HasDirectReceiverAccess Property 1750
HasLowFrequencyExtension Property 1751
HasItem Property 1752
HighAmplitude Property 1753
HighestOrderProduct 1754
HighestOrderProductInUse Property 1755
ID 1756
IDQuery Property 1757
IDString 1758
IF Bandwidth Option 1759
IF Bandwidth 1760
IFBW Property 1761
IFBWList Property 1762
IFDenominator_Property 1763
IFNumerator_Property 1764
IFFrequency
IFFrequencyMode
IFSideband
IFSideband (conv)
IFStartFrequency
IFStopFrequency
ImageRejectMethod Property
ImageRejectStrength Property
ImpedanceStates
Impulse Width
IMToneIFBandwidth
InactiveLabels Property
Include2ndOrderProduct
IncludePort Property
IncludeReverseSweep Property
IncludeUnknownMixer Property
IndependentPowerCalibration Property
IndexState
InitCmd Property
Input A
Input B
Input C
Input Denominator
Input Fixed Frequency
Input Is Greater Than LO
Input Linear Power Level Property
Input Numerator
Input Power
Input Range Mode
Input Range Mode cv Property
Input Start Frequency
Input Start Power
Input Stop Frequency
Input Stop Power
Internal DC Receiver Count Property
Internal DC Receiver Names Property
Internal DC Source Count Property
Internal DC Source Names Property
InternalSourcePortCount Property 1802
InternalSourcePortNames Property 1803
InternalTestsetPortCount 1804
InternalTestsetPortNames Property 1805
Interpolate Correction 1806
Interpolated 1807
InterpolateMemory Property 1808
InterpolateMemoryIsDefault Property 1809
InterpolateNormalization 1810
Interrupt 1811
Invert Property 1812
IOConfiguration Property 1813
IOEnable Property 1814
IsBalanced Property 1815
IsContinuous 1816
IsDevicePresent Property 1817
IsECALModuleFoundEx 1818
IsEnhancedModulationControl Property 1819
IsFrequencyOffsetPresent 1820
IsHold 1821
IsMarkerOn 1822
IsolationAveragingIncrement 1823
IsOn 1824
IsReceiverStepAttenuatorPresent 1825
IsReferenceBypassSwitchPresent 1826
IsSingleEnded Property 1827
IsSParameter 1828
Item Property 1829
Items Property 1832
IterationNumber 1833
IterationsTolerance 1834
Kaiser Beta 1835
L0 1836
L1 1837
L2 1838
L3 1839
Label 1840
Label Testset 1841
<table>
<thead>
<tr>
<th>Variable</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LANConfiguration</td>
<td>1842</td>
</tr>
<tr>
<td>LastCalPassedTolerance</td>
<td>1843</td>
</tr>
<tr>
<td>LastLevelingAsSPC</td>
<td>1844</td>
</tr>
<tr>
<td>LastModified</td>
<td>1845</td>
</tr>
<tr>
<td>LegacyGroupDelayApertureMathProperty</td>
<td>1846</td>
</tr>
<tr>
<td>LevelingIFBW</td>
<td>1847</td>
</tr>
<tr>
<td>Limit Property</td>
<td>1848</td>
</tr>
<tr>
<td>LimitCmd Property</td>
<td>1849</td>
</tr>
<tr>
<td>LimitFrequency Property</td>
<td>1850</td>
</tr>
<tr>
<td>Limit Line Begin Stimulus</td>
<td>1402</td>
</tr>
<tr>
<td>Limit Line End Stimulus</td>
<td>1851</td>
</tr>
<tr>
<td>Limit Line Begin Response</td>
<td>1401</td>
</tr>
<tr>
<td>Limit Line End Response</td>
<td>1852</td>
</tr>
<tr>
<td>LimitMax Property</td>
<td>1853</td>
</tr>
<tr>
<td>LimitMin Property</td>
<td>1854</td>
</tr>
<tr>
<td>LimitTestFailed</td>
<td>1855</td>
</tr>
<tr>
<td>LimitTestXPosition Property</td>
<td>1856</td>
</tr>
<tr>
<td>LimitTestYPosition Property</td>
<td>1857</td>
</tr>
<tr>
<td>Limit Type</td>
<td>1858</td>
</tr>
<tr>
<td>Line Display</td>
<td>1859</td>
</tr>
<tr>
<td>ListData Property</td>
<td>1860</td>
</tr>
<tr>
<td>LoadCharFromFile</td>
<td>1862</td>
</tr>
<tr>
<td>LoadImpedance Property</td>
<td>1863</td>
</tr>
<tr>
<td>LoadPort</td>
<td>1864</td>
</tr>
<tr>
<td>LocalLockoutState</td>
<td>1865</td>
</tr>
<tr>
<td>Locator</td>
<td>1866</td>
</tr>
<tr>
<td>Lock Property</td>
<td>1867</td>
</tr>
<tr>
<td>LODeltaFound</td>
<td>1868</td>
</tr>
<tr>
<td>LODenominator</td>
<td>1869</td>
</tr>
<tr>
<td>LOFixedFrequency</td>
<td>1870</td>
</tr>
<tr>
<td>LOFrequencyDelta</td>
<td>1871</td>
</tr>
<tr>
<td>OBS_LogMagnitudeOffset</td>
<td>1872</td>
</tr>
<tr>
<td>LOName</td>
<td>1873</td>
</tr>
<tr>
<td>LONumerator</td>
<td>1874</td>
</tr>
<tr>
<td>LOPower</td>
<td>1875</td>
</tr>
<tr>
<td>LORangeMode</td>
<td>1876</td>
</tr>
<tr>
<td>LORangeMode cv Property</td>
<td>1877</td>
</tr>
<tr>
<td>Loss</td>
<td>1878</td>
</tr>
<tr>
<td>Property Name</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Loss (sourceCal)</td>
<td>1879</td>
</tr>
<tr>
<td>LOStage</td>
<td>1880</td>
</tr>
<tr>
<td>LOStartFrequency</td>
<td>1881</td>
</tr>
<tr>
<td>LOStartPower</td>
<td>1882</td>
</tr>
<tr>
<td>LOStopFrequency</td>
<td>1883</td>
</tr>
<tr>
<td>LOStopPower</td>
<td>1884</td>
</tr>
<tr>
<td>LowAmplitude Property</td>
<td>1885</td>
</tr>
<tr>
<td>LowFrequencyExtension Property</td>
<td>1886</td>
</tr>
<tr>
<td>LXIDeviceIDState</td>
<td>1887</td>
</tr>
<tr>
<td>MagnitudeOffset</td>
<td>1888</td>
</tr>
<tr>
<td>MagnitudeSlopeOffset</td>
<td>1889</td>
</tr>
<tr>
<td>MainToneIFBandwidth</td>
<td>1890</td>
</tr>
<tr>
<td>MarkCoupControlsMkrState Property</td>
<td>1891</td>
</tr>
<tr>
<td>MarkCoupMethPresetIsChan Property</td>
<td>1892</td>
</tr>
<tr>
<td>MarkCoupPresetIsOn Property</td>
<td>1893</td>
</tr>
<tr>
<td>Marker Annotation</td>
<td>1894</td>
</tr>
<tr>
<td>Marker Bucket Number</td>
<td>1413</td>
</tr>
<tr>
<td>Marker Format (all)</td>
<td>1895</td>
</tr>
<tr>
<td>Marker Format (indiv)</td>
<td>1715</td>
</tr>
<tr>
<td>Marker Interpolate(all)</td>
<td>1897</td>
</tr>
<tr>
<td>Marker Interpolate (indiv)</td>
<td>1807</td>
</tr>
<tr>
<td>Marker Number</td>
<td>1898</td>
</tr>
<tr>
<td>Marker Position</td>
<td>1899</td>
</tr>
<tr>
<td>Marker Readout</td>
<td>1900</td>
</tr>
<tr>
<td>MarkerReadoutResponsePlaces Property</td>
<td>1901</td>
</tr>
<tr>
<td>Marker ReadoutSize</td>
<td>1902</td>
</tr>
<tr>
<td>MarkerReadoutsPerTrace Property</td>
<td>1903</td>
</tr>
<tr>
<td>MarkerReadoutStimulusPlaces Property</td>
<td>1904</td>
</tr>
<tr>
<td>MarkerReadoutXPosition Property</td>
<td>1905</td>
</tr>
<tr>
<td>MarkerReadoutYPosition Property</td>
<td>1906</td>
</tr>
<tr>
<td>Markers Property</td>
<td>1907</td>
</tr>
<tr>
<td>Marker State</td>
<td>1908</td>
</tr>
<tr>
<td>MarkerSymbol Property</td>
<td>1909</td>
</tr>
<tr>
<td>MarkerSymbolsAboveTrace Property</td>
<td>1910</td>
</tr>
<tr>
<td>Marker Type</td>
<td>1911</td>
</tr>
<tr>
<td>Marker X-axis Value</td>
<td>1912</td>
</tr>
<tr>
<td>Marker Y-axis Value</td>
<td>1913</td>
</tr>
<tr>
<td>MatchCorrectPower</td>
<td>1915</td>
</tr>
</tbody>
</table>
Maximum Frequency 1916
MaximumFrequency (capabilities) 1917
MaximumFrequency (sourceCal) 1918
MaximumIFFrequency 1919
MaximumIterationsPerPoint 1920
MaximumNumberOfChannels 1921
MaximumNumberOfTracesPerWindow 1922
MaximumNumberOfPoints 1923
MaximumNumberOfWindows 1924
MaximumReceiverStepAttenuator 1925
MaximumSourceALCPower 1926
MaximumSourceStepAttenuator 1927
MaximumUncertaintyPoints Property 1928
MaxOutput Property 1929
MaxOutputState Property 1930
MaxPreciseTuningIterations 1931
MaxProduct 1932
Mean 1933
Measurement Property 1934
MeasurementClassProperties Property 1935
MeasurementClass 1936
Medium 1937
Memory Property 1938
MemoryMarkers Property 1939
MemShareEnabled Property 1940
MemShareName Property 1941
Message Text Method 1942
Minimum Frequency 1943
MinimumFrequency (capabilities) 1944
MinimumFrequency (sourceCal) 1945
MinimumIFFrequency Property 1946
MinimumNumberOfPoints 1947
MinimumReceiverStepAttenuator 1948
MinimumSourceALCPower 1949
MinOutput Property 1950
MinOutputState Property 1951
MixerCharacterizationFile Property 1952
Mode 1953
<table>
<thead>
<tr>
<th>Property</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>NoiseSourceCalKitType</td>
<td>1993</td>
</tr>
<tr>
<td>NoiseSourceCold</td>
<td>1994</td>
</tr>
<tr>
<td>NoiseSourceConnectorType</td>
<td>1995</td>
</tr>
<tr>
<td>NoiseSourceState</td>
<td>1996</td>
</tr>
<tr>
<td>NoiseTuner</td>
<td>1997</td>
</tr>
<tr>
<td>NoiseTunerIn</td>
<td>1998</td>
</tr>
<tr>
<td>NoiseTunerOut</td>
<td>1999</td>
</tr>
<tr>
<td>NominalIncidentPowerState</td>
<td>2000</td>
</tr>
<tr>
<td>NormalizePoint Property</td>
<td>2001</td>
</tr>
<tr>
<td>NormalizePoint SMC Property</td>
<td>2002</td>
</tr>
<tr>
<td>Number (meas)</td>
<td>2003</td>
</tr>
<tr>
<td>Number</td>
<td>2004</td>
</tr>
<tr>
<td>NumberOfFrequencyPoints</td>
<td>2005</td>
</tr>
<tr>
<td>NumberOfPoints (PowerCalRange) Property</td>
<td>2006</td>
</tr>
<tr>
<td>Number of Points</td>
<td>2007</td>
</tr>
<tr>
<td>Number of Points (Meas)</td>
<td>2009</td>
</tr>
<tr>
<td>NumberOfPorts</td>
<td>2010</td>
</tr>
<tr>
<td>NumberOfPorts(Testset)</td>
<td>2011</td>
</tr>
<tr>
<td>NumberOfPowerPoints</td>
<td>2012</td>
</tr>
<tr>
<td>NumberOfSweeps</td>
<td>2013</td>
</tr>
<tr>
<td>OccupiedBandCenter Property</td>
<td>2014</td>
</tr>
<tr>
<td>OccupiedBandPercent Property</td>
<td>2015</td>
</tr>
<tr>
<td>OccupiedBandPowerdBm Property</td>
<td>2016</td>
</tr>
<tr>
<td>OccupiedBandSpan Property</td>
<td>2017</td>
</tr>
<tr>
<td>OccupiedBandState Property</td>
<td>2018</td>
</tr>
<tr>
<td>Offset</td>
<td>2019</td>
</tr>
<tr>
<td>OffsetReceiverAttenuator</td>
<td>2020</td>
</tr>
<tr>
<td>OffsetSourceAttenuator</td>
<td>2021</td>
</tr>
<tr>
<td>OmitIsolation</td>
<td>2022</td>
</tr>
<tr>
<td>OneReadoutPerTrace</td>
<td>2023</td>
</tr>
<tr>
<td>Options</td>
<td>2024</td>
</tr>
<tr>
<td>OrientECALModule</td>
<td>2025</td>
</tr>
<tr>
<td>OutputChannel Property</td>
<td>2026</td>
</tr>
<tr>
<td>OutputFixedFrequency</td>
<td>2027</td>
</tr>
<tr>
<td>OutputPort</td>
<td>2028</td>
</tr>
<tr>
<td>OutputPorts calset</td>
<td>2029</td>
</tr>
<tr>
<td>OutputPorts</td>
<td>2030</td>
</tr>
<tr>
<td>OutputRangeMode</td>
<td>2031</td>
</tr>
<tr>
<td></td>
<td>2032</td>
</tr>
<tr>
<td></td>
<td>2033</td>
</tr>
</tbody>
</table>
OutputRangeMode cv Property
OutputSideband (conv)
OutputSideband
OutputStartFrequency
OutputStopFrequency
Parameter
Parameter_elo
ParameterList Property
Parent
PassFailLogic
PassFailMode
PassFailPolicy
PassFailScope
PassFailStatus
Path Property
PathCalMethod
PathConfigurationElement Property
PathElement Property
PathElements Property
PathThruMethod
Peak Excursion
Peak Threshold
PeakTo Peak
Percent Property
PerformPowerCalibration
Period
Phase Offset
PhaseAsFixture
PhaseControlMode Property
PhaseCorrectionData Property
PhaseCorrectionEnabled Property
PhaseDisplayMinLevel Property
PhaseIterationNumber Property
PhaseParameter Property
PhaseParameterModes Property
PhaseProcessState Property
PhaseReference Property
PhaseReferencePort Property
PowerOnDuringRetraceMode
PowerRangeType Property
PowerSensorCalKitType Property
PowerSensorCalKitType
PowerSensorConnectorType
PowerSensorConnectorType Property
PowerSlopeState Property
PowerSpinResolution Property
PowerStep
PowerStepsIn3DSweep Property
PowerSweepRetracePowerMode
PowerTableFilename Property
PreciseTuningTolerance
PreferInternalTriggerOnChannelSingle
PreferInternalTriggerOnUnguidedCal
PreferSourcePowerCalFromCalset Property
PresetMaxFrequency Property
PresetMinFrequency Property
PresetPowerState Property
PrimaryFrequency Property
PrimaryMode Property
PrimaryPeriod Property
PrimaryWidth Property
PropertyNames Property
PropertyNamesByMeasurementClass Property
PropertyValue Property
PropertyValues Property
Pulse4OutAsADCActivity Property
PulseGeneratorID Property
PulseGeneratorNames Property
PulseMeasMode Property
PulseOffAlcMode Property
PulseProfileStart Property
PulseProfileStop Property
RangeCount (Independent Power Cal) Property
RangeCount
rangeNumber
RangeOffset Property
ResolutionBW
ResolutionBW (SA) Property
ResolutionBWMax
ResolutionBWMin
ResolutionBWMode Property
ResolutionBWVideoBWRatio Property
ResponseCorrectedPorts Property
Reverse2PortAdapter Property
ReverseLinearPowerLevel
RFOffOnReceiverOverload
RoleDevice Property
SADATAThreshold Property
SADATAThresholdOption Property
SafeMode
SafeSweepDCParameter Property
SafeSweepCoarsePowerAdjustment
SafeSweepEnable
SafeSweepFinePowerAdjustment
SafeSweepFineThreshold
SafeSweepMaximumDCLimit Property
SafeSweepMaximumLimit Property
SAMTRference Property
SAMTRferenceFreqOption Property
SAResBWList Property
SaturationLevel Property
SAVectorAverage Property
SAVectorAverageOption Property
SAVideoAverageOption Property
SAVideoBandwidth Property
SB_BalPortNegative
SB_BalPortPositive
SB_SEPPort
SBalMeasurement
ScaleCouplingMethod Property
ScaleCouplingState Property
Scope
Search Function
SearchFailures Property
Source
SourceCWFrequency Property
Sources Property
SourceAttenuator Property
SourceAttenuator PR Property
SourceCount
SourceImpedance Property
Source Port
SourcePortCount
SourcePointCount Property
SourcePortFixedFrequency Property
SourcePortFrequencyOrder Property
SourcePortFrequencyState Property
SourcePortMode
SourcePortNames
SourcePortPhaseOrder Property
SourcePortPhaseState Property
SourcePortPowerOrder Property
SourcePortPowerState Property
SourcePortStartFrequency Property
SourcePortStopFrequency Property
SourcePower Property
SourcePowerCalPowerOffset
Source Power Correction
SourcePowerPointCount Property
SourcePowerRepeatCount Property
Source Power Option
Source Power State
SourcePullForSParameters Property
SourceRepeatCount Property
SourceStartFrequency Property
SourceStartPower Property
SourceStepAttenuatorStepSize Property
SourceStopFrequency Property
SourceStopPower Property
SourceSweepFirstDimension Property
SourceSweepType Property
SourceSweepType2 Property
Span
Span (Meas)
SpanBinsCount Property
SpanResolutionBWRatio Property
SParameterCalPorts Property
SpectrumCenterFrequency
SpectrumSpanFrequency
SpectrumStartFrequency
SpectrumStopFrequency
SSB_BalPortNegative
SSB_BalPortPositive
SSB_SEPort1
SSB_SEPort2
SSBMeasurement
Stage1Coefficients
Stage1Frequency
Stage1MaximumCoefficient
Stage1MaximumCoefficientCount
Stage1MaximumCoefficientSum
Stage1MinimumCoefficientCount
Stage2Coefficients
Stage2MaximumCoefficient
Stage2MaximumCoefficientCount
Stage2MaximumCoefficientSum
Stage2MinimumCoefficientCount
Stage3FilterType Property
Stage3FilterTypes Property
Stage3Parameter Property
Stage3ParameterMaximum Property
Stage3ParameterMinimum Property
Stage3Parameters Property
Standard Deviation
Standard For Class
StandardReceiverNoiseBWList Property
Start Frequency_CS
Start Frequency
Start Power
Start
SweepMode Property
SweepModeOption Property
SweepOrder
SweepSpeedMode
Sweep Generation Mode
Sweep Time
SweepTimeOption
SweepType (imd)
SweepType (ims)
Sweep Type
SweepType Property
System Impedance Z0
SystemName
Target Value
TestPortNames Property
Test Port Power
TestSetType
Text
TextFileEnabled Property
ThruCalMethod (FCA)
ThruCalMethod
ThruPortList
TimeOut Property
Title
Title (Channel) Property
Title State
Tolerance
TonePower
TonePowerSetAt Property
TonePowerStart
TonePowerStop
TotalIterations
TotalNumberOfPoints
Touchscreen
TraceHoldType Property
Trace Math
TraceMax
TraceTitle
vsa.ADCStacking Property  2559
vsa.CenterFrequency Property  2560
vsa.IFOffset Property  2561
vsa.LOSide Property  2562
vsa.StreamDataToFile Property  2563
vsa.VSAConnection Property  2564
WGCutoffFreq  2565
WideBandDectionState Property  2566
Width  2567
Window Number  2568
Window State  2569
XAxisAnnotation  2570
XAxisDomain Property  2571
XAxis Point Spacing  2572
XAxis Property  2573
XAxisStart  2574
XAxisStop  2575
YAxisAnnotation  2576
YScale  2577
Z0  2578

Methods
Abort  2579
AbortPowerAcquisition  2580
Acquire Cal Standard  2581
Acquire Cal Standard2  2583
AcquireCalConfidenceCheckECALEx  2585
AcquirePowerReadingsEx  2586
AcquireStep  2588
Activate  2589
Activate Marker  2590
Activate Window  2591
Add (channels)  2592
Add (measurement)  2593
Add (naWindows)  2597
Add (PowerLossSegment)  2598
Add (PowerSensorCalFactorSegment)  2599
Add (segments)  2600
Add External Device Method  2601
Add GuidedPowerSensors Method
Add Testset
AddPowerCalRange Method
AddSegment Method
Allow All Events
AllowChannelToSweepDuringCalAcquisition
Allow Event Category
Allow Event Message
Allow Event Severity
Apply
ApplyDeltaMatchFromCalSet
ApplyPowerCorrectionValuesEx
ApplySourcePowerCalibrationTo
AssignSourceToRole
AutoOrient Method
AutoPortExtMeasure
AutoPortExtReset
Autoscale
Averaging Restart
BandnoiseData Method
BandpowerData Method
Build Hybrid Kit
Calculate Error Coefficients
Calculate
Calculate cv Method
Change Parameter
CheckPower
Clear
Clear fifo
Close CalSet
Close VISAPassthrough Method
ComputeErrorTerms
ConfigEnhancedNB2
ConfigEnhancedNBIFAtten
ConfigNarrowBand3
Configurations
Configure
Continuous Sweep
<table>
<thead>
<tr>
<th>Method/Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Copy</td>
<td>2651</td>
</tr>
<tr>
<td>CopyFrom Method</td>
<td>2653</td>
</tr>
<tr>
<td>CopyToChannel</td>
<td>2654</td>
</tr>
<tr>
<td>Create SParameter Method</td>
<td>2655</td>
</tr>
<tr>
<td>CreateCalSet</td>
<td>2656</td>
</tr>
<tr>
<td>CreateCustomCal</td>
<td>2657</td>
</tr>
<tr>
<td>CreateCustomCalEx</td>
<td>2658</td>
</tr>
<tr>
<td>CreateCustomMeasurementEx</td>
<td>2660</td>
</tr>
<tr>
<td>CreateCalConfiguration Method</td>
<td>1506</td>
</tr>
<tr>
<td>Create Measurement</td>
<td>2668</td>
</tr>
<tr>
<td>DataToMemory</td>
<td>2672</td>
</tr>
<tr>
<td>Deembed Method</td>
<td>2673</td>
</tr>
<tr>
<td>DefineParameter Method</td>
<td>2675</td>
</tr>
<tr>
<td>Delete</td>
<td>2676</td>
</tr>
<tr>
<td>Delete Marker</td>
<td>2677</td>
</tr>
<tr>
<td>Delete All Markers</td>
<td>2678</td>
</tr>
<tr>
<td>DeleteAllSegments Method</td>
<td>2679</td>
</tr>
<tr>
<td>DeleteCalSet</td>
<td>2680</td>
</tr>
<tr>
<td>DeleteConfiguration</td>
<td>2681</td>
</tr>
<tr>
<td>DeleteParameter Method</td>
<td>2682</td>
</tr>
<tr>
<td>DeleteSegment Method</td>
<td>2683</td>
</tr>
<tr>
<td>Delete ShortCut</td>
<td>2684</td>
</tr>
<tr>
<td>Delta Marker</td>
<td>1557</td>
</tr>
<tr>
<td>Disallow All Events</td>
<td>2685</td>
</tr>
<tr>
<td>DiscardChanges</td>
<td>2686</td>
</tr>
<tr>
<td>DisplayNAWindowDuringCalAcquisition</td>
<td>2687</td>
</tr>
<tr>
<td>DisplayOnlyCalWindowDuringCalAcquisition</td>
<td>2688</td>
</tr>
<tr>
<td>Do Print</td>
<td>2689</td>
</tr>
<tr>
<td>DoECAL1PortEx</td>
<td>2690</td>
</tr>
<tr>
<td>DoECAL2PortEx</td>
<td>2691</td>
</tr>
<tr>
<td>DoneCalConfidenceCheckECAL</td>
<td>2693</td>
</tr>
<tr>
<td>DoReceiverPowerCal</td>
<td>2694</td>
</tr>
<tr>
<td>DoResponseCal Method</td>
<td>2696</td>
</tr>
<tr>
<td>Embed Method</td>
<td>2697</td>
</tr>
<tr>
<td>ENREmbedAdapter Method</td>
<td>2699</td>
</tr>
<tr>
<td>EnumerateCalSets</td>
<td>2700</td>
</tr>
<tr>
<td>EnumerateItems</td>
<td>2701</td>
</tr>
<tr>
<td>Execute</td>
<td>2702</td>
</tr>
</tbody>
</table>
Execute Shortcut
Exists Method
ExportCSVfile Method
Find VISAPassthrough Method
FrequencyAutoTune Method
FrequencySpanFull Method
GenerateGlobalDeltaMatchSequence
GenerateErrorTerms
GenerateSteps
GetAllSegments
Get AuxIO
Get Cal Standard
GetCalKitTypeString Method
GetCompatibleCalKits Method
Get CalManager
Get CalSetByGUID
Get CalSetCatalog
Get CalSetUsageInfo
Get Cal Types
Get Complex
GetConnectedPhaseReferences Method
GetConverter
GetDataBuffer Method
GetDataBufferCompact Method
Get DataByString
Get Data
Get ECALModuleInfoEx
GetEcalUserCharacterizer
Get ENRData
Get ErrorCorrection
Get Error Term
Get Error Term2
Get Error Term By String
Get Error Term Complex
Get Error Term Complex2
Get Error Term Complex By String
Get Error Term List
Get Error Term List2

2703
2704
2705
2706
2708
2709
2710
2711
2712
2714
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2727
2728
2729
2731
2733
2736
2739
2740
2741
2742
2743
2745
2747
2749
2751
2753
2755
2757
GetErrorTermStimulus Method
Get ExtendedCallInterface
Get ExternalTestSetIO
Get Filter Statistics
Get Guid
get InputVoltageEX
Get Input1
GetIPConfigurationStruct
Get IsolationPaths
GetLibraryFunctions Method
GetLicenses Method
Get MaterialHandlerIO
Get NAComplex
Get NumberOfGroups
Get Output
Get Output Voltage
Get OutputVoltage Mode
Get Paired Data
Get Port
Get PortC Data
Get PortNumber
GetRaw2DData
GetRaw2DDataIm
GetRaw2DDataRe
GetRxLevelingConfiguration
GetSourceByRole
GetSourceRoles
Get Reference Marker
Get Required Eterm Names
Get Scalar
Get Shortcut
Get SnPData
Get SnpDataWithSpecifiedPorts
Get SourcePowerCalDataEx
Get SourcePowerCalDataScalarEx
Get Standard
Get Standard By String
Get Standard Complex
Get Standard Complex By String 2813
Get StandardsList 2814
Get Standard List2 2816
Get StandardsForClass 2817
Get StepDescription 2819
Get SupportedALCModes 2820
Get Test Result 2821
Get Trace Statistics 2823
Get X-Axis Values 2824
Get XAxisValues (Meas) 2826
Get X-axis Values Variant 2827
GetXDataBuffer Method 2828
GetXDataBufferCompact Method 2830
GetVISATimeout Method 2832
Has CalType 2833
Hold 2835
Hold (All Chans) 2836
ImportCSVfile Method 2837
ImportDataSet Method 2838
ImportLibrary Method 2840
Initialize 2841
InitializeEx Method 2842
Initialize ECal 2844
Interpolate Markers Method 1897
IsLibraryImported Method 2845
Item 2846
Item (Independent Power Calibration Port) Method 2848
LANConfigurationInitialize 2849
LaunchCalWizard 2850
Launch Dialog 2851
LaunchPowerMeterSettingsDialog 2853
Load Configuration 2854
LoadENRFile 2855
LoadFile ED Method 2856
LoadFile 2857
LoadTheme 2858
Manual Trigger 2859
Move 2860
NetworkPortMap
Next IF Bandwidth
Number of Groups
Open CalSet
Open VISAPassthrough Method
Output SNP From ECal Method
Parse
Preset (app and chan)
Previous IF Bandwidth
Print To File
Put Complex
Put Data Complex
PutENRData
Put ErrorTerm
Put ErrorTerm2
Put Error Term By String
Put ErrorTerm Complex
Put ErrorTerm Complex2
Put Error Term Complex By String
PutErrorTermStimulus Method
Put Formatted Scalar Data
Put NACComplex
Put Output
Put Output Voltage
Put Output Voltage Mode
Put Port
Put PortCData
Put Scalar
Put Shortcut
Put SourcePowerCalDataEx
Put SourcePowerCalDataScalarEx
Put Standard
Put Standard By String
Put Standard Complex
Put Standard Complex By String
Quit
RangeCount Method
ReadBinary Method
ReadBinaryCompact Method
Read Data
Read Raw
ReadStream VISAPassthrough Method
ReCalculate Method
Recall
Recall Kits
Recall (uncert) Method
Remove
RemoveAll Method
RemoveChannelNumber
RemoveItem
RemoveLibrary Method
Reset
Reset (CalAll) Method
Reset (Independent CalAll) Method
Reset (PhaseRef) Method
Reset (Power Range) Method
Reset VISAPassthrough Method
ResetLOFrequency
ResetNoise Method
ResetNoiseForAllPorts Method
ResetPortValues Method
ResetRepeatability Method
ResetTheme Method
ResetTuningParameters
Restore Cal Kit Defaults
Restore Cal Kit Defaults All
Restore Defaults
Resume
Save
Save (CalSet)
Save CalSets
SaveCitiDataData
SaveCitiFormattedData
SaveData Method
SaveENRFile
Save File
SaveFile ED Method
SaveToDiskMemory Method
SaveToECal
Save Kits
Save uncert Method
SearchCompressionPoint
SearchPowerNormalOperatingPoint Method
SearchPowerSaturation Method
Search Filter Bandwidth
Search Max
Search Min
Search Next Peak
Search Peak Left
Search Peak Right
Search Target
Search Target Left
Search Target Right
SegmentCalculate Method
SelectCalSet
Set All Segments
Set BBPorts
SetBPort Method
SetBSPorts Method
SetBSSPorts Method
Set Cal Info
SetCallInfoEx
SetCustomDUTTopology Method
Set Center
Set CW
SetCWFreq
SetDutPorts
Set Electrical Delay
Set FailOnOverRange
Set IsolationPaths
SetIPConfiguration
Set Power Acquisition Device
Set Frequency LowPass
SetPortMap
Set Reference Level
Set SBPorts
Set SSBBPorts
SetupMeasurementsForStep
Set StandardsForClass
Set Start
Set Stop
SetVISATimeout Method
Show Marker Readout
Show Status Bar
Show Stimulus
Show Table
Show Title Bars
Show Toolbar
Single
Store
StoreConfiguration
StoreTheme Method
StringToNACalClass
StringtoNAErrorTerm2
SweepOnlyCalChannelDuringCalAcquisition
toSA Method
TraceHoldClear Method
UserPreset
UserPresetLoadFile
UserPresetSaveState
WriteBinary Method
Write Data
Write Raw
WriteSnPData Method
WriteSnPFileWithSpecifiedPorts
WriteString VISAPassthrough Method
WriteUncertaintyFile Method
ZeroTermsInS4PFile Method

Events
OnCalEvent
OnChannelEvent
OnDisplayEvent 3053
OnHardwareEvent 3055
OnMeasurementEvent 3057
OnSCPIEvent 3059
OnSystemEvent 3061
OnUserEvent 3063

Examples

Active (Hot) Parameters 3064
C Example 3065
Cal All Independent Calibration Channels 3070
Cal All Multi-Channel Independent Calibration Channels 3072
Cal All SMC Split Cal 3074
CalSet_Examples 3076
Getting Trace Data from the Analyzer 3078
Perform a Guided Cal using COM 3081
Perform a Source Power Cal 3085
Independent Power Calibration 3089
Perform a Guided Cal using C 3091
Perform an Unguided Cal using COM 3098
Perform an Unknown Thru or TRL Cal 3101
Perform Global Delta Match Cal 3103
Perform a Guided Cal with CSharp 3104
Perform an ECal 3109
Perform an ECal User Characterization 3112
Perform a Comprehensive Guided 2-Port Cal 3118
Perform an ECAL Confidence Check 3125
Power Meter Uncertainty Using Standard Uncertainties 3128
Writing Cal Set Data using COM 3133
Upload a Source Power Cal 3135
Upload Segment Table 3138
Create Multiple Instances of Calibrate All Channels 3143
Create and Cal an SMC Measurement 3146
Create and Cal a VMC Measurement 3150
Create an SMC Fixed Output Meas 3155
Create a Segmented Sweep for Mixers 3158
Use Existing Power Cal for SMC 3162
Create a Balanced Measurement 3165
Create a Multi-Dimensional Sweep 3169
Create a PMAR Device and Measurement
Create a Wideband Pulsed Measurement using the VNA-X
Create an IM Spectrum Measurement
Create an iTMSA Measurement
Create and Cal a Gain Compression Measurement
Create and Cal a GCX Measurement
Create and Cal a Noise Figure Measurement
Create and Cal an NFX Measurement
Create and Cal an IMD Measurement
ENR File Management Example
Events with C
FOM Examples
Limit Line Testing Example with COM
Modify Display Colors
E5091 Testset Control
Errors and the SCPIStringParser Object
External Testset Control
PathConfiguration Example
PNA-X Create a Pulsed Measurement
Power Meter Uncertainty Using Standard Uncertainties
Power Range Example
Setup Basic Measurements
Setup Compression Marker
Set Up Embedded LO Measurement
Setup FastCW and FIFO
Setup Noise Figure Port Mapping
Setup Phase Control
Setup PNOP and PSAT Marker Search
Setup Receiver Leving
Show Custom Window during Calibration
Spectrum Analyzer
Events Example
Uncertainty on Power Meter
User Defined Power Meter Uncertainty File
Upload_Segment_Table_in_C
Using C#
Concepts
Configure for COM-DCOM Programming
COM Fundamentals 3282
Getting a Handle to an Object 3287
Collections in the Analyzer 3291
COM Data Types 3293
VNA Automation Interfaces 3296
Working with the Analyzer's Events 3298
Read and Write Calibration Data using COM 3303
C and the COM Interface 3305
Using .NET 3309

SCPI

Commands
SCPI Command Tree 3311
Common Commands 3313
Abort 3317
Automatic Fixture Removal (AFR) 3318

Calculate
Correction 3340
Custom 3352
Data 3362
DTOPology 3374
Equation 3376
Filter 3381
Format 3388
FSimulator 3392
FSimulator Draft 3397
FSimulator Active 3437
Function 3473
GCData 3479
GCMeas 3484
GroupDelay 3488
Limit 3491
Marker 3502
Math 3527
Measure 3530
AM Distortion 3554
BLIMIT 3558
Correction 3563
DATA 3571
FILter 3586
FUNCTION 3592
GCDATA 3598
GCMeas 3603
GDELAY 3607
LIMIT 3609
MARKer 3619
OFFSET 3668
PARAMeter 3670
RLIMIT 3681
SA 3686
SMoothing 3703
TRANSform 3705
X 3715
Uncertainty 3721
Mixer 3727
Normalize 3728
Offset 3731
Parameter 3734
RData 3747
SA 3749
Smoothing 3764
TDR 3767
Trace Hold 3800
Transform 3802
Uncertainty 3813
X Values 3819
Calpod 3822
Control 3830
Control Multiplexer 3864
CSET 3869
Display 3883
Format 3940
Hardcopy 3943
Initiate 3956
LXI 3959
Memory 3960
Output 3981
Route 3983
Sense
  Active (Hot) Parameters 3984
  Amplifier 3990
  Average 3996
  Bandwidth 3999
  Class 4002
  Control 4003
  Correction 4021
    Cal Kit 4061
    Cal Stds 4076
    Cal Sets 4104
    Extensions 4124
    Guided Cal 4140
    IMD 4189
    Session 4195
    SMC 4201
    VMC 4213
  Couple 4224
  DC 4226
  DIQ 4230
  Distortion Measurement 4259
  DUTControl 4304
  FOM 4315
  FOMSegment 4326
  Frequency 4337
  Gain Compression 4343
  IF (PNA-X) 4366
  IMD 4377
  IMS 4401
  Mixer 4416
  MixerEmbedLO 4442
  Multiplexer 4456
  Noise 4472
  Offset 4494
  Path 4498
  Phase Noise 4504
  Power 4514
Pulse 4516
Roscillator 4530
SA 4534
Segment 4621
Source 4653
Sweep 4657
SWITCh 4673
TDR 4704
XAxis 4710
Service 4711
Source 4713
Source Modulation 4733
Source Power Correction 4848
Status Register 4869
System 4890
CalAll 4938
CalPhase 4954
Capability 4962
Communicate 4992
Config mmWave 5015
ConfigExtDevice 5034
ConfigExtDC 5045
ConfigExtPMAR 5058
ConfigExtPulseGen 5072
CorrIntLinear 5077
FIFO 5080
Preferences 5084
TDR 5108
Uncertainty 5110
Trigger 5118

Examples
SCPI Example Programs 5142
Automatic Fixture Removal (AFR) 5146
AFR Using One Differential 2X THRU 5146
AFR Using One Differential OPEN 5148
AFR Using One Single Ended 2X THRU 5150
AFR Using One Single-Ended OPEN 5152
Active (Hot) Parameters 5154
*ESR? Sweep Complete
Cal All Channels Calibration
Catalog Measurements using SCPI
Channels, Windows, and Measurements using SCPI
Control, Talk, Listen using SCPI
Create a Balanced Measurement using SCPI
Create a measurement using SCPI
Create a Multi-Dimensional Sweep
Create a Narrowband Point-in-Pulse Measurement_PNA-X
Create a Narrowband Pulse Profile Measurement - PNA-X
Create a PMAR Device and Measurement
Create a Swept IMDX Measurement
Create a Wideband Pulse Measurement
Create an FOM Measurement
Create an iTMSA Measurement
Create an SMC Fixed Output Meas
Create_and_Cal_a_GCA_Measurement
Create and Cal a GCX Measurement
Create and Cal a Noise Figure Measurement
Create and Cal a VMC Measurement
Create and Cal an IMD Measurement
Create and Cal an NFX Measurement
Create and Cal an SMC Measurement
Create and Cal Multiple SMC Channels
Create a Mixing Plan for a Dual-Stage, Fixed-Output Converter
Create New Cal Kit using SCPI
Custom Power Meter Driver
ECALConfidence Check using SCPI
Establish a VISA Session
External Test Set using SCPI wLink
Fixture Simulator
  Create fixturing function (impedance conversion and port matching)
  Create fixturing function (2-port deembed, port extension, port matching, impedance conversion)
  Create fixturing function (port impedance conversion and port extension)
Getting and Putting Data using SCPI
GPIB Pass Through
GPIB using Visual C
Guided 2-Port or 4-Port Cal
Load Eterms into Cal Sequence
Modify a Calibration Kit using SCPI
Modulation Distortion Measurement
  Measurement Setup
  Measurement Setup Converters
  Create Modulation File
  Display Data Setup
  Source Modulation Calibration
Perform a Simple Source Power Cal
Perform a Source and Receiver Power Cal using SCPI
Perform a Source Power Cal with TWO Sensors
Perform an ECal User Characterization
Perform an Unguided Cal on a 4-Port PNA using SCPI
Perform Global Delta Match Cal
Perform Guided 1Port
Perform Guided 2-Port Comprehensive Cal
Perform Guided ECal
Perform Guided Mechanical Cal
Perform Guided TRL Calibration
Perform an Unguided 1-Port Cal on Port 2
Perform Unguided 2-Port Mech Cal
Perform Unguided ECAL
Perform Unknown Thru or TRL Cal
Phase Noise Measurement
  Setting Up a Phase Noise Measurement
  Setting Up a Source
  Spurious Measurement
  Integrated Noise Measurement
  Spot Noise Measurement
Power Meter Uncertainty
Pulse Narrowband Setup
Setup FastCW and FIFO
Setup Markers
Setup Noise Figure Port Mapping
Setup Phase Control
Setup PNOP and PSAT Markers
Setup RxLeveling
Setup Sweep Parameters using SCPI 5419
Setup the Display using SCPI 5420
Show Custom Window during Calibration 5422
Sliding Load Cal using SCPI 5426
SmartCal_Noise Figure_USBNoiseSource 5427
Socket Client 5429
Spectrum Analyzer 5438
Status Reporting using SCPI 5441
TDR Programming Examples
  TDR/TDT Measurement 5443
  Simulated Eye Diagram 5448
  2 Channel Measurement 5452
Transfer Data using GPIB 5461
Triggering the VNA using SCPI 5463
Unguided_Cal_on_Multiple_Channels 5470
Upload and Download a Segment List 5475
Uploading a Source Power Cal using SCPI 5482
VISA Pass Through 5486
Concepts
  GP-IB Fundamentals 5487
  The Rules and Syntax of SCPI Commands 5492
  How to Configure for GPIB, SCPI, and SICL 5497
Getting Data from the Analyzer 5503
Understanding Command Synchronization 5508
Calibrating the PNA Using SCPI 5515
The PNA as a USB Device 5521
Reading the Analyzer's Status Registers 5523
Referring to Traces, Measurements, Channels, and Windows Using SCPI 5528
Remote Control of SCPI USB Devices Connected to a PNA 5530
Configure for VISA and SICL 5534
VEE Examples
  VEE Pro Runtime 5538
  Basic Control VEE 5539
    ECal with Confidence Check using VEE 5541
Data Access Map
  DataMapSet 5543
IO Connectors
  Interface Control 5544
External TestSet IO Connector 5550
Material Handler IO Connector 5557
Pulse IO (PNA-X) 5570
Power IO (PNA-X) 5572
Programming Guide 953
Command Finder 5575
New Programming Commands 5576
COM versus SCPI 5697
Remotely Specifying a Source Port 5700
Shut Down the PNA Remotely 5702
LXI Compliance 211
Using Macros 460
Code Translator 5703
Using Fixture Simulator 5704

Applications
Active Hot Parameters 5709
Antenna Features 5724
Differential I/Q 5730
Fast CW Mode - Option S93118A/B 5724

Frequency Converter Application 5747
  FCA Overview 5747
  SMC Measurements 5763
  MixerConverter Setup 5781
  SMC plus Phase 5796
  Phase Reference Calibration 5802
  How to make an SMC Fixed Output Measurement 5814
  SMC with a Booster Amp 5821
  VMC Measurements 5824
  How to make a VMC Fixed Output Measurement 5842
  Embedded LO 5850

Frequency Offset
  Frequency Offset 5857
  Frequency Converting Device Measurements 5868
  Frequency Offset Calibration 5869
  Conversion Loss 5872
  Conversion Compression 5876
  Isolation 5879
  Harmonic Distortion 5883
Set Up a Band Power Measurement  6123
Set Up a NPR Measurement  6129
Set Up a Multi-Band Measurement  6137
Set Up a Power Sweep Measurement  6141

Noise Figure
   Noise Figure Application  6147
   Noise Figure on Converters  6182
   Noise Figure Cal  6194
   Noise Figure and TRL Cal  6206

Pulsed Application  6211
WideBand Pulsed App  6227

Phase Noise
   Overview  6229
   Starting and Exiting Phase Noise  6242
   Configuring Phase Noise  6244
   Displaying Phase Noise Parameters  6259
   Phase Noise Marker Search  819
   Calibration  506
   Phase Noise Measurement
      Setting Up a Phase Noise Measurement  6261
      Spurious Measurement  6268
      Integrated Noise Measurement  6274
      Spot Noise Measurement  6278
      AM Noise Measurement  6281

Spectrum Analyzer
   Spectrum Analyzer  6283
   SA Analysis Markers  819
   Noise Power Ratio (NPR)
      Noise Power Ratio (NPR) Settings  6328
      Create Modulation Files Settings  5995
      Source Modulation Calibration  6028

Swept IMD and IM Spectrum
   IMD Applications  6333
   Swept IMD and IM Spectrum Concepts  6335
   Swept IMD  6345
   IM Spectrum  6362
   Swept IMDx for Converters  6371
   IMx Spectrum for Converters  6382
Time Domain
Enhanced Time Domain Analysis
  Overview
    Features and Limitations
    TDR Quick Start
      TDR/TDT Measurement
      Simulated Eye Diagram
    TDR Screen Area
    TDR Measurement Considerations
Starting and Exiting TDR Application
Setting Up the Measurement
  Using Setup Wizard
  Performing Manual Setup
  Performing Error Corrections
Making Measurements
  Setting Up Parameters on Each Trace
  Controlling Trigger
  Using Scale/Zooming
  Using Marker and Marker Search
  Using Data and Memory
  Using Gating
  Using Trace Control
  Hot TDR Measurement
Eye Diagram and Mask Test
  Performing Eye Diagram Measurements
  Selecting Bit Pattern
  Using Mask Test
  Available Masks
Storing Data and Setting
  Saving/Recalling Setting
  Saving Data
  Saving Touchstone Data
  Saving Displayed Image
Advanced Waveform Analysis
  Overview
  Using Jitter Injection
  Using Emphasis
  Using De-embedding
Using Equalization
Advanced Mode
Overview
Activating and Deactivating Advanced Mode
Frequently Used Functions in Advanced Mode
Measurement Examples
TDR - PCB Impedance Measurement Example
2 Channel Measurement Example
Connecting with PathWave Vector Signal Analyzer (89600 VSA)

Low Frequency Extension (LFE)
Overview
Starting and Exiting LFE
LFE Calibration

Millimeter Wave Configuration
Overview
Supported Applications
Supported Configurations
Starting and Exiting Millimeter Wave Mode
Millimeter Configuration
Calibration
Broadband System Measurement Setup
Banded System Measurement Setup
Mixer Measurements

Networking the VNA
Drive Mapping

Product Support
Troubleshooting
List of Error Messages
About Error Messages
Accessories
USB to GPIB Adapter
Firmware Update
Configurations and Options
Option Enable
Instrument Calibration
Other Resources
SCPI Errors
Technical Support
Software Support 6739
Licenses 6743

Diagnostic Tools and Adjustments
  Adjustment Overview 6745
  System Verification 6751
  Operators Check 6762
  Display Test 6766
  10 MHz Reference Adjust 6768
  Source Adjustment 6771
  Phase-Lock IF Gain Adjust 6773
  Receiver Cal 6776
  Receiver Display 6779
  Mechanical Counter 6780

IF Access
  IF Path Configuration 6782
  N5251A mmWave Configuration 6792
  mmWave Measurement w/No Test Set 6809
  N5264B Measurement Receiver 6816

System Settings
  Configure External Devices 6820
    Configure an External Device 6820
    Configure an External Source 6834
    Configure a DC Device 6840
    External DC Meter Data Conversion 896
    Configure a Power Meter As Receiver 6848
    Configure an External Pulse Generator 6860
    Configure an External SMU 6865
    Synchronize an External PSG Source 6873

IO Connectors
  Interface Control 5544
  External TestSet IO Connector 5550
  Material Handler IO Connector 5557
  Pulse IO (PNA-X) 5570
  Power IO (PNA-X) 5572

Test Set Control 6875
Error Messages 6670
External Testset Control 6880
Display Colors 6894
Display Update 6898
Frequency Blanking 194
Interface Control 5544
Mechanical Devices 6899
Preferences 200
Power Limit and Power Offset 6902
Receiver Temperature 6906
System Impedance 6907
Global Source 399
Windows File Locations 109

Tutorials
App Notes 6908
Network Analyzer Basics 6911
Connector Care 115
ESD Protection 6912

Measurements
Absolute Output Power 6913
Active Probing 6916
AM-PM Conversion 6918
Amplifier Measurements 6923
Antenna Measurements 6926
Balanced Measurements 6929
Complex Impedance 6938
Comparing the Delay Functions 6941
Deviation from Linear Phase 6943
Directional Coupler Directivity Measurement 6947
External Source Control 6873
FIFO and other Antenna Features 5724
Filter Measurements 6951
Gain and Flatness 6956
Gain Compression 6959
Gated Measurement 6964
Group Delay 6969
High-Gain Amplifier Measurements 6976
High Power Measurements using a VNA-X 6978
Modulation Distortion Setup 6039
Noise Power Ratio (NPR) Measurement 6981
Phase Measurements 6996
Reverse Isolation 7000
Reflection Measurements 7003
SA Amplifier Harmonics Measurement 7008
SA Converter Spurious Measurement 7014
Time Domain Measurements 6391
Synchronize an External PSG Source 6873

GUI Reference
Avg BW 7022
Cal 7023
Channel 7024
Display 7025
Format 7027
Freq 7028
Macro 7029
Marker 7030
Math 7032
Meas 7034
Power 7035
Preset 7036
Save Recall 7037
Scale 7038
Search 7040
Setup 7043
Sweep 7044
System 7045
TDR 7048
Trace 7049
Trigger 7050
Undo 7051
Glossary 7052
Specifications 7078
Critical Information

What's New
VNA Applications
Measurement Tutorials
Links to VNA App Notes

VNA Connectivity
Product Support / Specs
Links to YouTube Videos

Still looking for answers?

Post your question at the Keysight Discussion Forums

See the very latest online PNAHelp at http://na.support.keysight.com/pna/help/
What's New

Notes

- The latest version includes changes from all previous versions.
- See New Programming Commands
- To see the code version that is currently installed on the VNA, click Help, then About NA...


- Save/Load mixer files in CSV format
- Smart Cal Ports
  - Use existing SrcCal when measuring cal standards
- Pulse Setup Primary Clock
- Support for U1832A/B/C/D and U1833A/B/C/D Noise Sources

What's New in VNA Code Version A.15.30.xx

- Software Support Licensing
- Updates to Fixture Simulator
- Compensate Only For De-Embeds (Fixture Generator)
- Modulation Distortion: Select font size of the Distortion Table

What's New in VNA Code Version A.15.05.xx

- SA Phase Stitching
- Added function to **Phase functions** for Gain Compression / Gain Compression for Converters applications

**What's New in VNA Code Version A.15.00.xx**

- Support for CPU 3.5
- Added 20 MHz Reference
- Added **PathWave Calibration Advisor**
- **IF Response Adjustment** (applies ONLY to instruments with new DDS source)


- **IF BW Shape**
- Enhance the **Fixture Simulator**
- **LO Feedthru Monitor**
- **Simulator** (Option S94050B and S94051B)
- **Residual Noise**
- Option S93072B Arbitrary Waveform Generation on XSB Port
- Added **Ext. Source Port** to Phase Control
- Added **Phase Control functions** to Differential I/Q application
- Added **Phase functions** to Gain Compression / Gain Compression for Converters applications
- Support for **P9336a + PSG external driver** (P9336A USB I/Q Arbitrary Waveform Generator with E8267D PSG Vector Signal Generator)
- Support for Waveguide Cal Kits

**What's New in VNA Code Version A.14.60.xx**

- **Multiline TRL Calibration**
- Support for **P9336A USB I/Q Arbitrary Waveform Generator**
• Support for USB Noise Source
• Connecting with PathWave Vector Signal Analysis (89600 VSA)
• Edit Multitone


• Phase Noise application (Option S93031xB)
• Modulation Distortion application enhancements:
  • Power Sweep
  • LO Feedthru Cal
• Global Source
• TMSA Enhancements
• Drivers for External Compound Sources


• Support for 64 bit 835x application
• Option XSB 3rd RF Source (can be enabled by firmware license and requires Option 422 or 423)
• Option UNY Enhanced Phase Noise (can be enabled by firmware license)
• Point leveling mode for receiver leveling
• Up to 500 Windows
• Up to 100 Traces per Window
• Enhancements to Correction Methods
• Enhancements to Balanced Setup
• Enhancements to Integrated True Mode Stimulus Application (iTMSA)
What's New in VNA Code Version A.13.95.xx

- Modulation Distortion Converters
- AM Distortion
- Trace deviation


- Modulation Distortion application enhancements (Option S93070xB)
- Receiver IF Cal for Modulation Distortion measurements
- Modulation Distortion Swept Cal Power
- Cal All - Split Cal and Independent Calibration Channels

What's New in VNA Code Version A.13.60.xx

- Modulation Distortion application (Option S93070xB)
- DC limits added to SMART Sweep Safe Mode
- Spectrum Analysis enhancements - dual-band configuration
- Device Expert (DE)
- Mechanical Counter


- TDR mode frequency limits
- Noise Figure Option S93027B (directly control mechanical tuners)
- Support for U205xXA/U206xXA Series USB Power Sensors

What's New in VNA Code Version A.13.50.xx
- Noise Power Ratio measurements
- Support for new Low Frequency Extension options
- Banded millimeter wave installation calibration
- Calibration with power meter uncertainties
- New SA Analysis Markers
- TDR mode frequency limits
- Support for Windows 10


- 1-port balanced measurements for iTMSA
- Support for external DC meter for SA channel measurements
- Active Hot Parameters application (Option S93110A/B, S93111A/B)
- Support for VDI PM5 power meter


- Enhanced Time Domain Analysis (Option S93011A/B)

**What's New in VNA Code Version A.13.00.xx**

- Support for Windows 10 operating system

**What's New in VNA Code Version A.12.90.xx**

- Multi-dimensional sweep on a spectrum analyzer channel
  
  - SCPI
  
  - COM
- CalPod temperature correction
- *ESR? sweep complete programming example


- Independent Power Calibration
- Automatic Fixture Removal (AFR) SCPI commands
- Maximum channels increased to 500
- Added ISegment3 and ISegments6 interface commands

What's New in VNA Code Version A.12.80.xx

- N5290A/N5291A Broadband Network Analyzer
- Low Frequency Extension (Option 205/425)
- Spectrum Analyzer Coherence and Data export features

What's New in VNA Code Version A.12.70.xx

- Modernized Graphical User Interface
  - Icons for adding or deleting trace/channel
  - Tabbed soft panel enabling quick access to desired functions
  - Access popup menus with long press or right click
  - Drag & drop trace/channel/window with finger or mouse
  - Easily make complex setups and calibrations using Wizards
  - Customize the icons to display in the top and bottom areas
  - Register frequently-used softkeys to Favorite menu
  - Group windows into Sheets
  - Magnify (zoom) a portion of the display
• Copy setups and user calibration data to other channels
• Improved Segment sweep
• Marker Search
  • Peak Search
  • Multi-Peak Search
  • Target Search
  • Multi-Target Search
  • Bandwidth Search
  • Notch Search
• Unique limit test functions
  • Ripple limit
  • Bandwidth limit
  • Point limit
• SCPI Parser Console for monitoring remote interface
• Port subset correction (devolve calibration)
• Spectrum Analyzer enhanced features
• Power Range SCPI and COM commands

What’s New in PNA Code Version A.10.49.09

• Cal All Channels SCPI programming examples
• Apply to all channels button in Calibrate All Selected Channels dialog

What’s New in PNA Code Version A.10.49.07

• Support for N755xA ECAl modules
What’s New in PNA Code Version A.10.49.05

- Read DC at Compression Point in Smart Sweep
- Memory Interpolation
- CalPod options (301, 302, and 304) are now standard
- CalPod - OSL Averages
- CalPod - VNA CalPod Utilities

What’s New in PNA Code Version A.10.49

- SA Power Sweep
- DC Source Sweep
- SA Gated Measurement Quick Setup

What’s New in PNA Code Version A.10.45

- Gated SA
- SA broadband and banded millimeter-wave measurements
- Support for Spectrum Analyzer measurements on PNA-L
- Cal All for millimeter wave measurement classes
- Support for UXG signal generators

What’s New in PNA Code Version A.10.40

- Spectrum Analyzer Application
- Dynamic Uncertainty for S-Parameters
- Frequency Step Setting
- 15 General-Purpose Markers per Trace
- Marker Symbols Above Trace
- Noise Marker Format
- Coupled Markers Method
- Limit Line Pass/Fail Indicator Positioning
- Trace Hold

Preferences

- Treat Marker 10 as Reference Marker
- Draw Limit Lines in Red
- Coupled Markers (3 preferences)

What's New in PNA Code Version A.10.25

- Differential IQ
- Narrowband Compensation to Noise Figure measurements
- CPM Direct Receiver Calibration

What's New in PNA Code Version A.10.20.03

- Cal All for External Sources
- Narrowband Compensation for Noise Figure

What's New in PNA Code Version A.10.20

- Cal Plane Manager
- Auto Fixture Removal

What's New in PNA Code Version A.10.15

- New N5249A PNA-X model (Configuration | Specs)
What's New in PNA Code Version A.10.00

- Windows® Embedded Standard 7
- HiSLiP
- Integrated CalKit Editor
- Increased Number of Points to 100K
- Cal Kits saved and recalled using *.xkt format
- U8480 Power Sensors Supported
- Mechanical Devices: Copy Active Channel
- Search Within (for Markers)
- Dialog Transparency
- Move App to Back
- System time on PNA Status Bar
- Right-click menus on Status Bar
- Elimination of 3 Toolbars
- Enhanced MATLAB® functionality

What's New in PNA Code Version A.09.90

- SMC Phase Reference Cal from 10 MHz
- CalPod as ECal
- External SMU Device
- Equation Editor enhancements
  - New marker functions
  - Use Short Names

What's New in PNA Code Version A.09.85
- Support for U2020 X-Series USB Power Sensors
- Delta Match Calibration required on ALL N5231A, N5232A, and N5239A models.
- Unguided TRL Cal allowed on all PNA 4-port models
- Noise Parameters

**What's New in PNA Code Version A.09.80**

- **Standard Measurements**
  - Calibrate All Channels
  - SE => Balanced Topology
  - Increased Segments in Power Sensor Loss Table
  - EXG Sources supported
  - CalPod

- **Application Enhancements**
  - SMC with Phase Reference
  - Noise Receiver Cal with Power Sensor
  - PNA models with 50 GHz Noise Receivers

**What's New in PNA Code Version A.09.60**

- New N523xA models

**What's New in PNA Code Version A.09.50**

**Standard Measurements**

- Unguided Cals can access 95 Cal Kits
- Preference setting to list Recall files
- YouTube Videos
• Display Menu changes
• Undo/Redo
• Quick Start
• Quickly change Scale, Reference Level, and Position
• Redesigned Receiver Leveling dialog
• New External Device configuration:
  • Pulse Generators
  • DC Sources and Meters
  • DC Source Control dialog

Application Enhancements

• External Pulse Generator in Integrated Pulse App
• IMSpectrum in mmWave

What’s New in PNA Code Version A.09.42

May 2013: Support for U2020X Power Sensors

• New N522x Models

Standard Measurements - available on all Models / Options

• Drag a trace to another window

Application Enhancements

• Copy Channels on all Applications
• "Src 2 out Port 2" factory configuration on PNA-X Opt 423.
• IMD f2 Tone using External Source / Combiner
• IMD and IM Spectrum Tone Power Leveling settings
- IMD, IMDx, and IM Spectrum "Min" and "Max" parameters
- Use a Power Table with mmWave SMC Measurements
- mmWave; Mixer mode - 2-port test set on 4-port PNA
- Guided Power Cal for SMC
- ESG and PSG Sources for Phase Control

What's New in PNA Code Version A.09.33

New Options

- Source Phase Control - Opt 088

Application Enhancements

- FCA Update - Opt 082, and 083

Standard Measurements - available on all Models / Options

- Security for External Sources
- 2-Port and 4-Port Fixture Extrapolation and Reverse Ports.
- Phase Coherent "R over R" measurements
- Use Multiple Power Sensors for Guided Power Cal
- Perform Source Power Cal with PMAR Device

Tip - Do you access the same PNA dialog often?

Your Favorites are always two keystrokes away.

What's New in PNA Code Version A.09.31

- Support for N1913A and N1914A Power Meters

What's New in PNA Code Version A.09.30
Application Enhancements

- Gain Compression on Converters (GCX)
- Support for Dual-Stage Converters in all Apps

Standard Measurements - available on all Models / Options

- Enhanced S-parameter Power Cal
- Marker Display enhancements
- Perform Source Power Cal at multiple power levels
- IF Gain Setting
- Receiver Overload/Compression Warning and Power OFF Preferences
- Confirm changes on Meas Class dialog
- DSP Version 5

What's New in PNA Code Version A.09.22

- Use any PNA-X Ports with Noise Figure Opt 028.

What's New in PNA Code Version A.09.20

Application Enhancements

- Integrated Pulse Measurements (Opt 008)
- Noise Figure using Standard PNA Receiver (Opt 028)
- Noise Figure on N5244A/45A (Opt H29)
- Edge and Level Trigger in Pulse
- Exclude SC12 Sweep for SMC (Opt 082/083)
- Include Phase with SMC (PNA-X with Opt 083)
• Fixturing in Apps
• Max Output Power for GCA (Opt 086)

**Standard Measurements - available on all Models / Options**

• Mechanical Device conflicts cause Channel Block (NOT Channel Hold)
• PSAT Marker and Power Normal Operating Point Marker
• Group Delay Aperture Setting
• Active Background Display Color
• Solid or Dotted Grid Lines
• Point Sweep on PNA "C" Models
• Fixture Power Compensation
• Sweep Delay
• Uncertainty equations using RSS Computations
• Preset Power Preference Setting
• Use Last Receiver Leveling Correction for SPC

**Data Save Enhancements**

• Recall .SNP files to view as trace
• "Save Data As" Dialog
• Save Balanced Data as SNP files

• Characterize Adaptor Macro Rev. A.02.10

  • Reverse S2P
  • Load the PNA Power Loss Table from an existing S2P file

---

**What's New in PNA Code Version A.09.10**

• Noise Figure on Converters (NFX)
• AgileUpdate for Customer Releases
What’s New in PNA Code Version A.09.00

- External Device Configuration
- Power Meter as Receiver (PMAR)
- Power Offsets and Limits
- Display and Print Colors
- Scale Coupling
- Mechanical Device Settings
- Device side USB
- Increased Number of Channels to 200
- ECal User Chars Saved to PNA Disk Memory

Application Enhancements

- GCA Compression Analysis
- GCA Compression from Saturation
- Receiver Leveling on IMD, FCA, and GCA
- Embedded LO on SMC and IMDx
- Limited Port Mapping on IMD
- Point Averaging on FCA and IMD
- SMC Measurements with mmWave Modules
- mmWave Measurements with no Test Set

What’s New in PNA Code Version A.08.60
- New 40 GHz and 50 GHz PNA-X Models

### What's New in PNA Code Version A.08.55

- IMDx (Swept and Spectrum) for Converters
- ADC Measurements in a Gain Compression channel.
- New 13.5 GHz PNA-X Model

### What's New in PNA Code Version A.08.50

- Fast Antenna Features for the PNA-X
- Receiver Leveling
- Phase Sweep in iTMSA
- Up to 25 User Macros
- Gain Compression Marker
- Port Extensions enhancements
- Electrical Delay enhancements
- Extra Security enhancement
- Save *.CSV and *.MDF File Types
- Noise Figure App enhancements:
  - Scalar Noise Figure measurement
  - Incident Noise Power parameters
- MM Module enhancements:
  - Power Level Control
  - Supports iTMSA
- Max point count to 32,001
- Faster Power Sweeps
What's New in PNA Code Version A.08.35

- New N5264A model

What's New in PNA Code Version A.08.33

- IMD Application (Opt 087)
- Fast Sweep Mode
- New Equation Editor functions
- 20001 Segments in Power Loss Table
- Data Format Units
- Marker=>CW Freq Function
- Up to 12 User Characterizations
- Characterize Adaptor Macro 2.0

What's New in PNA Code Version A.08.20

- iTMSA
- User Preferences dialog
- Uncoupled Power Sweep
- Equation Editor Import Functions
- Wider Traces Preference
- 24 Traces per Window
- LXI Compliance
- GCA Enhancements
- FCA - selectable ports
- Support for N5261A and N5262A MM test sets
What's New in PNA Code Version A.08.00

- Noise Figure Application (Opt 029)
- Gain Compression Application (Opt 086)
- 'Sweep' Trigger Mode
- Custom Cal Window settings (remote only)
- New Equation Editor Functions
- Minimum Number of Points = 1
KEYSIGHT SOFTWARE END-USER LICENSE AGREEMENT

ATTENTION: THIS SOFTWARE IS SUBJECT TO THE END-USER LICENSE AGREEMENT ("EULA") SET FORTH BELOW.

TO INSTALL OR USE THE SOFTWARE, YOU MUST FIRST AGREE TO THE EULA BELOW. IF THE EULA IS PRESENTED TO YOU ELECTRONICALLY AND IF YOU HAVE READ, UNDERSTAND AND AGREE TO BE BOUND BY THE TERMS OF THE EULA, CLICK "AGREE". IF THE EULA IS PRESENTED TO YOU IN A HARD COPY FORMAT, BY POWERING ON AND USING THE INSTRUMENT OR MACHINE, YOU AGREE THAT YOU HAVE READ, UNDERSTAND AND AGREE TO BE BOUND BY THE TERMS OF THE EULA.

1. Translations. Translations of this EULA are found at: www.keysight.com/find/sweula.

2. Software. "Software" means a single copy of one or more computer programs, whether stand-alone or bundled with other products, and related documentation, including any online or electronic documentation, data and license files.

3. License Grant. Keysight Technologies, Inc. ("Keysight") grants you a limited, non-exclusive license to use, in accordance with one of the license types listed below, the Software, for the Term (as defined below), subject to the terms and conditions herein:

3.1 Node Locked ("fixed") license. If you have obtained a Node Locked license, you may install one copy of the Software on one instrument or machine and use the Software only on that instrument or machine for your internal business use.

3.2 Transportable license. If you have obtained a Transportable license, you may use one copy of the Software on any single instrument or machine at one time for your internal business use. You may move such Software to a different instrument or machine for your internal business use provided only one copy is in use at any one time.

3.3 Floating ("concurrent use") license. If you have obtained a Floating license, you may install one or more copies of the Software on any instrument or machine within your internal computer network for your internal business use provided the total number of users who are accessing and/or using any of the Software at the same time does not exceed the maximum number of licensed users.

You may find the type of license you obtained, the Term of your license, and the licensed number of users (if applicable) in documentation associated with the Software. "Term" means either a set amount of time (an expiring license) or a Perpetual license. "Perpetual" means the lifetime of the instrument or machine. In the absence of documentation specifying the applicable license, you have a fixed license with a Perpetual Term.
4. License Restrictions.

4.1 No Copies. You may not make copies or adaptations of the Software except for backup and archival purposes or when copying or adaptation is an essential step in the licensed use of the Software including correction of errors. You must reproduce all copyright and other legal notices in the original Software on all permitted copies or adaptations. You may not copy the Software onto any public or distributed network.

4.2 No Reverse Engineering. You may not decompile, reverse engineer, disassemble, attempt to derive the source code of, decrypt, modify, or create derivative works of the Software (except to the extent any foregoing restriction is prohibited by applicable mandatory law or by licensing terms governing the use of open source components that may be included with the Software).

4.3 No Circumvention. The Software may include technological measures, whether in the Software or in bundled hardware or both, that are designed to prevent or detect unlicensed use of the Software. Circumvention of these technological measures is prohibited, except as expressly permitted by applicable law. Any attempt to circumvent technological measures may render the Software or certain features unusable or unstable, and may prevent you from updating or upgrading the Software.

4.4 Limited Use. Notwithstanding anything to the contrary herein, you may not use Software to make or distribute your own or a third party’s application, a principal purpose of which, as reasonably determined by Keysight, is to perform the same or similar functions as the Software or which replaces any component of the Software.

5. Third Party Software.

5.1 General. The Software may contain third party software subject to third party notices and/or additional terms and conditions. Such required third party software notices and/or additional terms and conditions can be found in the documentation associated with the Software. You have all rights necessary to use the Software as permitted in Sections 3 and 4. To the extent your use exceeds the grants and restrictions in Sections 3 and 4, third party license terms take precedence and will apply.

5.2 Separation of Components. Except as required by included open source software licenses, the Software is licensed as a single product and its component parts may not be separated for any other use except to the extent expressly otherwise licensed.

5.3 Additional Terms for Microsoft Windows Embedded Software. Microsoft Windows for Embedded Systems may be included with the instrument or machine. If so, your grant under this license prohibits accessing or using any Desktop Functions other than through, in support of and operating as part of the software and/or functions that provide the primary functionality of the instrument or machine. “Desktop Functions” means consumer or business tasks or processes, including word processing, spreadsheets, database, scheduling and personal finance.

6. Upgrades. This EULA does not entitle you to receive upgrades, updates or technical support. Upgrades, updates and technical support services may be purchased separately. The terms of this EULA govern any software updates or upgrades provided by Keysight unless replaced and/or supplement Software is accompanied by a separate license agreement in which case the terms of that license agreement will govern. Any comments, suggestions, improvements or other communications from you to Keysight regarding the Software (“Feedback”) may be used by Keysight without compensation or attribution.

7. Ownership. The Software and all copies thereof are licensed and not sold to you. The Software and all copies thereof are owned and copyrighted by Keysight or its third party suppliers and protected by copyright laws and other intellectual property laws and treaties. Keysight and its third party suppliers retain all right, title and interest in the Software. Keysight and its third party suppliers may protect their respective rights in the Software in the event of any violation of this EULA.

8. High Risk Activities. The Software is not specifically written, designed, manufactured or intended for use in the planning, construction, maintenance or direct operation of a nuclear facility, nor for use in on line control or fail safe operation of aircraft navigation, control or communication systems, weapon systems or direct life support systems.
9. Transfer. You may not sell or otherwise transfer the Software except as expressly specified below. Licensed users are Licensee’s employees, authorized agents, representatives, and subcontractors acting on behalf of Licensee for Licensee’s internal business use. You may not rent, lease, lend, allow for commercial time sharing, or otherwise permit access to the Software to any unlicensed third party or entity. You may not transfer the license granted to you here unless you obtain Keysight’s prior written authorization, deliver all copies of the Software to the transferee along with this EULA, and pay any applicable fees. For all transfers, the transferee must accept this EULA as a condition to any transfer and your license to use the Software will terminate upon transfer. Entitlement to receive technical support services for the Software may be transferred provided that you obtain Keysight’s prior written consent and pay any applicable fees. This section regarding transfer applies only to the extent permissible under applicable mandatory laws.

10. Term and Termination. This EULA shall continue for the Term unless terminated by Keysight as provided herein. Keysight may terminate this license upon notice for breach of this EULA. Upon expiration or termination, you must immediately destroy all copies of the Software.

11. Export Requirements. If you export, re-export or import Software, technology or technical data licensed hereunder, you assume responsibility for complying with applicable laws and regulations and for obtaining required export and import authorizations. Keysight may terminate this license immediately if you are in violation of any applicable laws or regulations.

12. Audit and Security Mechanisms. You agree that Keysight may restrict the number of copies you are using by security servers, security keys/modules, a hardware lock device, license administration software, a license authorization key to control access to the Software or other security mechanism and you consent to such use. You may not take any steps to avoid or defeat the purpose of any such measures. Use of the Software without the lock device or authorization key provided by Keysight is prohibited. Keysight may take all legal steps to eliminate unlicensed use and piracy of their Software. In this context, the Software may include a security mechanism to detect installation or use of unlicensed copies of the Software, and collect and transmit data about suspected unlicensed copies. Data collected does not include any customer data created with the Software. By using the Software, you consent to such detection and collection of data, as well as its transmission and use if suspected unlicensed copies are detected. Upon reasonable notice and reasonable suspicion of unlicensed use, Keysight may conduct during normal business hours (with the auditor’s costs being at Keysight’s expense) an audit of your compliance with this EULA. If an audit reveals underpayments then you will pay to Keysight such underpayments. If underpayments discovered exceed five (5) percent of the contract price, you will reimburse Keysight for the auditor costs.

13. U.S. Government Rights. The Software is “commercial computer software,” as defined by Federal Acquisition Regulation (“FAR”) 2.101. Pursuant to FAR 12.212 and 27.405-3 and Department of Defense FAR Supplement (“DFARS”) 227.7202, the U.S. government acquires commercial computer software under the same terms by which the software is customarily provided to the public. Accordingly, Keysight provides the Software to U.S. government customers under its standard commercial license, which is embodied in this EULA. The license set forth in this EULA represents the exclusive authority by which the U.S. government may use, modify, distribute, or disclose the Software. This EULA and the license set forth herein, does not require or permit, among other things, that Keysight: (1) Furnish technical information related to commercial computer software or commercial computer software documentation that is not customarily provided to the public; or (2) Relinquish to, or otherwise provide, the government rights in excess of these rights customarily provided to the public to use, modify, reproduce, release, perform, display, or disclose commercial computer software or commercial computer software documentation. If you are a U.S. government customer, you acknowledge that you have reviewed the Software and the terms of this EULA and agree that the license provided for herein is consistent with Federal law and otherwise satisfies the U.S. government needs. In addition, if you are a U.S. government customer you agree that this EULA reflects the entirety of the terms of Keysight’s customary commercial license applicable to U.S. government customers. No additional government requirements beyond those set forth in this EULA shall apply, except to the extent that those terms, rights, or licenses are explicitly required from all providers of commercial computer software pursuant to the FAR and the DFARS and are set forth specifically in writing elsewhere in this EULA. Keysight shall be under no obligation to update, revise or otherwise modify the Software.

14. WARRANTY. TO THE EXTENT ALLOWED BY APPLICABLE MANDATORY LAW, AND EXCEPT TO THE EXTENT KEYSIGHT HAS PROVIDED A SPECIFIC WRITTEN WARRANTY APPLICABLE TO THIS SOFTWARE, THIS SOFTWARE IS PROVIDED TO YOU "AS IS" WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, WHETHER ORAL OR WRITTEN, EXPRESS OR IMPLIED. KEYSIGHT, ON BEHALF OF ITSELF ITS SUBSIDIARIES, AFFILIATES AND
SUPPLIERS SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY,
SATISFACTORY QUALITY, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE. SHOULD THE
SOFTWARE PROVE DEFECTIVE, YOU ASSUME THE ENTIRE RISK AND COST RESULTING FROM OR RELATING TO
THE DEFECT. SOME JURISDICTIONS DO NOT ALLOW EXCLUSIONS OF IMPLIED WARRANTIES OR CONDITIONS,
SO THE ABOVE EXCLUSION MAY NOT APPLY TO YOU. YOU MAY HAVE OTHER RIGHTS THAT VARY ACCORDING
TO APPLICABLE MANDATORY LAW.

15. LIMITATION OF LIABILITY. TO THE EXTENT ALLOWED BY APPLICABLE MANDATORY LAW, IN NO EVENT WILL
KEYSIGHT OR ITS SUBSIDIARIES, AFFILIATES OR SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL,
CONSEQUENTIAL OR OTHER DAMAGES (INCLUDING DOWNTIME COSTS, LOSS OF DATA, RESTORATION COSTS
OR LOST PROFITS) REGARDLESS OF WHETHER SUCH CLAIMS ARE BASED ON CONTRACT, TORT, WARRANTY
OR ANY OTHER LEGAL THEORY, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. YOUR USE OF THE
SOFTWARE IS ENTIRELY AT YOUR OWN RISK. NOTWITHSTANDING THE FOREGOING, IN THE EVENT THE
SOFTWARE IS PROVIDED TO YOU AT NO CHARGE, KEYSIGHT OR ITS SUBSIDIARIES, AFFILIATES OR SUPPLIERS
SHALL HAVE NO LIABILITY FOR DIRECT DAMAGES. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR
LIMITATION OF LIABILITY FOR DAMAGES, SO THE ABOVE LIMITATION MAY NOT APPLY TO YOU.

16. Applicable Law. Disputes arising in connection with this EULA will be governed by the laws of the United States and of
the State of California, without regard to choice of law provisions. The United Nations Convention for Contracts for the
International Sale of Goods will not apply to this EULA.

17. Unenforceability. To the extent that any provision of this EULA is determined to be illegal or unenforceable, the
remainder of this EULA will remain in full force and effect.

18. Entire Agreement. Certain program, data and license files in the Software may be subject to supplemental license
terms found in the documentation associated with the Software or directly in the files to which the supplemental terms apply.
This EULA constitutes the entire agreement between you and Keysight with respect to the Software, and supersedes any
previous communications, representations or agreements between the parties, whether oral or written, except if you have a
separate written, valid agreement that is executed by both parties and the terms of such agreement conflict with the terms
contained herein, in which case the terms of such agreement apply. This EULA may not be changed except by an
amendment signed by an authorized representative of each party.
VNA User Accounts and Passwords

**Important:** When the VNA power is switched on, it AUTOMATICALLY logs into Windows using the default user name and password. You do NOT need to log on. This gives anyone full access to the analyzer. The following steps can be taken to increase security of your VNA.

- Require users to logon when the VNA computer is turned ON - Learn how.
- Setup individual accounts on the VNA with varying level of access - Learn how.

See Also: User-Specific VNA Settings

Please read about Anti-virus protection for your VNA

Existing User Accounts

The following user accounts already exist on new VNAs.

**Important:** For the highest security, it is recommended to change the passwords on your device from these defaults.

<table>
<thead>
<tr>
<th>User Name</th>
<th>Password</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instrument</td>
<td>measure4u</td>
<td>Administrator</td>
<td>Auto Log On is activated by default.</td>
</tr>
<tr>
<td>Administrator</td>
<td>Keysight4u!</td>
<td>Built-in Administrator</td>
<td>For user maintenance purpose.</td>
</tr>
</tbody>
</table>

**Note:** The user name is not case sensitive. The password is case sensitive.

**Note:** The VNA local policies are set so that, if logon is required, you must retype the user name (and password) every time. Do not change the local policies on the VNA.

Add or Change User Accounts, Passwords, and require Logon

If the analyzer is in a secure environment, you can setup VNA users by name and grant various levels of access.

You can designate a person as the administrator and then configure the VNA to allow others to use it with reduced permissions. That is, other people can be signed on to use the analyzer but they will not have the ability to perform all of the administrative functions that you can as the administrator.
The following are examples of some of the functions that can be performed with these account types:

- **Administrator** - Can download and install firmware. The administrator can modify system-wide settings in the operating system.

- **Standard User** - Can fully use the VNA to make measurements and save state and data files. Standard Users can NOT install firmware.

### How to add a user account and require logon

<table>
<thead>
<tr>
<th>Windows 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Click <strong>Start</strong>, then <strong>Control Panel</strong></td>
</tr>
<tr>
<td>Click <strong>User Accounts</strong></td>
</tr>
<tr>
<td>Click <strong>Manage another account.</strong></td>
</tr>
</tbody>
</table>

**CAUTION:** Although allowed by Windows, do NOT allow an Administrator account without a password. Internet viruses look for, and exploit, this condition.

Click **Create a new account.**

Enter an account name.

Select an account type.

Click **Create Account.**

**Optional:** Click **Create a password** to require the user to enter a password when logging on.
Windows 10

To add a new account:

- Click Start > Settings > Accounts > Other people > Add someone else to this PC.
- Enter the requested information then select Next.

To change an existing account:

- Click on the user account.
- Click Change account type.
- Under Account type, select Standard User or Administrator.
- Click OK.

User-Specific VNA Settings

Almost all persistent settings in the VNA are global (apply to all users).

The following exceptions reset to their defaults when a new user account is setup:

- Global PassFail Display State
- Recently used files list
- Global Power Limits
- Equation Editor – most recently used import dlls
- Preferences:
  - Power Sweep Retrace Mode
  - Is Power On During Retrace
VNA Computer Properties

The VNA uses a personal computer and a Windows operating system. The following are common tasks that you may need to perform on the VNA computer.

- View or change Full Computer Name
- Check IP Address
- Check the amount of RAM
- Check CPU Speed
- Check SSD Capacity
- Set Time and Date
- Internal and External Speakers (N52xxB only)
- Windows 10 Updates

Other Administrative Task Topics

View or change Full Computer Name

Your VNA has a unique computer name that identifies it on a network. To view or change the computer name, you must first minimize the VNA application.

<table>
<thead>
<tr>
<th>Windows 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>On the desktop, click <strong>Start</strong>.</td>
</tr>
<tr>
<td>Right-click on <strong>Computer</strong> then select <strong>Properties</strong>.</td>
</tr>
<tr>
<td>Scroll down to <strong>Computer name, domain, and workgroup settings</strong>.</td>
</tr>
<tr>
<td>To make changes, click <strong>Change Settings</strong>.</td>
</tr>
</tbody>
</table>
Windows 10
Click **Start > Settings** then go to **System > About**.
Click **Rename PC** button.
Type the new name for your computer.
Click **Next**.
A dialog is displayed prompting you to restart your computer now or later. If you want to change to the new name now, click the **Restart now** button to restart your computer. If you don't want to restart your computer now, click the **Restart later button**. Your computer's name will be updated the next time you restart it.

**Note:** To add your computer to a domain, or to set up the networking configuration, contact your company's I.T. department. This setup is custom for each company.

To restore the VNA application, click **Network Analyzer** in the task bar at the bottom of the screen.

**Check IP Address**
If your VNA is connected to a LAN, you can view the IP address and other networking information.

Windows 7
**Minimize the VNA** application
On the desktop, click **Start**.
In the Search field, type **cmd**, then click **OK**.
At a DOS prompt, type **ipconfig /all**.

Windows 10
Click **Start > Settings**.
Click on the **Network & Internet** icon.
Click the **View your network properties** link.
Check the amount of RAM

Random Access Memory (RAM) is the amount of working memory in your computer.

The amount of RAM in your VNA may limit your ability to upgrade firmware. See
http://na.support.keysight.com/pna/firmware/PNA_support_matrix.doc

To view the amount of VNA RAM, you must first minimize the VNA application.

<table>
<thead>
<tr>
<th>Windows 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>On the desktop, click <strong>Start</strong>.</td>
</tr>
<tr>
<td>Right-click on <strong>Computer</strong> then select <strong>Properties</strong>.</td>
</tr>
<tr>
<td>The amount of RAM appears under <strong>System</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Windows 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Click <strong>Start &gt; Settings</strong>.</td>
</tr>
<tr>
<td>Click <strong>System</strong> then select <strong>About</strong></td>
</tr>
</tbody>
</table>

To restore the VNA application, click **Network Analyzer** in the task bar at the bottom of the screen.

Check CPU Speed

The speed of the VNA processor (CPU) is a factor in determining how quickly the VNA processes data. Also, the CPU speed in your VNA may limit your ability to upgrade firmware. See

You can see which CPU is in your VNA by comparing your VNA rear-panel with the images at

Or, on the VNA click **Help**, then **About Network Analyzer**. Learn more.

Or, you can do the following to check your VNA CPU speed:

1. Minimize the VNA application.
2. Then do the following:
To restore the VNA application, click Network Analyzer in the task bar at the bottom of the screen.

Check Solid State Drive (SSD) Capacity

To view the capacity of drive space, you must first minimize the VNA application.

Then do the following:

**Windows 7**

On the desktop, click Start. —> Computer.

The amount of free drive space and capacity for each of the partitions appears.

**Windows 10**

Click Start > Settings.

Click System then select About

Learn how to determine your HDD part number (Internet connection required).

Set Time and Date

To set the time and date on your VNA, you must first minimize the VNA application.
### Windows 7

Move the cursor to the lower corner of the screen.

When the taskbar appears, click on the displayed time.

Select **Change data and time settings**. This opens the **Date and Time** dialog box.

Change the date, time, and time zone as appropriate.

### Windows 10

Click **Start** > **Settings**.

Click **Time & language**.

Click **Date & time**.

Ensure that **Set time automatically** is switched **Off**.

Click on the **Change** button to change the time, date, and time zone as appropriate.

To restore the VNA application, click **Network Analyzer** in the task bar at the bottom of the screen.

### Internal Speaker (N52xxB only)

There is an internal sound card and speaker in the VNA. However, there is no audio output jack.

There may be times when you might want to control the speaker volume or turn the speaker OFF, such as when the VNA is generating errors. [Learn more about errors.](#)

To control the VNA speaker volume:

1. On the VNA press **System** -> **System Setup** -> **Sound**, then click on the keypad icon.
2. Enter a value between 0 (speaker OFF) and 100 (highest volume).

Windows 10 Updates

Windows 10 updates cannot be turned off. There are two ways to control updates:

- Notify
- Automatically download and install

The VNA is set to Notify before downloading. When Windows finds updates that apply to this computer, users are notified that updates are ready to be downloaded. After going to Windows Update, users can download and install any available updates.
Run Error Check and Disk Defragmenter

When the VNA is shutdown unexpectedly or power is removed without first shutting down, large amounts of Hard Disk Drive space is rendered unusable. If shutdown in this manner enough times, the VNA could become unstable and no longer work.

This Hard Disk Drive space can be recovered by first running Windows Error-checking to find and correct errors on the disk, and then the Disk Defragmenter to recover Hard Disk Drive space. These programs should be run routinely, about every 1 to 4 weeks, depending on how often the VNA is unexpectedly shutdown.

To learn more about Disk Defragmenter, see the Windows Help file.

Follow this procedure to run these programs:

### Windows 7

1. On the desktop, click **Start -> Computer**
2. In the upper pane, click **System properties**
3. In the left pane, click **Performance Information and Tools**.
4. In the left pane, click **Advanced Tools**.
5. Click **Open Disk Defragmenter**.

### Windows 10

1. On the desktop, click **Start -> Windows Administrative Tools**.
2. Click **Defragment and Optimize Drives**.

**Error-checking**

- Click **Analyze disk**.

- Once Windows is finished analyzing the disk, you can check the percentage of fragmentation on the disk in the Last Run column. If the number is above 10%, you should defragment the disk.

**Defragmentation**

- Click **Defragment Disk**.
- Disk Defragmenter might take from several minutes to a few hours to finish, depending on the size and degree of fragmentation of your hard disk.
The leading cause of VNA failures is problems with the VNA Solid State Drive (SSD). These problems are usually preventable (see Preventing VNA SSD Problems), and in many cases, recoverable. The following could save you weeks of downtime and the cost of replacing your VNA SSD.

This document is now on the Keysight VNA Support Website:

If your VNA does experience a Solid State Drive problem, you will not be able to access this Help file, but you may be able to access the Internet from another computer.
Microsoft Windows Considerations

In this topic:

- Microsoft Windows on the VNA
- Using USB
- LAN Connections
- Mouse Configuration
- Windows Theme
- Printing

See Also

Windows File Locations
Your Programs on Windows
Microsoft EULA

Microsoft Windows on the VNA

The VNA is shipped from the factory with Windows® Embedded Standard 7 or Windows® 10 operating system. This OS supports both 32-bit and 64-bit applications.

**VERY IMPORTANT  Protect your hard drive!**

The leading cause of VNA failures is problems with the VNA Solid State Drive (SSD). These problems are usually preventable, and in many cases, recoverable. Learn more about protecting your VNA.

Using USB

The VNA has USB ports on the front panel and on the rear panel. The main advantages of USB are “hot” connects and disconnects and fast data transfer speeds. Electronic Calibration modules are also available with USB connections.

The first time you plug a device into a USB port there is some wait time. Windows reports it is identifying the hardware, then searching for the correct driver, then installing the driver (if it was found).
Connecting that same device back into that same port later is quick and easy, but if you move the device to a different USB port, you will have to wait through the hardware ID and driver search again.

Learn about USB limitations.

LAN Connections

Windows supports DHCP and fixed IP addressing. Also, “Hot” connect and disconnect of the LAN cable, as well as a visual indicator of LAN status in system tray area, makes LAN connections more intuitive. In addition, the Hardware Wizard helps users with system hardware configuration.

Mouse Configuration

By default, Windows does not allow you to select to make the button on the right the one you use for primary functions such as selecting and dragging. To change any mouse properties, click **Start > Control Panel > Mouse**. In the Mouse Properties dialog box, select any settings that you would like to change, click **Apply** and then click **OK**.

Windows Themes

The VNA application is designed for, and best viewed in, **Keysight Technologies** theme. To change the theme:

1. Minimize the VNA application.
2. Right-click on the Desktop, then click **Personalize**
3. Use the scrollbar, then select a Theme.

Printing

Adding a printer should be done outside of the VNA application. Learn more.
Windows File Locations

With Windows 7 or Windows 10, data file locations have been changed from previous Windows Operating Systems.

**State files and most data file storage locations:**

- Windows 7 or Windows 10: D: \n
**Firmware executable file locations:**

- Windows 7 or Windows 10: c:\program files\Keysight\network analyzer

**Support file locations**

- Windows 7 or Windows 10: c:\programdata\Keysight\network analyzer

These file locations can be queried remotely.

SCPI Command: SYSTem:CONFigure:DIRectory?

COM Command: DirectoryPath Property

See Also

Your Programs on Windows
Simulator (S94050B, S94051B)

All the features of the standard class are enabled by the simulator license. Without the license, the firmware enters Viewer Mode in which many of the features are disabled.

To install the VNA simulator, user has to

1. Download and install the IO library if it is not installed. https://www.keysight.com/find/iolibs

2. Download Firmware from Keysight web (Drivers, Firmwares & Software Tab on http://www.keysight.com/support/m9800a ). The firmware supports the simulator capability for all models including PNA and ENA.

3. Install the firmware to the PC. Follow the instructions for the installation.

4. Execute the Keysight License Manager from Start Menu in Windows to install the license. Click “What if I have a license file to install”, then follow the instruction to install the license file. Learn more on option enable.

5. For PXIe/USB VNA, you can open the Simulator setup dialog thru VNA SFP Launcher.

In this topic:

- VNA Simulator Setup Dialog (for PXIe/USB VNA only)
- VNA Simulator License
- Simulator Mode
- Dummy DUT

VNA Simulator Setup Dialog (for PXIe/USB VNA only)
Model Type  Select the series for simulator mode.

Model Number  Pulldown menu to select the model for simulator mode. It will change depending on the selected Model Type.

Number of Ports  To change the number of ports with the Up/Down button or edit that value using keyboard. Acceptable number as the number of ports is even number from 2 to 66. The maximum number of ports will change depending on the selected model.

Run  Start the firmware as Simulator mode with the selected setup. The selected Model Type, Model Number and Number of ports will be saved as new default values and these new default values will be used when the dialog will open next time.

Cancel  Does not apply any changes and close the simulator setup dialog.

Help  Open Help.

VNA Simulator License

One of the licenses below is required to enable VNA Simulator.

- S94050B – Standard Simulator
- S94051B – Advanced Simulator

When no simulator license exists, VNA Simulator starts with Data Viewer Mode.

Simulator Mode

There are three simulator modes as below:

1. Data Viewer

Data Viewer is a free version of VNA Simulator and supports STD Channel with restrictions. Advanced functions including applications and remote-control capability are disabled.

2. Standard Simulator

All the features of the standard class are enabled by the simulator base license. The option for the standard simulator is S94050B.

Standard Simulator supports STD Channel and remote-control capability. Applications are not supported.
3. Advanced Simulator

Option S94051B is needed to enable advanced simulator.

Advanced Simulator supports Standard Simulator features + Applications (SW options) listed below.

<table>
<thead>
<tr>
<th>VNA Family</th>
<th>Product</th>
<th>Supported Software Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNA</td>
<td>N522xB</td>
<td>S93xxx: 010,011,015,025,029,082,083,084,086,087,088,089,090,094,118,322,460</td>
</tr>
<tr>
<td></td>
<td>N523xB</td>
<td>S93xxx: 010,011,015,082,090,322</td>
</tr>
<tr>
<td></td>
<td>N524xB</td>
<td>S93xxx: 010,011,015,025,029,082,083,084,086,087,088,089,090,094,110,111,114</td>
</tr>
<tr>
<td>ENA</td>
<td>E5080A</td>
<td>S96xxx: 010,082,086</td>
</tr>
<tr>
<td></td>
<td>E5080B</td>
<td>S96xxx: 010,011,015,025,029,082,083,084,086,088,090,460</td>
</tr>
<tr>
<td>PXI</td>
<td>M980xA</td>
<td>S95xxx: 010,011,015,025,029,070,082,083,084,086,088,090,460</td>
</tr>
<tr>
<td></td>
<td>M937xA</td>
<td>M9372A-xxx: 009,010</td>
</tr>
<tr>
<td>Streamline</td>
<td>P500x/2xA</td>
<td>S97xxx: 010,011,015,025,029,082,083,084,086,088,090,460</td>
</tr>
<tr>
<td>(USB VNA)</td>
<td>P937xA</td>
<td>S97xxx: 010,082</td>
</tr>
</tbody>
</table>

The following options are not supported.

- S9x007B Automatic Fixture Removal for all models
- S96041B Impedance Analysis
- S95552B Multiport Calibration Assistant
- S95553B Multiport calibrated measurements with switch instruments

Hardware options:

- Software options include required hardware options (For M980xA/P50xxA/E5080B, option 021 is supported for S9x025A/B, option 090 is also supported for S9x090.)
- 2nd source (option 022) is enabled for N524xB, 4-port N522xB, 4-port E5080B and P502xA only
- N source is enabled for M980xA but supported N*2-port configurations only
- LFE option is enabled for N5222/27B (option 205) and N5242/45/47B (option 425)
- Analog Input (option 175) is enabled for E5080B
- Multi-port (unit) configuration in USB VNA is not supported
ECal and external devices such as source, power sensor are not supported. PXI internal devices such as the switch and M9341A/B Digital IO supports GUI only.

**Dummy DUT**

Dummy DUT capability provides measurement data manipulation for VNA simulator using sNp format files.

VNA Simulator automatically loads a “default” dummy DUT put under VNA installed folder (C:\Program Files\Keysight\Network Analyzer\simulations\DummyDUT.s6p)

Users can set their own default dummy DUT under user VNA document folder (C:\Users\Public\Documents\Network Analyzer\UserDummyDUT.s*p)

When there are multiple “UserDummyDUT.s*p” files, VNA simulator loads the first found one.

Dummy DUT Tool provides GUI for dummy DUT handling. The tool is Installed under VNA Macro softkey.

### How to create Dummy DUT snp file

Use one of the following methods to make a dummy DUT snp file

#### Using Hardkey /SoftTab /Softkey

1. Press Mac**ro** > **Mac**ro1 > Dummy D**UT** tool

#### Using a mouse

1. Click **Utility** > **Macro** > **Dummy DUT tool**

**Current DUT**  Show sNp file currently used as dummy DUT

**Load...**  Load user specific sNp file as current dummy DUT
**Clear**  Clear current dummy DUT

**Noise**  Set noise state ON/OFF

**Save Current As User Default**  Save current dummy DUT as user default
Connector Care

Proper connector care is critical for accurate and repeatable measurements. The following information will help you preserve the precision and extend the life of your connectors - saving both time and money.

- Connector Care Quick Reference Guide
- Connector Cleaning Supplies
- Safety Reminders
- About Connectors
- Gaging Fundamentals
- Connector Care Procedures

See Also

mmWave Connector Care at http://na.support.keysight.com/pna/connectorcare/Connector_Care.htm

Important Note

All of the copper N5247A front-panel loops are EXTREMELY FRAGILE.

A maximum 8 inch/lbs of torque should be applied to these connectors.

If more torque is applied, damage to the copper semi-rigid cables WILL occur, causing measurement inaccuracy.

Preventing Test Port Connector Damage
## Handling and Storing Connectors

<table>
<thead>
<tr>
<th>Do</th>
<th>Do Not</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keep connectors clean</td>
<td>Touch mating-plane surfaces</td>
</tr>
<tr>
<td>Protect connectors with plastic end caps</td>
<td>Set connectors contact-end down</td>
</tr>
<tr>
<td>Keep connector temperature same as analyzer</td>
<td>Store connectors loose in box or drawer</td>
</tr>
</tbody>
</table>

### Visual Inspection

<table>
<thead>
<tr>
<th>Do</th>
<th>Do Not</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inspect connectors with magnifying glass.</td>
<td>Use a connector with a bent or broken center conductor</td>
</tr>
<tr>
<td>Look for metal debris, deep scratches or dents</td>
<td>Use a connector with deformed threads</td>
</tr>
</tbody>
</table>

### Cleaning Connectors

<table>
<thead>
<tr>
<th>Do</th>
<th>Do Not</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clean surfaces first with clean, dry compressed air</td>
<td>Use high pressure air (&gt;60 psi)</td>
</tr>
<tr>
<td>Use lint-free swab or brush</td>
<td>Use any abrasives</td>
</tr>
<tr>
<td>Use minimum amount of alcohol</td>
<td>Allow alcohol into connector support beads</td>
</tr>
<tr>
<td>Clean outer conductor mating surface and threads</td>
<td>Apply lateral force to center conductor</td>
</tr>
</tbody>
</table>

### Gaging Connectors

<table>
<thead>
<tr>
<th>Do</th>
<th>Do Not</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inspect and clean gage, gage primary and device tested</td>
<td>Use an out of specification connector</td>
</tr>
<tr>
<td>Use correct torque wrench</td>
<td>Hold connector gage by the dial</td>
</tr>
<tr>
<td>zero gage before use</td>
<td></td>
</tr>
<tr>
<td>Use multiple measurements and keep record of readings</td>
<td></td>
</tr>
</tbody>
</table>

### Making Connections

<table>
<thead>
<tr>
<th>Do</th>
<th>Do Not</th>
</tr>
</thead>
<tbody>
<tr>
<td>Align connectors first</td>
<td>Cross thread the connection</td>
</tr>
<tr>
<td>Rotate only the connector nut</td>
<td>Twist connector body to make connection</td>
</tr>
<tr>
<td>Use correct torque wrench</td>
<td>Mate different connector types</td>
</tr>
</tbody>
</table>
# Connector Care and Cleaning Supplies

<table>
<thead>
<tr>
<th>Description</th>
<th>Web Site</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wipes</td>
<td><a href="https://www.techspray.com/wipes">https://www.techspray.com/wipes</a></td>
</tr>
<tr>
<td>Aerosol Duster</td>
<td><a href="https://www.techspray.com/techspray-air-duster">https://www.techspray.com/techspray-air-duster</a></td>
</tr>
<tr>
<td>Isopropyl</td>
<td><a href="https://www.techspray.com/isopropyl-alcohol-ipa">https://www.techspray.com/isopropyl-alcohol-ipa</a></td>
</tr>
<tr>
<td>Nitrile Gloves and Finger Cots</td>
<td><a href="https://www.techni-tool.com">https://www.techni-tool.com</a></td>
</tr>
</tbody>
</table>

## Safety Reminders

### When cleaning connectors:

- Always use protective eyewear when using compressed air or nitrogen.
- Keep isopropyl alcohol away from heat, sparks and flame. Use with adequate ventilation. Avoid contact with eyes, skin and clothing.
- Avoid electrostatic discharge (ESD). Wear a grounded wrist strap (having a 1 MΩ series resistor) when cleaning device, cable or test port connectors.
- Cleaning connectors with alcohol shall only be done with the instruments power cord removed, and in a well-ventilated area. Allow all residual alcohol moisture to evaporate, and the fumes to dissipate prior to energizing the instrument.

## About Connectors

- Connector Service Life
- Connector Grades and Performance
- Adapters as Connector Savers
- Connector Mating Plane Surfaces

### Connector Service Life

Even though calibration standards, cables, and test set connectors are designed and manufactured to the highest standards, all connectors have a limited service life. This means that connectors can become defective due to wear during normal use. For best results, all connectors should be inspected and maintained to maximize their service life.

**Visual Inspection** should be performed each time a connection is made. Metal particles from connector threads often find their way onto the mating surface when a connection is made or disconnected. See Inspection procedure.
Cleaning the dirt and contamination from the connector mating plane surfaces and threads can extend the service life of the connector and improve the quality of your calibration and measurements. See Cleaning procedure.

Gaging connectors not only provides assurance of proper mechanical tolerances, and thus connector performance, but also indicate situations where the potential for damage to another connector may exist. See Gaging procedure.

Proper connector care and connection techniques yield:

- Longer Service Life
- Higher Performance
- Better Repeatability

Connector Grades and Performance

The three connector grades (levels of quality) for the popular connector families are listed below. Some specialized types may not have all three grades.

- **Production** grade connectors are the lowest grade and the least expensive. It is the connector grade most commonly used on the typical device under test (DUT). It has the lowest performance of all connectors due to its loose tolerances. This means that production grade connectors should always be carefully inspected before making a connection to the analyzer. Some production grade connectors are not intended to mate with metrology grade connectors.

- **Instrument** grade is the middle grade of connectors. It is mainly used in and with test instruments, most cables and adapters, and some calibration standards. It provides long life with good performance and tighter tolerances. It may have a dielectric supported interface and therefore may not exhibit the excellent match of a metrology grade connector.

- **Metrology** grade connectors have the highest performance and the highest cost of all connector grades. This grade is used on calibration standards, verification standards, and precision adapters. Because it is a high precision connector, it can withstand many connections and disconnections and, thus, has the longest life of all connector grades. This connector grade has the closest material and geometric specifications. Pin diameter and pin depth are very closely specified. Metrology grade uses an air dielectric interface and a slotless female contact which provide the highest performance and traceability.

**Note:** In general, Metrology grade connectors should not be mated with Production grade connectors.

Adapters as Connector Savers

Make sure to use a high quality (Instrument grade or better) adapter when adapting a different connector type to the analyzer test ports. It is a good idea to use an adapter even when the device under test is the same connector type as the analyzer test ports. In both cases, it will help extend service life, and protect the test ports from damage and costly repair.

The adapter must be fully inspected before connecting it to the analyzer test port and inspected and cleaned frequently thereafter. Because calibration standards are connected to the adapter, the adapter...
should be the highest quality to provide acceptable RF performance and minimize the effects of mismatch.

**Connector Mating Plane Surfaces**

An important concept in RF and microwave measurements is the reference plane. For a network analyzer, this is the surface that all measurements are referenced to. At calibration, the reference plane is defined as the plane where the mating plane surfaces of the measurement port and the calibration standards meet. Good connections (and calibrations) depend on perfectly flat contact between connectors at all points on the mating plane surfaces (as shown in the following graphic).

![Mating Plane Surfaces](image)

**Gaging Fundamentals**

Connector gages are important tools used to measure center conductor pin depth in connectors. Connector pin depth, measured in terms of recession or protrusion, is generally the distance between the mating plane and the end of the center conductor, or the shoulder of the center conductor for a stepped male pin.

**Typical Connector Gage**

<table>
<thead>
<tr>
<th>RECESSION</th>
<th>PROTRUSION</th>
</tr>
</thead>
</table>

Recession and Protrusion

Pin depth is negative (recession) if the center conductor is recessed below the outer conductor mating plane, usually referred to as the "reference plane". Pin depth is positive (protrusion) if the center conductor projects forward from the connector reference plane.

Pin Depth

1. Recession of female contact
2. Recession of male pin shoulder

Difference with Type-N Connectors
Type-N connectors have the mating plane of the center conductors offset from the connector reference plane. In this case the zero setting "gage primarys" generally offset the nominal distance between the center conductor mating plane and the connector reference plane.

### When to Gage Connectors

- Before using a connector or adapter the first time.
- When visual inspection or electrical performance suggests the connector interface may be out of range.
- After every 100 connections, depending on use.

### Connector Gage Accuracy

Connector gages (those included with calibration and verification kits), are capable of performing coarse measurements only. This is due to the repeatability uncertainties associated with the measurement. It is important to recognize that test port connectors and calibration standards have mechanical specifications that are extremely precise. Only special gaging processes and electrical testing (performed in a calibration lab) can accurately verify the mechanical characteristics of these devices. The pin depth specifications in the Keysight calibration kit manuals provide a compromise between the pin depth accuracy required, and the accuracy of the gages. The gages shipped with calibration and verification kits allow you to measure connector pin depth and avoid damage from out-of-specification connectors.

**Note:** Before gaging any connector, the mechanical specifications provided with that connector or device should be checked.

### To Gage Connectors

1. Wear a grounded wrist strap (having a 1 MΩ series resistor).
2. Select proper gage for device under test (DUT).
3. Inspect and clean gage, gage primary, and DUT.
4. Zero the connector gage.
   
   a. While holding gage by the barrel, carefully connect gage primary to gage. Finger-tighten connector nut only.
   
   b. Use proper torque wrench to make final connection. If needed, use additional wrench to prevent gage primary (body) from turning. Gently tap the barrel to settle the gage.
   
   c. The gage pointer should line up exactly with the zero mark on gage. If not, adjust "zero set" knob until gage pointer reads zero. On gages having a dial lock screw and a movable dial, loosen the dial lock.
screw and move the dial until the gage pointer reads zero. Gages should be zeroed before each set of measurements to make sure zero setting has not changed.

d. Remove gage primary.

5. Gage the device under test.

   a. While holding gage by the barrel, carefully connect DUT to gage. Finger-tighten connector nut only.

   b. Use proper torque wrench to make final connection and, if needed, use additional wrench to prevent DUT (body) from turning. Gently tap the barrel to settle the gage.

   c. Read gage indicator dial for recession or protrusion and compare reading with device specifications.

**Caution:** If the gage indicates excessive protrusion or recession, the connector should be marked for disposal or sent out for repair.

6. For maximum accuracy, measure the device a minimum of three times and take an average of the readings. After each measurement, rotate the gage a quarter-turn to reduce measurement variations.

7. If there is doubt about measurement accuracy, be sure the temperatures of the parts have stabilized. Then perform the cleaning, zeroing, and measuring procedure again.

**Connector Care Procedures**

- Inspecting Connectors
- Cleaning Connectors
- Making Connections
- Using a Torque Wrench
- Handling and Storing Connectors

**To Inspect Connectors**

Wear a grounded wrist strap (having a 1 MΩ series resistor).

Use a magnifying glass (≥10X) and inspect connector for the following:

- Badly worn plating or deep scratches
- Deformed threads
Metal particles on threads and mating plane surfaces

Bent, broken, or mis-aligned center conductors

Poor connector nut rotation

Caution: A damaged or out-of-specification device can destroy a good connector attached to it even on the first connection. Any connector with an obvious defect should be marked for disposal or sent out for repair.

To Clean Connectors

1. Wear a grounded wrist strap (having a 1 MΩ series resistor).

2. Use clean, low-pressure air to remove loose particles from mating plane surfaces and threads. Inspect connector thoroughly. If additional cleaning is required, continue with the following steps.


4. Clean contamination and debris from mating plane surfaces and threads. When cleaning interior surfaces, avoid exerting pressure on center conductor and keep swab fibers from getting trapped in the female center conductor.

5. Let alcohol evaporate–then use compressed air to blow surfaces clean.

6. Inspect connector. Make sure no particles or residue remains.

7. If defects are still visible after cleaning, the connector itself may be damaged and should not be used. Determine the cause of damage before making further connections.

To Make Connections
1. Wear a grounded wrist strap (having a 1 MΩ series resistor).

2. Inspect, clean, and gage connectors. All connectors must be undamaged, clean, and within mechanical specification.

3. Carefully align center axis of both devices. The center conductor pin—from the male connector—must slip concentrically into the contact finger of the female connector.

4. Carefully push the connectors straight together so they can engage smoothly. Rotate the connector nut (not the device itself) until finger-tight, being careful not to cross the threads.

5. Use a torque wrench to make final connection. Tighten until the "break" point of the torque wrench is reached. Do not push beyond initial break point. Use additional wrench, if needed, to prevent device body from turning.

To Separate a Connection

1. Support the devices to avoid any twisting, rocking or bending force on either connector.

2. Use an open-end wrench to prevent the device body from turning.
3. Use another open-end wrench to loosen the connector nut.

4. Complete the disconnection by hand, turning only the connector nut.

5. Pull the connectors straight apart.

**To Use a Torque Wrench**

1. Make sure torque wrench is set to the correct torque setting.

2. Position torque wrench and a second wrench (to hold device or cable) within 90° of each other before applying force. Make sure to support the devices to avoid putting stress on the connectors.

3. Hold torque wrench lightly at the end of handle—then apply force perpendicular to the torque wrench handle. Tighten until the "break" point of the torque wrench is reached. Do not push beyond initial break point.

**To Handle and Store Connectors**

- Install protective end caps when connectors are not in use.

- Never store connectors, airlines, or calibration standards loose in a box. This is a common cause of connector damage.

- Keep connector temperature the same as analyzer. Holding the connector in your hand or cleaning connector with compressed air can significantly change the temperature. Wait for connector temperature to stabilize before using in calibration or measurements.

- Do not touch mating plane surfaces. Natural skin oils and microscopic particles of dirt are difficult to remove from these surfaces.
- Do not set connectors contact-end down on a hard surface. The plating and mating plane surfaces can be damaged if the interface comes in contact with any hard surface.

- Wear a grounded wrist strap and work on a grounded, conductive table mat. This helps protect the analyzer and devices from electrostatic discharge (ESD).
VNA Front-Panel Tour

Click on an area of the image to learn more.

See Also

- PNA-X Models/Options
- N522xB Models/Options
- N523xB Models/Options
- Display area
- Rear-panel Tour
- PNA-X and N522xB
- N523xB

**Power Switch**

Used for choosing between power-on ( | ) and standby (O) state.

Learn to power ON and OFF the VNA.

**LCD Touchscreen**

VNA is equipped with a 12.1-inch TFT color, touch-sensitive LCD screen for displaying traces, scales, settings, softkeys and other measurement related information. The touchscreen LCD allows to manipulate softkeys by touching the LCD screen directly. For more on the LCD touchscreen, see Screen Display Tour.

**Note:**

Do not press the surface of the LCD touchscreen with a sharp object (e.g., a nail, pen, or screwdriver). Pressing the surface with a sharp-pointed object will damage the LCD screen surface or cause the screen to fail.

Valid pixels are 99.998% and more. Below 0.002% of fixed points of black, blue, green or red are not regarded as failure.

**Test Ports**

The models are available with 2 or 4 test ports.

See Specs for more information about the Test port connectors and Input damage levels.

The LED of each test port shows the source output status. When the signal comes from the port, its LED is lighted.

**Front panel Access Jumpers**

These connectors provide direct access to the VNA source and receivers. This allows you to make a wide variety of measurements and improve dynamic range.

See front panel jumpers specifications
Port 1 and Port 3 SW SRC OUT - COMB IN jumpers moved from rear-panel (J8 through J11) to front-panel to minimize path loss.

**Important:** All of the N5247B front-panel jumpers are EXTREMELY FRAGILE. Learn more.

**USB Hub**

This USB hub contains four USB ports to power your VNA peripherals. There are also four USB ports on the rear panel.

**Limitation:** The total power consumption for all eight USB ports is limited to 4.0 amps. If this limit is exceeded, all USB ports are disabled until a device is removed and power consumption falls below the limit. When first connected, Keysight ECal modules 8509x and N4431 draw significantly more current than other modules.
Note:

The **FIRST TIME** each USB device (ECal module, power sensor, and so forth) by serial number is connected to a specific VNA USB port, you must be logged in to the VNA with an **Administrator** account. Learn how.

When a **New Hardware Found** dialog appears, click **OK** to install the device.

After being installed, when that same USB device is connected to that same USB port, you can be logged in to the VNA with a Limited/User account.

**Ground terminal**

Connect a banana-type plug to this terminal for grounding to the VNA chassis.

**No probe power**

Probe power is NOT provided on VNA models. Learn more about Active Probing

**Hardkeys**

**INSTRUMENT Keys**

Manages the Traces and Channels on the VNA display.

<table>
<thead>
<tr>
<th>Hard Key</th>
<th>Invokes these Softkeys</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prev</td>
<td>Makes the previous Trace/Channel/Window active.</td>
</tr>
<tr>
<td>Next</td>
<td>Makes the next Trace/Channel/Window active.</td>
</tr>
<tr>
<td>Trace</td>
<td>Invokes the Traces softkey menu which allows you to manage traces.</td>
</tr>
<tr>
<td>Channel</td>
<td>Invokes the Channels softkey menu which allows you to manage channels.</td>
</tr>
<tr>
<td>Display</td>
<td>Invokes the Display softkey menu which allows you to manage display functions.</td>
</tr>
<tr>
<td>Setup</td>
<td>Invokes the Setup softkey menu which allows you to set up a measurement.</td>
</tr>
</tbody>
</table>

**RESPONSE Keys**

Performs operations on measurement traces after data is measured - not including Data Analysis operations.

<table>
<thead>
<tr>
<th>Hard Key</th>
<th>Invokes these Softkeys - Click to learn more</th>
</tr>
</thead>
<tbody>
<tr>
<td>Meas</td>
<td>S-Param</td>
</tr>
<tr>
<td></td>
<td>• Meas Class</td>
</tr>
<tr>
<td>Balanced Source/Topology</td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td></td>
</tr>
</tbody>
</table>

**Receivers**

**Waves**

**Auxiliary**

**Meas Setup**

- Conversions
- Correction
- Trace Hold
- Equation Editor
- Memory
- Time Domain
- Pulse Setup

**Format**

<table>
<thead>
<tr>
<th>Format 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format</td>
</tr>
<tr>
<td>Group Delay Aperture</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Format 2</th>
</tr>
</thead>
</table>

**Scale**

<table>
<thead>
<tr>
<th>Main</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autoscale</td>
</tr>
<tr>
<td>Scale</td>
</tr>
<tr>
<td>Reference Level</td>
</tr>
<tr>
<td>Reference Position</td>
</tr>
<tr>
<td>Scale Coupling</td>
</tr>
</tbody>
</table>

**Electrical Delay**

- Delay Time
- Delay Distance
- Distance Units
- Velocity Factor
- Media -Waveguide/coax
- Waveguide cutoff freq
## Constants
- System Z0
- Phase Offset
- Mag Offset
- Mag Slope

## Math
### Memory
- Data/ Memory Math
- Normalize
- Data Math
- Display
- 8510 Mode

## Analysis
- Conversions
- Equation Editor
- Statistics
- Uncertainty Analysis
- Limits
- Limit Table

## Time Domain
- Transform
- Start Time
- Stop Time
- Center Time
- Span Time
- TD Mode
- TD Toolbar
- Time Domain Setup

## Time Gating
- Gating
- Gate Start
- Gate Stop
<table>
<thead>
<tr>
<th><strong>Avg BW</strong></th>
<th><strong>Main</strong></th>
</tr>
</thead>
</table>
|            | Averaging  
|            | Averaging Restart  
|            | Average Type  
|            | IF Bandwidth  
|            | LF Auto BW  
| Smoothings | Smoothing  
|            | Smooth Percent  
|            | Smooth Points  
| Delay Aperture | Aperture Percent  
|            | Aperture Points  
|            | Aperture Frequency  

<table>
<thead>
<tr>
<th><strong>Cal</strong></th>
<th><strong>Main</strong></th>
</tr>
</thead>
</table>
|         | Basic Cal  
|         | Other Cals  
|            | Cal All  
|            | Smart Cal  
|            | ECal  
|            | Response Cal  
|            | Source Power Cal  
|            | Correction  

- Src Power Correct
- Interpolation
- Correction Methods
- Properties

Port Extension

- Select
- Port Extension
- Time
- Distance
- Velocity Factor
- DC Loss
- Port Extensions
- Auto Port Extension

Cal Sets & Cal Kits

- Cal Set
- Cal Set Viewer
- Cal Kit
- ECal
- Cal Pod
- Uncertainty Setup

Fixtures

- Apply Fixtures
- Power Comp
- Fixture Setup
- Cal Plane Manager
- Auto Fixture Removal
<table>
<thead>
<tr>
<th><strong>Marker</strong></th>
<th>Markers 1-7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Markers 8-15</td>
</tr>
<tr>
<td></td>
<td>Marker Setup</td>
</tr>
<tr>
<td></td>
<td>● Delta</td>
</tr>
<tr>
<td></td>
<td>● Discrete</td>
</tr>
<tr>
<td></td>
<td>● Type</td>
</tr>
<tr>
<td></td>
<td>● Format</td>
</tr>
<tr>
<td></td>
<td>● Coupled</td>
</tr>
<tr>
<td></td>
<td>● Marker Display</td>
</tr>
<tr>
<td></td>
<td>● Marker Table</td>
</tr>
<tr>
<td></td>
<td>● All Off</td>
</tr>
<tr>
<td><strong>Marker Functions</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>● Marker -&gt; Start</td>
</tr>
<tr>
<td></td>
<td>● Marker -&gt; Stop</td>
</tr>
<tr>
<td></td>
<td>● Marker -&gt; Center</td>
</tr>
<tr>
<td></td>
<td>● Marker -&gt; Span</td>
</tr>
<tr>
<td></td>
<td>● Marker -&gt; Ref Level</td>
</tr>
<tr>
<td></td>
<td>● Marker -&gt; Delay</td>
</tr>
<tr>
<td></td>
<td>● Marker -&gt; CW Freq</td>
</tr>
<tr>
<td></td>
<td>● Marker -&gt; SA</td>
</tr>
<tr>
<td><strong>Search</strong></td>
<td>Main</td>
</tr>
<tr>
<td></td>
<td>● Max Search</td>
</tr>
<tr>
<td></td>
<td>● Min Search</td>
</tr>
<tr>
<td></td>
<td>● Domain</td>
</tr>
<tr>
<td></td>
<td>● Domain Start</td>
</tr>
</tbody>
</table>
- Domain Stop
- Tracking

Peak
- Peak Search
- Peak Right >> Search
- << Peak Left Search
- Next Peak Search
- Threshold
- Excursion
- Peak Polarity
- Tracking

Target
- Target Search
- Target Right >> Search
- << Target Left Search
- Target Value
- Transition
- Tracking

Multi Peak & Target
- Multi Peak Search
- Peak Threshold
- Peak Excursion
- Peak Polarity
- Multi Target Search
- Target Value
Transition

Tracking

Bandwidth & Notch

- Bandwidth Search
- BW Ref To Marker/Peak
- BW Level
- Notch Search
- Notch Ref To Marker/Peak
- Notch Level
- Tracking

Comp & Sat

- Compression Search
- Comp Level
- Saturation Search
- Pmax Backoff
- Tracking

Normal Op Pt

- Normal Op Search
- Backoff
- Pin Offset
- Tracking

STIMULUS Keys

Controls settings that determine what data (stimulus range), and how data (sweep type and triggering), is measured.
<table>
<thead>
<tr>
<th><strong>Hard Key</strong></th>
<th><strong>Invokes these Softkeys - Click to learn more</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Freq</strong></td>
<td>Frequency Range</td>
</tr>
<tr>
<td></td>
<td>Frequency Offset Mode</td>
</tr>
<tr>
<td><strong>Power</strong></td>
<td>Main</td>
</tr>
<tr>
<td></td>
<td>• Power level</td>
</tr>
<tr>
<td></td>
<td>• RF Power</td>
</tr>
<tr>
<td></td>
<td>• Start Power</td>
</tr>
<tr>
<td></td>
<td>• Stop Power</td>
</tr>
<tr>
<td></td>
<td>• Power and Attenuators</td>
</tr>
<tr>
<td></td>
<td>Port Power</td>
</tr>
<tr>
<td></td>
<td>• Select</td>
</tr>
<tr>
<td></td>
<td>• Power level</td>
</tr>
<tr>
<td></td>
<td>• Start Power</td>
</tr>
<tr>
<td></td>
<td>• Stop Power</td>
</tr>
<tr>
<td></td>
<td>• Source State</td>
</tr>
<tr>
<td></td>
<td>• Coupling</td>
</tr>
<tr>
<td></td>
<td>Leveling &amp; Offsets</td>
</tr>
<tr>
<td></td>
<td>• Select</td>
</tr>
<tr>
<td></td>
<td>• Slope</td>
</tr>
<tr>
<td></td>
<td>• Offset</td>
</tr>
<tr>
<td></td>
<td>• Limit</td>
</tr>
<tr>
<td></td>
<td>• Offsets and Limits</td>
</tr>
<tr>
<td></td>
<td>• ALC Hardware</td>
</tr>
<tr>
<td></td>
<td>• Receiver Leveling</td>
</tr>
<tr>
<td><strong>Sweep</strong></td>
<td>Main</td>
</tr>
<tr>
<td></td>
<td>• Number of Points</td>
</tr>
<tr>
<td></td>
<td>• Sweep Type</td>
</tr>
<tr>
<td></td>
<td>• Start</td>
</tr>
<tr>
<td></td>
<td>• Stop</td>
</tr>
<tr>
<td></td>
<td>• X-axis Type</td>
</tr>
<tr>
<td></td>
<td>• Sweep Setup</td>
</tr>
<tr>
<td></td>
<td>Sweep Timing</td>
</tr>
</tbody>
</table>
- Sweep Time
- Dwell Time
- Sweep Delay
- Sweep Mode
- Sweep Sequence
- Fast Sweep

Source Control

- Frequency Offset
- Pulse Setup
- Balanced Source
- Phase Control
- DC Source

Segment Table

- Add Segment
- Insert Segment
- Delete Segment
- Delete All Segments
- Segment Table
- Show Table

<table>
<thead>
<tr>
<th>Trigger</th>
<th>Main</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hold</td>
</tr>
<tr>
<td></td>
<td>Single</td>
</tr>
<tr>
<td></td>
<td>Groups</td>
</tr>
<tr>
<td></td>
<td>Continuous</td>
</tr>
<tr>
<td></td>
<td>Manual Trigger</td>
</tr>
<tr>
<td></td>
<td>Restart</td>
</tr>
<tr>
<td></td>
<td>Trigger Source</td>
</tr>
<tr>
<td></td>
<td>Trigger</td>
</tr>
</tbody>
</table>
**UTILITY Keys**

Performs global VNA operations.

<table>
<thead>
<tr>
<th>Hard Key</th>
<th>Invokes these Softkeys - Click to learn more</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Save Recall</strong></td>
<td>File Recall</td>
</tr>
<tr>
<td></td>
<td>• Recall State</td>
</tr>
<tr>
<td></td>
<td>• Recall Register</td>
</tr>
<tr>
<td></td>
<td>• Recall Calset</td>
</tr>
<tr>
<td></td>
<td>• Recall Data</td>
</tr>
<tr>
<td></td>
<td>• Recall Order</td>
</tr>
<tr>
<td><strong>Save State</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Save State</td>
</tr>
<tr>
<td></td>
<td>• Auto Save</td>
</tr>
<tr>
<td></td>
<td>• Save State As</td>
</tr>
<tr>
<td></td>
<td>• Save Register</td>
</tr>
<tr>
<td></td>
<td>• Save Type</td>
</tr>
<tr>
<td></td>
<td>• Delete State</td>
</tr>
<tr>
<td><strong>Save Other</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Save Calset</td>
</tr>
<tr>
<td></td>
<td>• Save Data</td>
</tr>
<tr>
<td></td>
<td>• Save Screen</td>
</tr>
<tr>
<td></td>
<td>• Save User Preset</td>
</tr>
<tr>
<td></td>
<td>• Manage Files</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Macro</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Favorite 1</td>
<td></td>
</tr>
<tr>
<td>Favorite 2</td>
<td></td>
</tr>
<tr>
<td>Favorite 3</td>
<td></td>
</tr>
<tr>
<td>Macro 1</td>
<td></td>
</tr>
<tr>
<td>Macro 2</td>
<td></td>
</tr>
<tr>
<td>Macro 3</td>
<td></td>
</tr>
<tr>
<td>Key Setup</td>
<td></td>
</tr>
<tr>
<td>System</td>
<td>Main</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td></td>
<td>Show Taskbar</td>
</tr>
<tr>
<td></td>
<td>Move App to Back</td>
</tr>
<tr>
<td></td>
<td>Minimize Application</td>
</tr>
<tr>
<td></td>
<td>Exit</td>
</tr>
<tr>
<td></td>
<td>Security</td>
</tr>
<tr>
<td></td>
<td>Control Panel</td>
</tr>
<tr>
<td></td>
<td>Manage Files</td>
</tr>
</tbody>
</table>

System Setup

- Next/Prev Keys
- Preferences
- Sound
- Remote Interface
- LAN Status
- Code Emulation

Print

- Print
- Print to file
- Page Setup
- Print Colors

Help

- NA Help
- On The Web
- Error Display
- View Error Log
- About NA

Service
- Update Firmware
- Verification
- Adjustment Routines
- Diagnostics
- Option Enable

<table>
<thead>
<tr>
<th>Undo</th>
<th>Main</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Help</th>
<th>Launches the Help file.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Preset</th>
<th>Main</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**ENTRY Keys**

<table>
<thead>
<tr>
<th>Hard Key</th>
<th>Invokes these Softkeys</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OK</strong></td>
<td>Closes a dialog box and enters any values made in the dialog box.</td>
</tr>
<tr>
<td><strong>Cancel</strong></td>
<td>Closes a dialog box.</td>
</tr>
<tr>
<td><strong>Bk Sp</strong></td>
<td>Back Space. Backs up the cursor and deletes any previous selection.</td>
</tr>
<tr>
<td><strong>0 to 9</strong></td>
<td>Selects values for measurement settings, then press Enter or G/n - M/u - k/m to complete the selection.</td>
</tr>
<tr>
<td><strong>T/p</strong></td>
<td>Completes the value selection, assigning a unit of measurement.</td>
</tr>
<tr>
<td><strong>G/n</strong></td>
<td>– G/n (Giga/Nano) E12 or E-12</td>
</tr>
<tr>
<td><strong>M/u</strong></td>
<td>– M/u (Mega/micro) E6 or E-6</td>
</tr>
<tr>
<td><strong>k/m</strong></td>
<td>– k/m (kilo/milli) E3 or E-3</td>
</tr>
<tr>
<td><strong>Enter</strong></td>
<td>Enters the values that you select for the measurement settings.</td>
</tr>
<tr>
<td><strong>Decimal point</strong></td>
<td>Enters a decimal point to designate fractions of a whole number.</td>
</tr>
<tr>
<td><strong>+/-</strong></td>
<td>Plus - Minus Toggles between a positive and negative value entry if it is the first key pressed in the entry.</td>
</tr>
</tbody>
</table>

**Knob**
Rotate to increase or decrease the value of the active entry.

**Navigation Keys**

These keys allow you to navigate through menus and dialog boxes and select choices from the active entry toolbar.

<table>
<thead>
<tr>
<th>Hard Key</th>
<th>Invokes these Softkeys</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Left / Right</strong></td>
<td>Moves left and right through menus.</td>
</tr>
<tr>
<td></td>
<td>Moves tab-left and tab-right within dialog boxes.</td>
</tr>
<tr>
<td><strong>Up / Down</strong></td>
<td>Moves up and down through menus.</td>
</tr>
<tr>
<td></td>
<td>Behaves as follows in a dialog box:</td>
</tr>
<tr>
<td></td>
<td>• Modifies a numeric value</td>
</tr>
<tr>
<td></td>
<td>• Moves through items in a drop-down list</td>
</tr>
<tr>
<td></td>
<td>• Moves through options buttons in a group of option buttons</td>
</tr>
<tr>
<td><strong>Click</strong></td>
<td>Makes a selection just like a mouse click.</td>
</tr>
</tbody>
</table>
10 MHz Reference IN/OUT

10 MHz Reference Input  When a 10 MHz external reference signal is detected at this port, it will be used as the instrument frequency reference instead of the internal frequency reference.

10 MHz Reference Output  This BNC(f) connector outputs a frequency reference signal for use by other test equipment.

- Press Setup > Internal Hardware > Reference... to switch external/internal reference.
- From SCPI, use SENS:ROSC:SOUR
USB Hub

This USB hub contains two SuperSpeed USB ports to power your VNA peripherals. There is also one USB port below the LAN connector, on USB device port, and four USB ports on the front panel.

**Limitation:** The total current limit for all rear panel USB ports is 2.3 amps. The total current limit for all front panel USB ports is 2 amps. When first connected, Keysight ECal modules 8509x and N4431 draw significantly more current than other modules. See Specifications.

See Important First-time USB connection note.

USB Device

LAN Connector

This 10/100/1000 BaseT Ethernet connection has a standard 8-pin configuration and auto selects between the data rates.

Mini DisplayPort Connector

The Mini DisplayPort is a miniature DisplayPort connector for connection to external displays.

Line Power

SRC3 Connector

Provides a synthesizer output from 10 MHz to 13.5 GHz (Option XSB) and also requires Option 422 or 423.

GPIB Controller and Talker/Listener Ports

The PNA-X can be a GPIB Controller and Talker/Listener. Learn more.
**PCIe X4 Connector**

The PCIe X4 connector is a 4-lane slot for future enhancements.

---

**RF Path Access**

These connectors are NOT available on the N522x and N5264B models.

These connectors allow RF Path Configuration.

Ports 3 and 4 are not available on 2-port models.

N5247A - J8 thru J11 are moved to the front-panel.

---

**RF and LO OUT**

The RF OUT connector is NOT available on the N5264B

For the N5247B and N5227B:
- Added RF2 OUT (J12) for 4-port 110 GHz single sweep VNA. Enables driving two mmWave modules simultaneously. [Learn more.](#)

---

**Caution:** LO OUT has more power than previous VNA models.

[See specifications](#)

---

**IF Path Inputs**

Option 020 adds these connectors, which allow access to the VNA Receiver / IF paths.
These are labeled A, B, C/R1, D/R2, R.

- For 2-port models, use A, B, R1, R2.
- For 4-port models, use A, B, C, D, R.

See IF Path Configuration settings and block diagram.

**Power I/O**
See Details

**+28 VDC (BNC output)**

![28V output](image)

Used to power a noise source for the Noise Figure App.

- Maximum steady-state current = 300 mA.
- Maximum in-rush current = 500 mA.
- This output is protected with a self-healing fuse.

**External and AUX Trigger I/O**

![Trigger connectors](image)

**MEAS TRIG IN** - When enabled, VNA is triggered by signals on this connector. Learn more.

**MEAS TRIG RDY** - When enabled, VNA outputs a 'READY' signal on this connector to other devices. Learn more.

**AUX TRIG 1&2 IN** - When enabled, VNA accepts signals on these connectors which indicates that the external devices are ready to be triggered. Learn more.

**AUX TRIG 1&2 OUT** - When enabled, VNA outputs signals on these connectors either before or after a measurement. Learn more.
Test Set I/O
See Details

Bias IN and Fuses

Connect your DC Power Supply to apply Bias to the VNA ports through these BNC connectors.

- The bias fuses are rated for 0.5A. You are responsible to ensure that devices connected to the test port do NOT draw more current than 0.5A. This will occur, for example, if a calibration SHORT is connected to the test port with bias power ON. The fuse Keysight part number for the PNA-X is 2110-0824.

- The VNA will meet all of its RF specifications with bias up to 200 ma. As the DC bias is increased, corrected source match and directivity will degrade at low RF frequencies.

Material Handler I/O
See details.

Pulse I/O
See Details

CPU
See CPU Speed / Performance

See Determine Your VNA's CPU Version (Internet connection required)

Solid State Drive (SSD)
See Service Guide to learn how to remove the SSD. (Internet connection required)

See Preventing VNA SSD Problems
Screen Tour

Click on image areas to learn more.

Change VNA Display Settings

The VNA display is controlled by a third-party display driver. This driver is capable of producing many display effects which are beyond the basic needs of the VNA measurement environment. Keysight does not specify or warrant these display settings.

**WARNING:** The VNA display must remain in the 16 bit color setting in order to comply with international emissions regulations.

Restore VNA Display Settings

When using external programs to control the VNA, it is possible that the display settings may become
corrupt. When this occurs, the following will restore the VNA display settings:

- Press **System > Main > Minimize Application**.
- Right-click on the desktop, then click **Personalize**.
- On the Themes tab, select **Keysight Theme**.

See Also

- About the Touchscreen
- Front Panel Tour
- Marker Drag
- Expanded Mouse capabilities
- Learn how to Customize the Screen
- Learn to test the display screen.

---

About the Touchscreen

The VNA is equipped with a 12.1-inch Hi resolution, color, touch-sensitive LCD screen for displaying traces, softkeys, and other measurement related information. The touch screen LCD allows you to make measurement settings by touching the LCD screen directly with a finger.

**Important**

- Do not press the surface of the LCD screen with a sharp object such as a pen. This will damage the LCD screen surface or cause the screen to fail.
- Occasionally, a few pixels may appear on the screen as a fixed point of blue, green or red. This is not a failure of the LCD screen and does not affect the performance of your product.
- Because of the LCD screen, burn-in is not likely. In addition, we do not recommend using a screen-saver on the VNA.

---

**How to Calibrate or turn ON | OFF the Touchscreen**

Using **Hardkey /SoftTab /Softkey**
To turn Touchscreen ON or OFF:

1. Press Display > Display Setup.

2. Click Touchscreen to turn ON/OFF.

To calibrate Touchscreen:

Note: The Touchscreen is self calibrating and should not require calibration. However, if the touch locations are not accurate, perform the following procedure to calibrate the Touchscreen.

1. For Windows 7, perform the following procedure.
   
   a. In the PNA application, press System > Main > Control Panel.
   
   b. In the Control Panel, select Hardware and Sound > Tablet PC Settings > Calibrate the screen for pen or touch input.

2. For Windows 10, perform the following procedure.
   
   a. Close the PNA application.
   
   b. Press or select the Windows key on an external keyboard, or click on the Windows Start icon in the lower-left corner.
   
   c. Select Settings to access the Settings dialog.
   
   d. In the search field of the Settings dialog, type calibrate then select Calibrate the screen for pen or touch input.
e. The **Tablet PC Settings** dialog is displayed:

![Tablet PC Settings Dialog](image)

f. Select the **Display** tab.
g. Click on the **Calibrate**... button.

h. In the **User Account Control** dialog, click **Yes**.

i. Follow the instructions displayed on the screen.

**To calibrate Touchscreen when external monitor is connected to PNA:**

When a second monitor is added to a Windows PC where the primary monitor has a touchscreen, Windows may associate touch inputs with the window shown on the extended monitor rather than on the monitor that has the touchscreen. Since the computer in a PNA runs on the Windows operating system, the PNA can exhibit this condition.

**Note:** The Touchscreen is self calibrating and should not require calibration. However, if the touch locations are not accurate, perform the following procedure to calibrate the Touchscreen.

1. Minimize the PNA application window by selecting **File** then **Minimize Application**.

2. Press or select the Windows key on an external keyboard, or click on the Windows Start icon in the lower-left corner.

3. Select **Settings** to access the Settings dialog.

4. In the search field of the Settings dialog, type **calibrate** then select **Calibrate the screen for pen or touch input**.
5. The **Tablet PC Settings** dialog is displayed:

6. Select the **Display** tab.
6. Click on the **Setup...** button. At this point, both the PNA screen and the external monitor will display a white background. Only one screen will also display the following message:

   *Touch the screen to identify it as the touchscreen.*
   *If this is not the Tablet PC Screen, press Enter to move to the next screen. To close tool, press Esc.*

7. If the message is on the PNA screen, touch the screen. The calibration is now finished and the **Calibrate touch** dialog will reappear. Press **OK**.

8. If the message is not on the PNA screen, press **Enter** on the external keyboard until the message is displayed on the PNA screen. Touch the PNA screen, then press **OK** on the **Calibrate touch** dialog.

9. **Note:** If the calibration makes the touch operation worse, click on the **Reset** button in the **Tablet PC Settings** dialog to reset the calibration data.

The touchscreen ON | OFF setting remains until changed again from this menu, the Preferences dialog, or remotely.

**Active Entry**

Allows you easily select the tools. Learn more.

**Trace Status**

Provides details of each trace in the window. Highlighted trace indicates the active trace. Learn more.

**Entry Toolbar**

Along with the softkeys, allows numeric values to be entered for settings. Learn about all toolbars.

**Marker Readout**

Provides stimulus and response information for markers. Learn about customizing the marker readout area. See also Marker Drag.

**Softkeys**

The combination of hardkeys and softkeys allows easy access to all VNA features without a mouse.

**SoftTabs**

Pressing these tabs will display corresponding softkeys.

**Hardkeys**

Performs interface operations that are equivalent to those of keys in the INSTRUMENT keys, RESPONSE keys, STIMULUS keys and UTILITY keys on the front panel of VNA. Learn more.
Window Number

Provides window identification which is useful for remote programmers.

Status Bar

Provides detail about all aspects of the status of the analyzer. Learn more.

Stimulus Range

Displays the start and stop values of the sweep range.

Softkey Toolbar

These icons provide shortcuts to quickly select the softkey tools. Learn more.

Marker Drag

Drag a displayed marker across the trace using a finger (with touchscreen ON) or a mouse. Learn more.

Expanded Mouse Capabilities

- Cursor changes to a “hand” when hovering over a clickable object.
- Right-click on the Entry toolbar to launch a mouse-compatible numeric pad.

Windows

- Right-click or long press on a window area to make selections pertaining to that window.
- Double-click on a window area to maximize the window. To return to original window configuration, right-click on window area, then click Tile.
- Left-click on X-axis annotation to select the active channel/trace.
- Right-click on X-axis annotation and click Start/Stop/Center to change stimulus properties. Applications are not fully supported.
- Quickly change Scale, Reference Level, and Position. Learn how.
- Right-click on Y-axis annotation and click Scale to change Scale.
- Drag a trace from one window to another. Click or touch either the trace or the Trace Status. Drag the trace to another window, then release the mouse or lift your finger.

Traces
• Left-click a trace or Trace Status to make it the selected trace.

• Double-click on a trace or Trace Status to maximize the trace. Double-click again to return to the original trace configuration.

• Set a preference to **always** widen the active trace.

• Set a preference to **briefly** widen the active trace.

• Drag a trace from one window to another. Click or touch either the trace or the Trace Status. Drag the trace to another window, then release the mouse or lift your finger.

### Markers

• Right-click on a trace or Trace Status to add a marker.

• Right-click a marker to make selections pertaining to that marker, such as Marker Search or Function.

### Softkeys

• Use the Touchscreen or adjacent buttons to select from eight dynamic softkey menu choices.

• To Show the softkeys, press any front-panel hardkey and the corresponding softkey menu will be launched.

### Softkey Annotations

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Menu ...</td>
<td>Selection launches a dialog box.</td>
</tr>
<tr>
<td>Menu ▼</td>
<td>Selection launches another level of softkeys.</td>
</tr>
<tr>
<td>Item ▶</td>
<td>Indicates the item (marker, trace, window) is ON. Any number of objects can be ON.</td>
</tr>
<tr>
<td>Item ▼</td>
<td>Indicates the item (marker, trace, window) is OFF. Click to turn item ON.</td>
</tr>
<tr>
<td>Item ◁</td>
<td>Indicates the item IS selected.</td>
</tr>
<tr>
<td>Item ◆</td>
<td>Indicates the item is NOT selected. Click to select. Only one item in the collection can be ON.</td>
</tr>
<tr>
<td>Item *</td>
<td>Enter value in Entry toolbar.</td>
</tr>
<tr>
<td>Item on</td>
<td>Capitalization indicates the current setting.</td>
</tr>
<tr>
<td>Item OFF</td>
<td></td>
</tr>
</tbody>
</table>

158
Powering the VNA ON and OFF

The following is described in this topic:

- How to Log Off, Shut Down, or Restart the VNA
- ON Mode
- Turn OFF Autostart
- Shutdown

**Notes:** During boot up of Windows or of the Network Analyzer application program, do **NOT** press keys on the front panel, rotate the RPG knob, or connect a USB device. Doing so MAY lead to a front panel lockup state.

If the VNA front-panel keypad or USB ports are not responding, SHUTDOWN or RESTART the VNA.

---

### How to Log Off, Shut Down or Restart the VNA.

1. Minimize the VNA application

2. Click **Window Start**.

3. Choose from the following:

   - Shut down
   - Log off (closes programs)
   - Restart (shutdown and start)

   **OR**

1. Press the front-panel VNA power button (only for Shutdown).

**Note:** ONLY if the VNA is locked and you cannot operate the mouse or keypad - Press and hold the power button for at least four seconds. **This practice should be avoided!** Repeated shutdowns in this manner WILL damage the solid state drive. [Learn more about damaging the VNA solid state drive](#).
ON Mode

- To turn ON the VNA press the power button.
- The power indicator will change to green when power is ON.

Turn OFF VNA Autostart

The VNA application (835x.exe) always starts automatically when power is turned ON. To cause the VNA to NOT Autostart, do the following:

1. Minimize the VNA application.
2. From Windows Explorer, navigate to and double-click the following file: C:\Program Files(x86)\Keysight\Network Analyzer\Service\Toggle_PNA_Autostart.exe

The script toggles the VNA Autostart mode ON and OFF.

Shutdown Mode

- In shut down mode the current instrument VNA is NOT automatically saved before the VNA is powered OFF.
- When the VNA is again powered ON, a full system boot-up is performed and the VNA powers-up in the preset settings.
- A password may be required to resume VNA operation after being in Shutdown mode. Learn more.
- To guarantee that your measurements meet the VNA specified performance, allow the VNA to warm-up for 90 minutes after the power indicator has turned green.
- The power indicator will change to yellow when power is OFF.

Note: If the VNA is locked and you cannot operate the mouse or keypad, shut down the VNA by pressing and holding the power button for at least four seconds. This practice should be avoided! Repeated shutdowns in this manner WILL damage the solid state drive. Learn more about damaging the VNA solid state drive.

Unplugging the VNA
- Remove the power cord from the VNA ONLY when the power indicator is yellow, in either Hibernate or Shutdown mode. If the power cord is removed while the power indicator is green (VNA ON), damage to the solid state drive is possible.

- The indicator will remain yellow for several seconds after the power cord has been removed.
Traces, Channels, Windows, and Sheets on the Analyzer

It is critical to understand the meaning of the following terms as they are used on the analyzer.

- Traces - Managing (Trace Manager)
- Channels - Managing
- Windows - Managing
- Sheets - Managing

Other Quick Start topics

Traces

Traces are a series of measured data points. There is no theoretical limit to the number of traces. However, the practical limit is the maximum number of windows times the maximum number of traces per window (100).

In addition, one memory trace can be stored and displayed for every data trace. Learn more about Math / Memory traces.

Trace settings affect the presentation and mathematical operations of the measured data.

The following are Trace settings:

- Parameter
- Format and Scale
- Smoothing
- Correction ON / OFF
- Electrical Delay
- Phase Offset
- Trace Math
- Markers
Managing Traces

- How to Add a trace
- How to Select a trace
- How to Delete a trace
- How to Move a trace
- How to Maximize a trace
- How to perform Trace Hold (Max or Min)
- How to Create a new trace
- How to Change the trace parameter
- How to display a custom trace title (separate topic)
- How to display a wide active trace (separate topic)

How to Add a trace

The only measurements that can be selected are those in the same measurement class as is currently assigned to the channel. To select a measurement other than these, first select the appropriate measurement class to a new or existing channel. Learn how.

A trace must be selected (active) before its trace settings can be changed.

How to know which trace is Active?

**Using Hardkey/SofTab/Softkey**

1. For Traces 1-7, press Trace > Trace 1-7 > click left side Trace 1-7 small button

**Using a mouse**

1. Right click in the grid box and then select
(Example: Click on left side Trace 1 small button and Trace 1 is active when it turns green, so Trace 1 added).

2. For Traces 8-15, press Trace > Trace 8-15 > click left side Trace 8-15 small button

New Trace...
(Example: Click on left side Trace 9 small button and Trace 9 is active when it turns green, so Trace 1 is added).

3. Another method of adding traces is by pressing Trace > Trace 1-7 > New Traces....

4. For other traces numbers, press Trace > Trace Setup > Add Trace, then select New Trace, New Trace + Channel, New Trace + Window, New trace + Channel + Window, or New Traces....

How to Select a Trace

The only measurements that can be selected are those in the same measurement class as is currently assigned to the channel. To select a measurement other than these, first select the appropriate measurement class to a new or existing channel. Learn how.

A trace must be selected (active) before its trace settings can be changed.

How to know which trace is Active?

**Using Hardkey/SoftTab/Softkey**

1. Press Trace > Trace Setup > Select.
2. Select a trace number which corresponds to the desired measurement parameter.

**Using a mouse**

1. Click on Trace Status label of any trace above the grid box.
How to Delete a Trace

Using **Hardkey/SoftTab/Softkey**

1. For Traces 1-7, press **Trace > Trace 1-7** > click left side **Trace 1-7** small button

Using a mouse

1. Right-click the **Trace Status** label above the grid box, then click **Delete**
(Example: Click on left side Trace 1 small button and Trace 1 is inactive when it is not green).

2. For Traces 9-16, press **Trace** > **Trace 8-15** > click left side **Trace 8-15** small button
(Example: Click on left side Trace 9 small button and Trace 9 is inactive when it is not green).

3. For other traces numbers, press **Trace > Trace Setup > Delete Trace**, then select a trace number.

---

**How to Move a trace to a different Window**

You can **DRAG** a trace from one window to another, or...

**Using Hardkey/SoftTab/Softkey**

1. Press **Trace > Trace Setup > Trace Manager...**

2. Under the Window Column, reassign the active trace to another window number at the pulldown then click OK.

For some models

1. Press **Trace > Trace Setup > Move Trace....**

2. Select a window number in the following dialog, and then click OK.

**Using a mouse**

1. Right-click the **Trace Status** label above the grid box, then click **Trace Manager...**

2. Under the Window Column, reassign the active trace to another window number at the pulldown then click OK.

For some models,

1. Right-click the **Trace Status** label above the grid box, then click **Move Trace....**

2. Select a window number in the following dialog, and then click **OK**.

---

**Trace Manager** dialog box help
Trace Manager allows the user to see and modify all traces/channels/windows/sheets/formats in one table. The changes are updated immediately.

There is one row for each trace. By Clicking on a column heading will display the rows in ascending or descending order as defined by that column.

A row is selected by clicking on any cell in the row and the popup menu for the cell will appear. Multiple rows can be selected by click-drag-release. All selected rows will be highlighted and popup menu for the column will appear.

**Undo**  Reverse back to the previous settings

**Redo**  Change the settings again.

**Trace Column**
User is not able to edit trace numbers. User may select multiple rows; this allows user to delete multiple traces using the popup menu.

There is a special case where the trace entry in the row is empty. This is used to show when there is an empty window. If the users select this row and choose "Delete Trace", it will delete the empty window; this is a convenient feature; it is deleting a null trace.

New Trace  To add a new trace. Selecting the cell will open the "Meas" dialog.

Delete Trace  The selected trace will be deleted.

Table Columns  A pop up dialog is opened which allows user to define the columns visible in the table. The default table column is: Meas/Class/Channel/Windows/Sheet.

Meas Column

Selecting the cell will open the "Meas" dialog. User cannot select multiple rows from a Meas cell.

Class Column
Selecting the cell will open the "Measurement Class" dialog. User cannot select multiple rows from a Class cell.

When the measurement class is changed, all traces numbers on the currently active channel will be assigned to the new measurement class. The window and sheet settings will be the same, but the "Meas" setting will be changed to default values for the selected class.

Channel Column

Selecting the cell will open a pulldown, and will select the row. User can click and drag to select multiple rows.

User may use the pulldown to reassign the active trace to another channel and delete the active channel.

When the channel for a trace is changed, the "Meas" setting will likely be changed.
Window Column

Selecting the cell will open a pulldown, and will select the row. User can click and drag to select multiple rows.

User may use the pulldown to reassign the active trace to another window, delete the active window and all traces on that window and change the layout of the windows.

If the active row has no trace assigned, then the window pulldown will not allow user to change the window number.

Sheet Column

Selecting the cell will open a pulldown, and will select the row. User can click and drag to select multiple rows.

User may use the pulldown to reassign the active trace to another sheet, delete the active sheet and
all traces on that sheet and change the layout of the sheets.

If the active row has no trace assigned, then the sheet pulldown will not allow user to change the sheet number.

**Format Column**

Selecting the cell will open a pulldown, and will select the row.

User may use the pulldown to select the trace formats. When the measurement class is SA, the selectable formats are only LinM and LogM.
**Move Trace** dialog box help (E5080A only)

![Move Trace dialog box](image)

**Note:** Only ONE trace can be moved at a time.

1. Click the **Trace Status** label to select the trace to move.

2. **Move Trace N to window** - Transfer the selected trace to any Window listed or to a New Window.

---

**Trace Max**

How to maximize the active trace - the active trace is the ONLY trace on the screen display. All other traces are hidden.

**Using Hardkey/SoftTab/Softkey**

1. Press **Trace > Trace Setup > Trace Maximize (ON)**. With Trace Max (ON), select a different trace to make that trace visible.

2. To make all traces visible again, select **Trace Maximize (OFF)**.

**Using a mouse**

1. Right-click the **Trace Status** label above the grid box, then click **Trace Maximize**.

2. Double click on the active trace to make all traces visible again.
Trace Hold

How to hold the active trace at the maximum or minimum points.

Using **Hardkey/SoftTab/Softkey**

1. **Trace** > **Trace Setup** > **Trace Hold** > **OFF | Max | Min.**
2. **Restart** resets the trace.

Using a **mouse**

Not available

Maximum/Minimum trace hold can be applied with several conditions:

- Feature is applicable to any data trace, but NOT to memory traces.
- When the stimulus or any data post processing setting is changed, the trace hold data will be reset. These settings include:
  - Smoothing on/off.
  - Smoothing Aperture.
  - Gating on/off.
  - Transform on/off.
  - Conversion state change, conversion type change.
  - Data Math Function (Data/Mem) change.
  - Equation Editor state change, formula change.
  - Parameter change.
  - Formatting change.
- Minimum/maximum comparison is done with formatted data. For Smith and Polar formats, absolute data is used and not phase.
- Trace hold data can be recalled.
- Data save files formats
  - SnP does NOT save trace hold data
  - Citifile, CSV, MDF, PRN DOES save trace hold data
Note: Citifiles can be recalled and viewed in the VNA.

- Use SCPI commands to get trace hold data. If trace hold is active, then the data returned from the remote interfaces will be the trace hold data.

Channels

Channels contain traces. The analyzer can have up to maximum 500 independent channels.

Note: Actual maximum number of channel depends on the setup. A large number of NOPs and traces limit the maximum number of channel.

Channel settings determine how the trace data is measured. All traces that are assigned to a channel share the same channel settings. A channel must be selected (active) to modify its settings. To select a channel, click the Trace Status button of a Trace in that channel. The following are channel settings:

- Frequency range
- Power level
- Calibration
- IF Bandwidth
- Number of Points
- Sweep Settings
- Average
- Trigger (some settings are global)

Managing Channels

How to Select a Channel

A channel must be selected (active) before its settings can be changed.

To make a channel active, select a trace in that channel or click the Trace Status button of a Trace in that channel.
### How to Add a channel

#### Using Hardkey/SoftTab/Softkey

1. Press **Channel** > **Channel 1-8** > click left side **Channel 1-8** small button

#### Using a mouse

Not available
(Example: Click on left side Channel 1 small button and Channel 1 is active when it turns green, so Channel 1 is added).

2. For other channel numbers, press **Channel** > **Channel Setup** > **Add Channel**, then select **New Trace + Channel** or **New Trace + Channel + Window**.

    No programming commands are available for this feature

### How to Delete a channel

**Using Hardkey/SoftTab/Softkey**

1. Press **Channel** > **Channel 1-8** > click left side **Channel 1-8** small button

**Using a mouse**

Not available
2. For other channel numbers, press Channel > Channel Setup > Delete Channel, then select a channel.

---

**Windows**

Windows are used for viewing traces.

- The analyzer can show between 1 and 500 windows on the screen with the following limitations:
  - The COM property MaximumNumberOfWindows returns 500.
  - The SCPI status register can track the status of up to 576 traces.
- Each window can contain up to 100 traces.
- Windows are completely independent of channels.
- See Customize the analyzer screen to learn how to make other window settings.

The following is a window containing two traces. Both traces use the same channel 1 settings as indicated by the annotation at the bottom of the window.

![Image of a window containing two traces](image)

The window number shows in the lower-left corner of the window. The following shows window 5.
## Managing Windows

### How to Add a window

<table>
<thead>
<tr>
<th>Using Hardkey/SofTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Display &gt; Window 1-8</strong> &gt; click left side <strong>Channel 1-8</strong> small button</td>
<td>1. Right-click any area of grid box and then select <strong>New</strong></td>
</tr>
</tbody>
</table>
(Example: Click on left side Window 1 small button and Window 1 is active when it turns green, so Window 1 is added).

2. For other windows, press Display > Window Setup > Add Window, then select New Window, New Trace + Window, or New Trace + Channel + Window.

---

**How to Delete a Window**

**Using Hardkey/SoftTab/Softkey**

1. Press Display > Window 1-8 > click left side Channel 1-8 small button

**Using a mouse**

1. Right-click any area of grid box and then select Close Window.
(Example: Click on left side Window 1 small button and Window 1 is inactive when it is not green).

2. For other windows, press **Display > Window Setup > Delete Window**, then select a window.

### How to Move a Window to a different Sheet

**Note:** This feature is NOT available on M948xA and E5080A.

**Using Hardkey/SoftTab/Softkey**

1. Select a Window to move.

2. Press **Display > Window Setup > Move Window...**

3. Select a sheet number in the following dialog, and then click OK.

**Move Window dialog box help**

**Note:** Only ONE window can be moved at a time.

1. **Move Window N to Sheet N** - Transfer the selected window to any sheet listed or to a New Sheet.
How to Change Window Layout

**Note:** This feature is NOT available on M948xA and E5080A.

This is a window auto-layout option, for quicker selection instead of selecting the trace, channel, window and sheet separately. 7 auto-layout options are available.

**Using Hardkey/SoftTab/Softkey**

1. Press **Display > Window Setup > Window Layout**.
2. Select 1 Window, 2 Windows, 3 Windows, 4 Windows, 1 Trace per Window, 1 Channel per Window, or Tile Windows.

---

How to maximize the active window - the active window is the ONLY window on the screen display. All other windows are hidden.

**Using Hardkey/SoftTab/Softkey**

1. Press **Display > Window Setup > Window Max (ON)**. With Window Max (ON), select a different window to make that window visible.
2. To make all windows visible again, select **Window Max (OFF)**.

**Using a mouse**

1. Right-click in any area of the grid box and then select **Maximize**.

---

Sheet

Sheets are used to group VNA windows. The sheet tabs provide an easy way to switch multiple display settings quickly.

Features and actions that can performed with tabbed sheets:

- Add/Delete/Select sheet
- Move window to sheet
- Measurement can be performed on traces/channels in inactive sheets
- Easy setup for channel per window
- Easy setup for channel per sheet

## Managing Sheet

### How to Add Sheet

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Display &gt; Sheet Setup &gt; Add Sheet</strong>.</td>
<td>1. Click on the sheet tab.</td>
</tr>
<tr>
<td>2. The select one a <strong>New Sheet, New Trace + Sheet</strong> or <strong>New Trace + Channel + Sheet</strong>.</td>
<td></td>
</tr>
</tbody>
</table>

### How to Delete a Sheet

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Display &gt; Sheet Setup &gt; Delete Sheet</strong>.</td>
<td>1. Click on the sheet tab.</td>
</tr>
<tr>
<td>2. Then select a sheet.</td>
<td></td>
</tr>
</tbody>
</table>

### How to View a Sheet

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Display &gt; Sheet Setup &gt; Select</strong>.</td>
<td>1. Click on the sheet tab.</td>
</tr>
<tr>
<td>2. Then select a sheet.</td>
<td></td>
</tr>
</tbody>
</table>

### How to Change Sheet Title

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Display &gt; Sheet Setup &gt; Sheet Title</strong>.</td>
</tr>
<tr>
<td>2. In the pop up Sheet title box, enter the title and click OK.</td>
</tr>
</tbody>
</table>
How to Change Sheet Layout

This is a sheet auto-layout option, for quicker selection instead of selecting the trace, channel, window and sheet separately. 4 auto-layout options are available.

Using Hardkey/SoftTab/Softkey

1. Press Display > Sheet Setup > Sheet Layout.

2. Select 1 Sheet, 1 Trace per Sheet, 1 Channel per Sheet, or 1 Window per Sheet.
Quick Start Dialog

Quick start is a simple wizard which helps to setup the settings for typical measurements. This feature allows users to select from a set of pre-configured measurement layouts.

How to Open Quick Start Dialog Box

Using Hardkey/SoftTab/Softkey

1. Press Setup > Main > Quick Start...

The measurement comprises the following THREE steps.

Step 1: Layout Templates

You are able to select a layout template for typical measurements.

If "Create in new channel" checkbox is enabled, a new channel and window(s) will be created.
If "Create in new channel" checkbox is disabled, when a template is selected then the active channel will be used for the new measurements. If the active trace is displayed in a window with traces on other channels, then the trace will be deleted and a new window(s) will be opened for the new measurements.

If "Display Smith Chart" checkbox is enabled, the active trace in a window will turn to display Smith Chart.

If "Display Smith Chart" checkbox is disabled, no changes on the active trace in a window.

### Step 2: Stimulus Settings Dialogs

This step is used to set stimulus for the measurement.

![Frequency Sweep Settings](image)

**S-Parameters**

Option Required: None

Enter:

- Start/Stop frequency
- Center Frequency
- Span Frequency
- Sweep Type: Lin or Log Frequency
- Power

Creates S11 and S21 measurements in a single channel and window.
**Differential (Balanced)**

Option Required: None

Create Sdd11 and Sdd21 measurements in a single channel and window.

Enter:

- Start/Stop frequency
- Center Frequency
- Span Frequency
- Sweep Type: Lin or Log Frequency
- Power
- IF Bandwidth
- Number of Points

Learn more about **Differential (Balanced) measurements.**

Create R1 and B receiver measurements in a single channel and window. This allows you to view the DUT input power (R1) and output (B) power.
**Power Frequency Sweep**

Option Required: None

Enter:

- Start/Stop Frequency
- Center Frequency
- Span Frequency
- **Sweep Type:** Lin or Log Frequency
- Power
- IF Bandwidth
- Number of Points

Learn more about Power Sweep measurements.

Creates a power sweep while viewing R1, B, and S21 measurements in a single channel and window. This allows you to view the DUT input power (R1), output power (B), and DUT gain (S21).
**Power**

**Power Sweep**

Option Required: None

Enter:

- Start/Stop Power
- CW Frequency
- IF Bandwidth
- Number of Points

Learn more about Power Sweep measurements.

**Receiver Frequency Offset**

Option Required: S93080A

Enter

- Source Start

Creates Frequency Offset Measurement while viewing R1 and B receivers in a single channel and window.


- Source Stop
- Receiver Start
- Receiver Stop
- Power Level
- IF Bandwidth
- Number of Points

Learn more about FOM.

Creates an S11 measurement and enables Time Domain.

Enter:

- Start/Stop Time
- Transform Mode (Time Domain Settings dialog auto-select the start frequency if a LPF transform mode is selected.)
- Start/Stop Frequency
- Power
- IF Bandwidth
- Number of Points

Learn more about Time Domain measurements.

If any one of the SMC Measurements is selected in Step 1, the Mixer...
Quick Settings dialog will appear.

Enter:

- Input, LO, and Output Frequencies and configuration.

Learn more about SMC Measurements

Steps 3: Cal Wizard Dialog (Optional)

- If "Calibrate this setup" checkbox is enabled, the Cal Wizard Dialog will appear when Stimulus Settings Dialog is dismissed with the "OK" button.

- If "Calibrate this setup" checkbox is disabled, the Cal Wizard Dialog will NOT appear.
Basic Measurement Sequence

The following process can be used to setup all analyzer measurements:

**Step 1. Set Up Measurements**
Reset the analyzer, create a measurement state, and adjust the display.

**Step 2. Optimize Measurements**
Improve measurement accuracy and throughput using techniques and functions.

**Step 3. Perform a Measurement Calibration**
Reduce the measurement errors by performing a calibration.

**Step 4. Analyze Data**
Analyze the measurement results using markers, math operations, and limit tests.

**Step 5. Print, Save or Recall Data**
Save or print the measurement data.
**Frequency Blanking**

For security reasons, you can prevent frequency information from appearing on the screen and printouts.

**How to set Frequency Blanking**

**Using Hardkey/SoftTab/Softkey**

1. Press **System > Main > Security**.

**Using a mouse**

1. Click **Utility**.
2. Select **System**.
3. Select **Security**.

---

**Security Setting** dialog box help

![Security Setting dialog box](image)

**Notes**

- To learn how to erase memory before moving your analyzer out of a secure area, see [http://na.support.keysight.com/pna/security.html](http://na.support.keysight.com/pna/security.html).
- VNA 'Undo' is disabled with **High** and **Extra** security levels. [Learn more](http://na.support.keysight.com/pna/security.html).

**Security Levels**

**None** - All frequency information is displayed on the screen and printouts.

**Low** security level - Frequency information is blanked from the following:
- Display annotation
- Calibration properties
- All tables
- All toolbars
- All printouts

**External sources** - See Also: Preference to Deactivate External Devices on Preset. **Note:** Frequency Blanking is fully supported ONLY on Keysight MXG sources with option 006. On MXG models without option 006 and all PSG models, the window state is turned OFF. When the "local" button is clicked on the source, then frequency is re-displayed.

**High** security level - Low security level settings PLUS:

- GPIB console is inactive

**Extra** security level - High security level settings PLUS:

- All ASCII data saving capability (.snp, .prn, .cti) is saved without frequency information. The X-axis information is replaced with data point numbers. Before A.08.50, saving these file types was NOT allowed.

- Mixer setup files (*.mxr) can NOT be saved.

**For ALL security levels:**

Frequency information is **NOT** blanked from the following:

- Service Adjustment Programs
- Your COM or SCPI programs.

**Instrument State and Cal Sets**

The security level is always saved and recalled with an instrument state. However, the instrument state may contain a Cal Set or link to a Cal Set. Learn more. This may influence the security level when the instrument state is recalled. Here is how.

- When a new Cal Set is created at the end of a calibration, the current system security level is stored with it.
- The only way to change an existing Cal Set’s security level is by writing a new calibration into the Cal Set.

- When later applied to a channel, if the Cal Set has a **higher** security level than the current system security level, the system security level will become upgraded to that of the Cal Set.

- When saving an instrument state to either a *.csa or *.cst file, the security levels of the system and Cal Set are saved separately. When recalled, the higher security level of the two is applied.

- To view the security level of a Cal Set, see **Cal Set Properties**.

<table>
<thead>
<tr>
<th>Re-displaying frequency information</th>
</tr>
</thead>
<tbody>
<tr>
<td>- When in <strong>Low</strong> security level, do any of the following:</td>
</tr>
<tr>
<td>- Revisit this dialog box and select <strong>None</strong></td>
</tr>
<tr>
<td>- Perform an <strong>instrument preset</strong></td>
</tr>
<tr>
<td>- Recall an Instrument State/Cal Set with security level of <strong>None</strong>.</td>
</tr>
<tr>
<td>- When in <strong>High</strong> or <strong>Extra</strong> security level, do any of the following:</td>
</tr>
<tr>
<td>- Perform an <strong>instrument preset</strong></td>
</tr>
<tr>
<td>- Recall an Instrument State/Cal Set with security level of <strong>None</strong>.</td>
</tr>
</tbody>
</table>
**Internal Second Source**

The following VNA models include an internal second source.

<table>
<thead>
<tr>
<th>Model</th>
<th>Total # of Ports</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNA-X Opt 224</td>
<td>2</td>
</tr>
<tr>
<td>ALL PNA-X models</td>
<td>4</td>
</tr>
<tr>
<td>N522xA Opt 400's</td>
<td></td>
</tr>
</tbody>
</table>

**How to use the second source**

- Set frequency using the **Frequency Offset Opt S93080A dialog**.
- Set power using the **Advanced Power dialog**.
- **Source power calibration** of the second source is performed as usual.
- Using FCA, **click the LO button** to set frequency and power.
- The **specifications** of the second source are the same as source 1.

**Benefits / Uses of the second source**

- Up to five times faster than stepping an external source.
- Measure Mixers with internal swept or fixed LO.
- Measure TOI or Intermodulation distortion.

**Internal Second Source Restrictions**

Source 1 and Source 2 are available at specific ports as follows:

**4-port models**

- Source 1 power is available at Port 1 OR Port 2; NOT at both ports simultaneously.
- Source 2 power is available at Port 3 AND Port 4; BOTH ports simultaneously. (Although it is possible, the VNA firmware typically prevents both ports from sweeping simultaneously for measurement integrity.)
purposes.)

- Other routing configurations are possible using the RF Path Configurator.

**PNA-X Opt 224 (PNA-X 2-port model):**

- Source 1 power is available at **Port 1** OR **Port 2**; NOT at both ports simultaneously.
- Source 2 (**SRC 2**) power is available at **Out 1** AND **Out 2**; BOTH ports simultaneously.
- Other routing configurations are possible using the RF Path Configurator.

**Remotely Accessing the Internal Second Source**

See **Remotely Specifying a Source Port.**
Networking and Connecting the VNA

The VNA as a PC

- VNA User Accounts and Passwords
- Drive Mapping
- Using VNC to Control the VNA User Interface

GPIB / COM Programming

- Configure for COM/DCOM Programming
- Configure for GPIB, SCPI, and SICL

Controlling External Devices

- Configure an External Device
- E5091 TestSet Control
- External Testset Control
- Interface Control Feature
- TestSetIO Connector
- Handler IO Connector
Preferences

Preferences are settings that survive a Preset or Shutdown. Preferences are listed on this page with links to locations that provide more information.

### How to set Preferences

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>System &gt; System Setup &gt; Preferences</strong>...</td>
<td>1. Click <strong>Utility</strong>.</td>
</tr>
<tr>
<td></td>
<td>2. Select <strong>System</strong>.</td>
</tr>
<tr>
<td></td>
<td>3. Select <strong>System Setup</strong>.</td>
</tr>
<tr>
<td></td>
<td>4. Select <strong>Preferences</strong>.</td>
</tr>
</tbody>
</table>

Preferences dialog box help
Preferences survive a Preset and a Shutdown.

A checked box makes the following statements true unless stated otherwise.

- **Avg: On PRESET set two-point group delay aperture** (Default) - Group delay aperture is set to 11 points.

- **Avg: On PRESET set two-point group delay aperture** - Group delay aperture set to 2 points. Learn more.

- **Avg: Calculate Group Delay using legacy PNA Math** (Default) - Do not use legacy group delay aperture computation methods.

- **Avg: Calculate Group Delay using legacy PNA Math** - Use legacy group delay aperture computation methods.

- **Cal: Always use Internal Trigger during cal** - Only use Internal Trigger source

- **Cal: Always use Internal Trigger during cal** (default) - use the other Trigger source

- **Cal: ECAl Extrapolation for IMD** (default)

- **Cal: ECAl Extrapolation for IMD** - Allows Swept IMD and IMDx channels to be calibrated beyond the stop frequency of the ECAl module by extrapolating the error terms. Learn more.
Cal: For Frequency Offset, use Primary Frequencies (default)

☑ Cal: For Frequency Offset, use Primary Frequencies - Use when making mmWave measurements without a test set. [Learn more.]

This setting only affects calibrations performed using SCPI. Cals performed from the User Interface ALWAYS offer a choice to save to a named Cal Set.

☐ Cal: (SCPI only) Auto-generate a User Cal Set (default) - Completed calibrations are automatically saved to Cal Registers; NOT to User Cal Sets.

☑ Cal: (SCPI only) Auto-generate a User Cal Set - Completed calibrations are automatically saved to an auto-named User Cal Set. Caution: this can cause a lot of saved User Cal Sets. [Learn more.]

The following message appears when both the Cal Set choices above and below are selected:

"Cal: Auto-save preferences conflict "

Cal: (SCPI) Auto-save to User Cal Set (above)- or - Cal: (SCPI) Auto-save to current Cal Set (below)

Uncheck one of these.

This setting only affects calibrations performed using SCPI. Cals performed from the User Interface ALWAYS offer a choice to save to a named Cal Set.

☑ Cal: (SCPI) Auto-save to current Cal Set - Always automatically save a completed Cal to the Cal Set that is currently selected on the specified channel, which could be the channel Cal Register. If the channel does not yet have a selected Cal Set, the Cal will be saved to a new User Cal Set with an automatically-generated name.

☐ Cal: (SCPI) Auto-save to current Cal Set (default)- Do NOT automatically save a completed Cal to the Cal Set that is currently selected on the specified channel.

☑ Cal: Use legacy behavior for Series-C & Shunt-L fixtures - Legacy operation provided as a user-convenience for backward compatibility.

☐ Cal: Use legacy behavior for Series-C & Shunt-L fixtures (default) - Do NOT use legacy operation.

☐ Display: Selected trace changes width briefly. (default) - The selected trace does NOT change width briefly in order to improve visibility.

☑ Display: Selected trace changes width briefly.

☐ Display: Selected Trace is wider. (default) - The selected trace is the narrow, default size.

☑ Display: Selected Trace is wider. The active (selected) trace is always wider.
Display: Touchscreen ON (default) - Selections can be made by touching the screen.

Display: Touchscreen ON - Selections can NOT be made by touching the screen.

Ext Device: De-activate on PRESET and recall. External devices are de-activated when the VNA is Preset or when a Instrument State is recalled.

Ext Device: De-activate on PRESET and recall. External devices remain active when the VNA is Preset or when a Instrument State is recalled.

Learn more about External Devices.

Ext Reference: Modify Settings on Preset and Recall - External Reference settings will be affected by Recall/Preset.

Ext Reference: Modify Settings on Preset and Recall - External Reference settings will be maintained until changed.

Limit: Draw failed trace segments in red Failed segments are drawn in red. Learn more.

Limit: Draw failed trace segments in red Failed data points (dots) are drawn in red.

Limit: Draw Limit Lines in Red Limit lines are drawn in the same color as the trace.

Limit: Draw Limit Lines in Red All Limit lines are drawn in Red.

Limit: Test the nearest measurement point - When the stimulus of measurement point is not the same as the limit test point, the nearest limit test point is used for pass/fail judgement.

Limit: Test the nearest measurement point - The pass/fail is judged at only the stimulus of limit test point.

Markers: Coupling controls on/off state of markers - Turning a marker on or off will have no effect on the markers on other traces.

Markers: Coupling controls on/off state of markers - With Coupled Markers ON, when a marker is turned on, the same-numbered marker on all coupled traces will also be turned on. Likewise, turning off a marker will turn it off on all coupled traces.

Markers: On Preset, Coupled Markers is ON - Coupled Markers is OFF after Preset

Markers: On Preset, Coupled Markers is ON - Coupled Markers is ON after Preset

Markers: On Preset, Coupling Method is Channel - Marker Coupling Method is set to ALL after Preset.

Markers: On Preset, Coupling Method is Channel - Marker Coupling Method is set to Channel after Preset.
Marker: On Preset, set BW/Notch search reference to Peak - BW/Notch marker search reference is set to current marker position after Preset.

Marker: On Preset, set BW/Notch search reference to Peak - BW/Notch marker search reference is set to peak after Preset.

Marker: Programming treats Mkr 10 as Reference A marker programming command that includes 10 as its marker number argument will operate on the Reference Marker (NOT the general-purpose Marker 10). See Marker commands.

Marker: Programming treats Mkr 10 as Reference A marker programming command that includes 10 as its marker number argument will operate on the general-purpose Marker 10 (NOT the Reference marker).

Marker: Use single marker for marker search (default) - Use one marker for marker search. Sub Marker is displayed and used for Bandwidth, Notch searches.

Marker: Use single marker for marker search - Use multi marker for marker search.

Meas: Mathematical offset for receiver attenuation The reported test port receiver power is mathematically offset by the amount of receiver attenuation. Default for all models.

Meas: Mathematical offset for receiver attenuation The reported test port receiver power is NOT mathematically offset by the amount of receiver attenuation.

Learn more.

Meas: Mathematical offset for source attenuation The reported reference receiver power is mathematically offset by the amount of source attenuation.

Meas: Mathematical offset for source attenuation The reported reference receiver power is NOT mathematically offset by the amount of source attenuation.

Learn more.

Memory: Data Math 8510 Mode Standard data processing chain.

Memory: Data Math 8510 Mode Simulate the Keysight 8510 data processing chain as it pertains to Trace Math and Memory. Learn more.

Memory: Interpolate ON is default condition Set memory interpolation to OFF as the default.

Memory: Interpolate ON is default condition Set memory interpolation to ON as the default. Learn more.

Power: On Preset turn power on Instrument Preset always turns source power ON.

Power: On Preset turn power on When the current source power setting is OFF, source power remains OFF after Preset. When the current power setting is ON, source power is turned ON after Preset. Learn more.

For SCPI behavior only. Learn more.

Power: Report source unleveled events as errors Source unleveled events are reported as
errors.

- **Power: Report source unlevelled events as errors** Source unlevelled events are NOT reported as errors.

- **Power: Report when receiver is overloaded** A warning message is displayed on the VNA screen indicating that a receiver is overloaded or in compression. The displayed data is probably not accurate. One error per sweep appears and is reported in the Error Log.

- **Power: Report when receiver is overloaded** Do NOT show overload warnings on the screen or report these errors in the error log.

- **Power: Force RF power Off at end of sweep** - Turn RF power Off during a retrace of single-band frequency or segment sweeps.

- **Power: Force RF power Off at end of sweep** (default) - Leave RF power On during a retrace of single-band frequency or segment sweeps. Learn more.

- **Power: Turn Source Power Off when receiver is overloaded.** (Default) - Power remains ON when a receiver is overloaded.

- **Power: Turn Source Power Off when receiver is overloaded.** Turn OFF power to ALL ports when a receiver is overloaded. A notification dialog appears. Click OK, then lower the power level, then turn power ON. (Click Stimulus, then Power)

- **Power: Use Start Power during Power Sweep retrace** At the end of a power sweep, while waiting to trigger the next sweep, the VNA maintains source power at the start power level.

- **Power: Use Start Power during Power Sweep retrace** Maintain source power at the STOP power level. Learn more.

- **Preset: Confirm preset** - When Preset hardkey button is pressed, VNA firmware immediately presets (Hardkey is required only).

- **Preset: Confirm preset** (default) - When Preset > Preset is pressed, VNA firmware immediately presets (Hardkey and Softkey are required).

- **Preset: On Preset enable TDR**. Enable TDR on preset and power-up.

- **Preset: On Preset enable TDR**. Do not enable TDR on preset and power-up.

- **Preset: On Preset show Quick Start dialog** - Open Quick Start dialog on Preset.

- **Preset: On Preset show Quick Start dialog** - Do not open Quick Start dialog on Preset.

- **Recall: Softkey order is most recently used** - Recall softkey order which is most recently used.

- **Recall: Softkey order is most recently used (default)** - Do NOT recall softkey order which is most recently used.
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Scale:</strong></td>
<td><strong>On Preset Couple scale to Window</strong> - Scale coupling is set to Window when <strong>Preset</strong>.</td>
</tr>
<tr>
<td><strong>Scale:</strong></td>
<td><strong>On Preset Couple scale to Window (default)</strong> - Scale coupling is set to Off by default when <strong>Preset</strong>.</td>
</tr>
<tr>
<td><strong>Sweep:</strong></td>
<td><strong>On Preset set Sweep Mode to Stepped</strong> - Sweep Mode set to Stepped after Preset.</td>
</tr>
<tr>
<td><strong>Sweep:</strong></td>
<td><strong>On Preset set Sweep Mode to Stepped</strong> - Sweep Mode set to Auto after Preset.</td>
</tr>
<tr>
<td><strong>Sweep:</strong></td>
<td><strong>Use only ramp sweeps for Auto Sweep Mode</strong> - Auto Sweep Mode set to use continuous ramp sweeps after Preset.</td>
</tr>
<tr>
<td><strong>Sweep:</strong></td>
<td><strong>Use only ramp sweeps for Auto Sweep Mode</strong> - Auto Sweep Mode set to not use ramp sweeps after Preset.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>Enable sound (default)</strong> - Instrument speaker turns ON.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>Enable sound</strong> - Instrument speaker turns OFF.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>On Power-on show dialog if detect mm testset</strong> - Display the Millimeter Configuration dialog after power-on if a millimeter test set is detected.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>On Power-on show dialog if detect mm testset</strong> - Hide the Millimeter Configuration dialog after power-on if a millimeter test set is detected.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>On Power-on show Keys toolbar</strong> - Display softkey toolbar after power-on.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>On Power-on show Keys toolbar</strong> - Hide softkey toolbar after power-on.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>On VNA Start-up Run Self Tests (default)</strong> - Module self tests and connection checks are executed at start up.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>On VNA Start-up Run Self Tests</strong> - Module self tests and connection checks are NOT executed at start up.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>Use keyboard to navigate softkeys</strong> - Enable the keyboard to browse the softkeys.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>Use keyboard to navigate softkeys (default)</strong> - Disable the keyboard to browse the softkeys.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>Optimize memory for use with many channels</strong> - (M9485A only) The maximum number of channels will be extended but measurement speed may be decreased. The maximum number of channels depends on PC memory, NOP and traces.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>Optimize memory for use with many channels (default)</strong> - (M9485A only) Standard mode (No memory optimization)</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>Use parallel processing</strong> (default) - Enable parallel processing in the CPU which provides higher calculation speeds.</td>
</tr>
<tr>
<td><strong>System:</strong></td>
<td><strong>Use parallel processing</strong> - Disable parallel processing in the CPU.</td>
</tr>
</tbody>
</table>
System: Set front panel remote state when a SCPI command is received - Enable changing from local to remote status when a SCPI command is received.

System: Set front panel remote state when a SCPI command is received - Disable changing from local to remote status when a SCPI command is received.

Sets the scope of External Trigger Output signal properties. The VNA is Preset after changing this setting.

Trigger: External Trigger OUT is Global Channels can have different External Trigger OUT settings. Default for PNA-X and N522xA models. On the Trigger Setup dialog, Trigger Mode = Point is ignored for external triggering.

Trigger: External Trigger OUT is Global All channels have same External Trigger OUT settings. Default for VNA “C” and PNA-L models. Aux Trig OUT properties apply to all channels except the Per Point setting. To set Per Point for specific channels: On the Trigger Setup dialog, set Trigger Scope = Channel, under Channel Trigger State, select the channel, and set Trigger Mode = Point.

See External Triggering dialog.

<table>
<thead>
<tr>
<th>The More buttons launch dialogs that contain predefined preferences:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Saves</strong> - <strong>Define Data Saves</strong> - While not explicitly called Preferences, all of these settings survive a shutdown. Learn more.</td>
</tr>
<tr>
<td><strong>Power Limit</strong></td>
</tr>
<tr>
<td><strong>Offsets and Limits</strong> - Sets Power Limits and Offsets. Learn more.</td>
</tr>
<tr>
<td><strong>Transparency</strong>...</td>
</tr>
<tr>
<td><strong>Dialog Transparency</strong> - Some dialogs can be viewed in various levels of transparency. Learn more.</td>
</tr>
<tr>
<td><strong>Language</strong>...</td>
</tr>
<tr>
<td><strong>Help</strong> - Sets the language of the built-in help (English or other localized language). Learn more.</td>
</tr>
<tr>
<td><strong>User Preset</strong>...</td>
</tr>
<tr>
<td><strong>User Preset</strong> - Specify the Instrument State file that the analyzer will use when Preset. Learn more.</td>
</tr>
<tr>
<td><strong>Page Setup</strong>...</td>
</tr>
<tr>
<td><strong>Page Setup</strong> - Standard printer settings (Paper, Orientation, and Size) do NOT survive a shutdown. All other settings DO survive a shutdown. Learn more.</td>
</tr>
</tbody>
</table>
Colors...

**Display Colors** - Sets display items to custom colors. [Learn more.]

**Print Colors** - Sets print items to custom colors. [Learn more.]

Toolbars...

**Show Toolbars/Other Bars** - Select toolbars to display.

**Defaults** - Restore preferences to their default values.

**Global Sources...** - Allows a source to be set globally and retain its settings even after an instrument preset. [Learn more.]

Millimeter settings

Sets MM Wave configurations. [Learn more.]

Although they are called preferences, the following settings do NOT survive a shutdown.

<table>
<thead>
<tr>
<th>Calibration</th>
<th>UI Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show or not, the first 'Method' Page of the Cal Wizard.</td>
<td>Cal Preferences</td>
</tr>
<tr>
<td>Set and order default Cal Types</td>
<td>Cal Preferences</td>
</tr>
<tr>
<td>Perform orientation of the ECal module during calibration?</td>
<td>ECal Wizard</td>
</tr>
<tr>
<td>Specify ECal port mapping when orientation is OFF</td>
<td>ECal Wizard</td>
</tr>
<tr>
<td>Show or hide custom Cal Windows during Cal</td>
<td>Cal Window (remote commands only)</td>
</tr>
</tbody>
</table>
Using VNC to Control the VNA User Interface

VNC (Virtual Network Computing) allows you to control the User Interface of a VNA from any PC. The VNA display appears on the connected PC display. Mouse and keyboard control can occur from both the VNA and PC, although not simultaneously.

**Note:** For similar functionality, use Windows Remote Desktop. A display setting of 1280X800 pixels is recommended. See the Windows help file for more information.

Both the VNA and PC must be connected to the same network. The responsiveness of the VNA while using VNC is dependent of the speed of your internet connection.

Every VNA is shipped with VNC installed. However, you must download and install the VNC software onto the PC.

The following procedures can help you configure VNC to view and control the VNA application from your PC.

**On the VNA, run VNC Server**

To do this:

1. Click View, then Minimize Application.
2. Click **Start**, then **All Programs**, then **TightVNC**, then **Tight VNC Server (Application Mode)**, then **Launch VNC Server**.

   * When the server is running, the icon is visible in the lower right corner of the display. If not visible, click the up arrow to expand the list of icons.
   * The first time you run VNC Server, you must first set a password to control access from remote PCs.

**On the PC, run VNC Viewer**

To do this:

2. From the PC Desktop, click **Start**, then **Programs**, then **TightVNC**, then **TightVNC Viewer**
3. When prompted for the Hostname, type the full computer name or IP address of the VNA.
4. When prompted for the password, type the password you set when configuring VNC on the VNA.
LXI-1.1 and VXI-11.3 Compliance

PNA-X, N522x, and VNA-C models are LXI-1.1 and VXI-11.3 compliant.

LXI-1.1 Compliance

A VNA is LXI-1.1 compliant if the logo appears on the dialog box shown below.

Learn more about LXI at http://www.lxistandard.org/

VXI-11.3 Compliance

To be compliant with VXI-11.3, the VNA must have been either:

- Shipped from the factory with VNA version A.08.20 or higher, or
- Had the Hard Disk Drive (HDD) upgraded since about June 2008 when A.08.20 was released and using VNA Rev. A.08.20 or higher.

Learn more about VXI at http://www.vxi.org/

LAN Status

When a LAN connection is used with the VNA, the LAN Status dialog allows you to see the IP address and other LAN connection properties.

How to view LAN Status

Using Hardkey/SoftTab/Softkey

1. System > System Setup > Lan Status....

Using Menus

1. Click Utility.
2. Select System.
4. Select LAN Status.

LAN Status dialog box help
Indicator  Shows the current status of the LAN connection.

NORMAL - Indicates that the VNA LAN is ready for communication.

IDENTIFY - Indicates that a remote computer has invoked an LXI identification operation on the VNA using the web-based interface or LXIDeviceIDState COM property.

FAULT - Indicates that the VNA LAN interface is not connected to the Internet.

IP Address  Shows the current IP address of the VNA.

IPv6 Link-Local Address  Shows the current IPv6 address of the VNA.

Computer Name  Shows the full computer name of the VNA. Learn how to change this. If you see the IP address listed here, that means there is no DNS server specified in the network setup.

MAC Address  Shows the unique address of the VNA computer. Also known as HostID.

LAN Reset  Provides a LAN Configuration Initialize (LCI) mechanism. Press to return the following settings to factory default conditions:

- IP Address Configuration (DHCP): Enabled
- ICMP Ping Responder: Enabled
- Web Password for configuration: Resets the password to 'Keysight'.

Web Server Software
If your VNA is LXI Class C compliant (see above), you can connect to the VNA using a web browser over an internet connection.

To do this, when the above dialog indicates a **NORMAL** condition:

1. From a web browser, type `http://<your_VNA_computer_name>`.
   For example, to connect to the fictitious VNA in the dialog above, type: `http://vna1-22`

2. Type the log on User Name and Password

3. You will see the welcome screen with connection links.
Dialog Transparency

Most VNA dialogs can be made to appear with various amounts of transparency. This allows you to view the VNA traces through the dialog as you make dialog settings.

How to set Transparency Level

There are three ways to make the transparency level setting:

1. Right-click in any non-control area of a dialog that allows transparency to see the following selections:

   ![Transparent levels selection]

2. In tabbed dialogs, cycle through the above transparency settings by pressing multiple times.

   ![Tabbed dialog]

3. Launch the Transparency dialog (below) from the Preferences dialog.

   ![Preferences dialog]

This setting is not programmable
Dialog Transparency dialog box help

Note: This single Transparency setting applies to ALL supported VNA dialogs.

- Opaque (NOT Transparent) - Default setting
- 8% Transparency
- 25% Transparency
- 50% Transparency

Double-click changes transparency - When checked, cycle through the above transparency settings by double-clicking in any non-control area of a dialog that allows transparency.

Notes

- The transparency setting is stored as a VNA Preference.
- The setting survives a VNA Shutdown and Preset.
- It is NOT saved and recalled with instrument state.
# Keyboard Shortcuts

<table>
<thead>
<tr>
<th>Function</th>
<th>Shortcut</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Instrument Keys</strong></td>
<td></td>
</tr>
<tr>
<td>PREV</td>
<td>PAGE UP</td>
</tr>
<tr>
<td>NEXT</td>
<td>PAGE DOWN</td>
</tr>
<tr>
<td>TRACE</td>
<td>SHIFT + CONTROL + PERIOD</td>
</tr>
<tr>
<td>CHANNEL</td>
<td>SHIFT + CONTROL + H</td>
</tr>
<tr>
<td>DISPLAY</td>
<td>SHIFT + CONTROL + D</td>
</tr>
<tr>
<td>SETUP</td>
<td>SHIFT + CONTROL + U</td>
</tr>
<tr>
<td><strong>Response Keys</strong></td>
<td></td>
</tr>
<tr>
<td>MEAS</td>
<td>SHIFT + CONTROL + M</td>
</tr>
<tr>
<td>FORMAT</td>
<td>SHIFT + CONTROL + F</td>
</tr>
<tr>
<td>SCALE</td>
<td>SHIFT + CONTROL + S</td>
</tr>
<tr>
<td>MATH &gt; Memory</td>
<td>SHIFT + CONTROL + O</td>
</tr>
<tr>
<td>MATH &gt; Analysis</td>
<td>SHIFT + CONTROL + N</td>
</tr>
<tr>
<td>AVG BW</td>
<td>SHIFT + CONTROL + A</td>
</tr>
<tr>
<td>CAL</td>
<td>SHIFT + CONTROL + C</td>
</tr>
<tr>
<td>MARKER</td>
<td>SHIFT + CONTROL + R</td>
</tr>
<tr>
<td>SEARCH</td>
<td>SHIFT + CONTROL + E</td>
</tr>
<tr>
<td><strong>Stimulus Keys</strong></td>
<td></td>
</tr>
<tr>
<td>FREQ</td>
<td>SHIFT + CONTROL + Q</td>
</tr>
<tr>
<td>POWER</td>
<td>SHIFT + CONTROL + P</td>
</tr>
<tr>
<td>SWEEP</td>
<td>SHIFT + CONTROL + W</td>
</tr>
<tr>
<td>TRIGGER</td>
<td>SHIFT + CONTROL + I</td>
</tr>
<tr>
<td><strong>Utility Keys</strong></td>
<td></td>
</tr>
<tr>
<td>SAVE RECALL &gt;Recall</td>
<td>SHIFT + CONTROL + L</td>
</tr>
<tr>
<td>SAVE RECALL &gt; Save</td>
<td>SHIFT + CONTROL + Y</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>SHIFT + CONTROL + Y</td>
</tr>
<tr>
<td>MACRO</td>
<td>SHIFT + CONTROL + G</td>
</tr>
<tr>
<td>SYSTEM &gt; Help</td>
<td>CONTROL + H</td>
</tr>
<tr>
<td>Undo</td>
<td>CONTROL + Z</td>
</tr>
<tr>
<td>Redo</td>
<td>CONTROL + Y</td>
</tr>
<tr>
<td>PRESET</td>
<td>SHIFT + CONTROL + X</td>
</tr>
<tr>
<td><strong>Other Features</strong></td>
<td></td>
</tr>
<tr>
<td>Trace 1</td>
<td>SHIFT + CONTROL + 1</td>
</tr>
<tr>
<td>Trace 2</td>
<td>SHIFT + CONTROL + 2</td>
</tr>
<tr>
<td>Trace 3</td>
<td>SHIFT + CONTROL + 3</td>
</tr>
<tr>
<td>Trace 4</td>
<td>SHIFT + CONTROL + 4</td>
</tr>
<tr>
<td>Hardkeys Toolbar</td>
<td>SHIFT + CONTROL + K</td>
</tr>
<tr>
<td>Minimize Application</td>
<td>SHIFT + CONTROL + Z</td>
</tr>
<tr>
<td>Mainframe Menu show/hide</td>
<td>SHIFT + CONTROL + B</td>
</tr>
<tr>
<td>File Open dialog</td>
<td>CONTROL + O</td>
</tr>
<tr>
<td>Save file</td>
<td>CONTROL + S</td>
</tr>
<tr>
<td>Feature</td>
<td>Keyboard Shortcuts</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>Save As dialog</td>
<td>CONTROL + A</td>
</tr>
<tr>
<td>Print dialog</td>
<td>CONTROL + P</td>
</tr>
<tr>
<td>Print to File dialog</td>
<td>CONTROL + T</td>
</tr>
<tr>
<td>Focus on Mainframe Menu</td>
<td>ALT</td>
</tr>
<tr>
<td>Softkey 1 to 8</td>
<td>CONTROL + 1, to CONTROL + 8</td>
</tr>
</tbody>
</table>
Using Help

This topic discusses the following:

- Documentation
- Printing Help
- Copying Help to your PC
- Launching Help
- Searching Help
- GUI Reference Help
  - Accessing GUI Reference Help with Help Hardkey
  - Accessing GUI Reference Help from Table of Contents
- Help Languages
- Documentation Warranty

See Also

- Programming Guide (for finding SCPI/COM commands for remote programming)
- Help, About Network Analyzer

Other Quick Start Topics

Help Documentation

This Help file, which is embedded in the analyzer, is the Users Guide and Programming Manual for the VNA. The help file is automatically updated on the VNA when firmware is updated. Only the VNA Installation and Quick Start Guide is shipped with new VNA instruments.

Hardcopy manuals are no longer available for purchase with the VNA.

All VNA documentation, including the latest online Web Help version of this Help file, and a printable .PDF version of the Help file, are available at http://na.support.keysight.com/pna/help/index.html.
Printing Help

A printable .PDF version of this Help file is available at http://na.support.keysight.com/pna/help/index.html.

Copying Help to your PC

With the Help system on your PC, you can read about the analyzer while away from it. You can also Copy and Paste programming code from this Help system directly into your programming environment.

The Help file is located on your analyzer hard-drive at C:\Program Files (x86)\Keysight\Network Analyzer\Help\<filename>.chm. If both the analyzer and PC are connected to LAN, you can map a drive and copy the file directly.

The Help file can also be downloaded from http://na.support.keysight.com/pna/help/index.html.

Launching Help

The Help system can be launched in the following ways:

1. From the front panel Help button.
2. From the Help drop-down menu.
3. From Dialog Box Help buttons.

Search Tab

TIP: To Search any topic for a keyword, press Ctrl and F.

The following rules apply for using full-text search:

- Searches are not case-sensitive.
- You can search for any combination of letters (a-z) and numbers (0-9).
- Punctuation marks (period, colon, semicolon, comma, and hyphen) are ignored during a search.
- You can group the words of your search using double quotes or parentheses. Examples: "response calibration" or (response calibration). This requirement makes it impossible to search for quotation marks.
Use Wildcard expressions:

- To search for one undefined character use a question mark (?). For example, searching for cal? will find calc and calf.

- To search for more than one undefined character use an asterisk (*). Searching for Cal* will find calibration and calculate.

Use Boolean operators to define a relationship between two or more search words.

<table>
<thead>
<tr>
<th>Search for</th>
<th>Example</th>
<th>Results will show topics containing:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two words in the same topic</td>
<td>response AND calibration</td>
<td>Both the words &quot;response&quot; and &quot;calibration&quot;.</td>
</tr>
<tr>
<td>Either of two words in a topic</td>
<td>response OR calibration</td>
<td>Either the word &quot;response&quot; or the word &quot;calibration&quot; or both.</td>
</tr>
<tr>
<td>The first word without the second word in a topic</td>
<td>response NOT calibration</td>
<td>The word &quot;response&quot; but not the word &quot;calibration&quot;.</td>
</tr>
<tr>
<td>Both words in the same topic, close together.</td>
<td>response NEAR calibration</td>
<td>The word &quot;response&quot; within eight words of the word &quot;calibration&quot;.</td>
</tr>
</tbody>
</table>

GUI Reference Help

Accessing GUI Reference Help with Help Hardkey

The GUI Reference help can be accessed at any time by clicking on the Help hardkey:
The GUI Reference help accessed by clicking on the Help hardkey corresponds to the currently active Measurement Class, currently selected Hardkey, and currently selected Soft Tab to provide context sensitive help.

The following procedure is a typical example of how to find GUI Reference help using the Help hardkey. This example assumes the Standard Measurement class is currently active, the Sweep hardkey is selected, and the Main Soft Tab is selected.
1. Click on the Help hardkey. The following is displayed:

[Image of a help window]

2. Click on a link for information. The links correspond to the Softkeys.

**Accessing GUI Reference Help from Table of Contents**

The GUI Reference topics displayed in the table of contents are arranged in alphabetical order:
The GUI Reference corresponds to the GUI Hardkeys to help find information quickly:
The following procedure is a typical example of how to find information using the GUI Reference. This example shows how to search for the Avg BW Hardkey information with the Standard Measurement class selected.

1. Under GUI Reference in the table of contents, select the Avg BW topic. The following is displayed:
The links shown on this page correspond to Measurement Class names because some menus change with Measurement Class.

2. Click on the Standard link. The following is displayed:
Main, Smoothing, and Delay Aperture correspond to the soft tab labels. The links correspond to the softkey labels.

3. Click on a link for information. The links correspond to the Softkeys.

Help Languages
This help file is offered in English ONLY.

**Glossary**

The Glossary holds definitions of words, in alphabetical order.

![Glossary Image]

**Documentation Warranty**

THE MATERIAL CONTAINED IN THIS DOCUMENT IS PROVIDED "AS IS," AND IS SUBJECT TO BEING CHANGED, WITHOUT NOTICE, IN FUTURE EDITIONS. FURTHER, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, KEYSIGHT DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED WITH REGARD TO THIS MANUAL AND ANY INFORMATION CONTAINED HEREIN, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. KEYSIGHT SHALL NOT BE LIABLE FOR ERRORS OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, USE, OR PERFORMANCE OF THIS DOCUMENT OR ANY INFORMATION CONTAINED HEREIN. SHOULD KEYSIGHT AND THE USER HAVE A SEPARATE WRITTEN AGREEMENT WITH WARRANTY TERMS COVERING THE MATERIAL IN THIS DOCUMENT THAT CONFLICT WITH THESE TERMS, THE WARRANTY TERMS IN THE SEPARATE AGREEMENT WILL CONTROL.
Click System > Help > About NA... to learn the capabilities of your analyzer.

- Model number
- See list of PNA models
- Frequency range
- Serial number
- Options (Learn how to install software options)
- Application Code (firmware) Version
• Solid State Drive Version

• System CPU Version - Learn more

• DSP (Digital Signal Processor.) Version. Contact Keysight to upgrade the DSP.

• Computer Name - Learn more. This is also reported on the LAN Status / LXI Compliance dialog.
Preset the Analyzer

When you Preset the analyzer, it is set to known, or preset conditions. You can use the factory default preset conditions, or define your own User Preset conditions.

Note: Presetting the analyzer will not remove the calibration data in the Channel default CALREG calset.

- Preset (Default) Conditions
- User Preset Conditions

See other ‘Setup Measurements’ topics

Preset Default Conditions

How to Preset the Analyzer

Using Hardkey/SoftTab/Softkey

1. Press Preset > Main > Preset.

When Confirm Preset is Off,

1. Press Preset.

Tip: Press the Preset button to start the VNA application if it is not already running.

User Preset Conditions

The analyzer can be preset to either factory default conditions or User Preset conditions.
How to set User Preset

Using Hardkey/SoftTab/Softkey

1. Press **Preset > Main > User Preset**.

User Preset dialog box help

With a User Preset saved and enabled, when the VNA is Preset, the User Preset settings are recalled instead of the factory default settings. Calibration data is NOT recalled with a User Preset. Learn more about instrument state settings.

**User Preset Enable**
Check - The VNA is preset to **User Preset** conditions when the Preset button is pressed.
Clear - The VNA is preset to **Default** conditions when the Preset button is pressed.

**Save current state as User Preset** Click to store the current instrument state as the User Preset conditions. File is stored as d:\users\public\documents\network analyzer\UserPreset.sta.
Note: If the "D:" drive is not found, the "C:" drive will be used.

**Load existing file as User Preset**  Click to retrieve an instrument state to be used as the User Preset conditions.
Measurement Classes

Measurement Classes are categories of measurements that can coexist on a channel.

- What are Measurement Classes
- How to assign a Measurement Class to a Channel
- Measurement Class Dialog Box Help

See other 'Setup Measurements' topics

What are Measurement Classes

**Note:** Measurement classes vary according to the VNA model and options installed.

The dialog below is an example showing the Measurement Classes currently available for a VNA. Within each of these classes there are a number of measurements.

Measurement Classes are categories of measurements that can coexist on a channel. A measurement from one class can NOT reside in a channel with a measurement from another class. For example, a Noise Figure measurement can NOT reside in a channel that is currently hosting Scalar Mixer Measurements.

The Measurement Class dialog is accessed in the following ways:

**How to assign a Measurement Class to a Channel**

**Using Hardkey/SoftTab/Softkey**

1. Press **Meas > S-Param > Meas Class...**

**Using a mouse**

1. Click **Instrument.**
2. Select **Meas Class...**

Measurement Class dialog box help
Measurement class dialog box shows the supported classes for your unit. The supported classes depends on the product and installed options. The above dialog box shows an example of PNA.

Measurements in a measurement class can NOT coexist in a channel with a measurement of a different measurement class.

Select a measurement class for the active channel or new measurement channel.

- The **Standard** measurement class contains S-Parameters. Balanced parameters, and Receiver measurements.
  
- All other measurement classes are commonly called "Applications".

**Title Bar** Indicates the active channel to which the measurement class will be assigned.

**Show setup dialog**

- Check to launch the selected Measurement Class dialog.
- Clear (default setting) to not launch the selected Measurement Class dialog. This setting survives a Preset and VNA Shutdown.

**Confirm changes**

- Check (default setting) to launch the Confirm Measurement Class Change dialog.
- Clear to perform the ‘OK’ actions without confirmation. This setting survives a Preset and VNA Shutdown.

**New Channel** Click to create the measurement class in a new channel and new window. A default measurement for that class is created in the channel.

To change the measurement, click **Trace**, then select a new measurement.
Choose to do the following:

- **OK** - Delete the existing measurements in the active channel. Create the new measurement class, and default measurement, in that channel.

- **Cancel** - Do not create the new measurement class. Leave the old measurements (and class) in that channel and return to the Measurement Class dialog box.
Measurement Parameters

This topic contains the following information:

- **S-Parameters** (pre-selected ratios)
- **Ratioed** (choose your own ratio)
- **Unratioed Power** (absolute power)
- **New / Change Measurement dialog box help**
- **How to Select a Measurement Parameter**
- **Option Sx090A/B Spectrum Analyzer Measurement Parameters** (separate topic)
- **Option S93070xB Modulation Distortion Measurement Parameters** (separate topic)
- **Option S93031xB Phase Noise Measurement Parameters** (separate topic)
- **Auxiliary**

Learn about Balanced Measurements

See other 'Setup Measurements' topics

---

S-Parameters

S-parameters (scattering parameters) are used to describe the way a device modifies a signal. For a 2-port device, there are **four S-Parameters**. The syntax for each parameter is described by the following:

\[
S_{out - in} \]

- **out** = analyzer port number where the device signal output is measured (receiver)
- **in** = analyzer port number where the signal is applied (incident) to the device (source)

**Move the mouse over each S-parameter to see the signal flow:**
For two-port devices:

- When the source goes into port 1, the measurement is said to be in the **forward** direction.
- When the source goes into port 2, the measurement is said to be in the **reverse** direction.

The analyzer automatically switches the source and receiver to make a forward or reverse measurement. Therefore, the analyzer can measure all four S-parameters for a two-port device with a single connection.

See the **block diagram** (including receivers) of your VNA.

**Common Measurements with S-Parameters**

<table>
<thead>
<tr>
<th>Reflection Measurements (S11 and S22)</th>
<th>Transmission Measurements (S21 and S12)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return loss</td>
<td>Insertion loss</td>
</tr>
<tr>
<td>Standing wave ratio (SWR)</td>
<td>Transmission coefficient</td>
</tr>
<tr>
<td>Reflection coefficient</td>
<td>Gain/Loss</td>
</tr>
<tr>
<td>Impedance</td>
<td>Group delay</td>
</tr>
<tr>
<td>S11, S22</td>
<td>Deviation from linear phase</td>
</tr>
<tr>
<td></td>
<td>Electrical delay</td>
</tr>
<tr>
<td></td>
<td>S21, S12</td>
</tr>
</tbody>
</table>
Receiver Measurements

All analyzer models have test port receivers and reference receivers. See the block diagram of your VNA.

For 4-port models...

- R1, R2, R3, and R4 are reference receivers. They measure the signal as it leaves the analyzer source.
  - R1 measures the signal out of Port 1
  - ...
  - R4 measures the signal out of Port 4
- A, B, C, and D are test port receivers. They measure the signal out (or reflecting off) of the DUT.
  - A measures the signal into VNA Port 1
  - B measures the signal into VNA Port 2
  - C measures the signal into VNA Port 3
  - D measures the signal into VNA Port 4

Models with more than 4 ports must specify receivers using Logical Receiver Notation. Learn more.

Ratioed Measurements

Ratioed measurements allow you to choose your own ratio of any two receivers that are available in your analyzer. S-parameters are actually predefined ratio measurements. For example S11 is A/R1.

The following are common uses of ratioed measurements:

- Comparing the phase between two paths of a device. An example could be something simple like a power splitter or more complicated like a dual-channel receiver.
- Measurements that require a higher dynamic range than the analyzer provides with S-parameters.

Your VNA MAY have front-panel jumper cables that go directly to measurement receivers. Learn about the front-panel jumpers on your VNA.

Unratioed (Absolute Power) Measurements

The unratioed power parameter measures the absolute power going into any of the receivers that are available on your analyzer.
The reference receivers are internally configured to measure the source power for a specific analyzer port. Performing an absolute power measurement of a reference receiver using a different source port will measure very little power unless the front panel jumpers are removed and signal is applied directly to the receiver. An example of this would be an R1 measurement using port 2 as the source.

- **Measuring phase** using a single receiver yields meaningless data. Phase measurements must be a comparison of two signals.

- Averaging for Unratioed parameters is computed differently from ratioed parameters. [Learn more.](#)

- To calibrate ratioed or unratioed receiver (power) parameters, the recommended method is the [Guided Power Calibration](#). The [Unguided Response Calibration](#) can also be used to calibrate a single unratioed or ratioed parameter at a time.

---

### New / Change Measurement dialog box help

**Note:** The only measurements that are available are those in the measurement class currently assigned to the active channel. Other measurements are NOT compatible.

To create a measurement other than these, first assign the appropriate measurement class to a new or existing channel. [Learn how.](#)

Click a tab to create or change measurements.

- When creating NEW measurements, you can choose more than one.

- When changing an EXISTING measurement, you can choose ONLY one.

### Tabs

**S-Parameter** Select a predefined ratioed measurements. [Learn more about S-parameters.](#)

For Setup:<= 4-Port

---
**Balanced**  Select a balanced measurement type.

**Topology**  Click to invoke the Balanced DUT Topology / Logical Port mappings dialog box. Learn more about Balanced Measurements.

For Setup: \( \leq 4\)-Port

**Select All**  Will only select the parameters shown and will not select the check box of the Receiver selector at the bottom.
**Receivers** Select receivers to make Ratioed and Unratioed (absolute power) measurements. [Learn more about receiver measurements.](#)

![Receiver Selection](image)

**Ratioed** Click on the check box to select the parameters and create measurement. Receiver selector at the bottom allow you to define ratios. Select a receiver for the Numerator, select another receiver for the Denominator, then select a source port for the measurement.

The **Source port** is ALWAYS interpreted as a logical port number.

For convenience, the table is populated with common choices.

- **Select All** Will only select the parameters shown and will not select the check box of the Receiver selector at the bottom.

  - [Learn about External Test Sets and Ratioed Measurements](#)
  - [Learn more about Ratioed Measurements.](#)

**Unratioed** Same as Ratioed, but select 1 as the Denominator.

  - [Learn More about Unratioed Measurements.](#)
  - [See the block diagram of receivers in YOUR VNA.](#)
  - The internal ADCs (Analog-Digital Converters) can be used as measurement receivers. [Learn more.](#)
Waves  Select receiver notation to make ratioed and unratioed measurements.

Click on the check box to select the parameters and create measurement. Wave selector at the bottom allow you to define ratio.

**Select All**  Will only select the parameters shown and will not select the check box of the Wave selector at the bottom.

**Receiver Notation**

Receivers can be also selected using logical receiver notation. This "8510-style" notation makes it easy to refer to multi-port receivers.

- **aN** - Reference receiver for logical port N
- **bN** - Test port receiver for logical port N

For example:

- For **Ratioed** measurements: "b12/a1" refers to the logical test port 12 receiver / the logical port 1 reference receiver.
- For **Unratioed** measurements: "b10" refers to the logical test port 10 receiver.

The VNA-style notation (A, B, R1 and so forth) can still be used to refer to **physical** receivers in less than 4 ports. [Learn more.](#)
However, ratioed measurements MUST use the same notation to refer to both receivers; either the physical receiver notation (A, R1) or the logical receiver notation (aN, bN). For example, the following mixed notation is NOT allowed: A/b3 and a5/R2.

**Programming**

When entering receiver letters using programming commands, neither logical or physical receiver notation are case sensitive.

**AUX**  Select input of Auxiliary on the rear panel to make DC measurement.

Click on the check box to select the input of auxiliary and create measurement. Auxiliary selector at the bottom allow you to define auxiliary and other parameters such as PMAR and DVMs.

**Select All**  Will only select the parameters shown and will not select the check box of the Auxiliary selector at the bottom.

**Channel / Window Selections**

These selections are NOT AVAILABLE when changing an EXISTING measurement. Learn how to
change a measurement.

**Channel Number**  Select the channel for the new traces.

**Create in New Window**

- Check to create new traces in a new window.
- Clear to create new traces in the active window. When the *traces per window limitation* has been reached, no more traces are added.

*About Measurement Parameters* (top of page)

---

**Balanced Source / Topology** dialog box help

**Topology Tab**

![Balanced Setup](image)

See the dialog for *Integrated True Mode Stimulus Application* (iTMSA).
Create or edit DUT Topology and Logical Port Mapping.

A Logical Port is a term used to describe a physical analyzer test port that has been remapped to a new port number. You can assign logical single-ended ports to logical balanced ports.

**Note:** These selections apply to ALL measurements in the channel. If the device topology is changed, any existing measurements in the channel that are incompatible with the new topology will be automatically changed to one that is compatible.

**Topology:** Describes your DUT as you would like it tested. The following device topologies can be measured by a multiport analyzer.

- **BAL** DUT has a single balanced port.
- **BAL-BAL** DUT has two balanced ports.
- **BAL-SE** DUT has one balanced port and one single-ended port.
- **BAL-SE-SE** DUT has one balanced port and two single-ended ports.
- **SE-BAL** DUT has one single-ended port and one balanced port.
- **SE-SE-BAL** DUT has two single-ended ports and one balanced port.

- **All SE**
  (Changes all entries to Single-ended)

- **All BAL**
  (Changes all entries to Balanced)

- **All Unused**
  (Changes all entries to Unused) This is a convenience feature to help set up custom topologies, but it is not possible to have only Unused ports. If the OK button is clicked and all ports are listed as Unused, the topology will be set up to have 1 SE Port.

- **Custom**
  (Maps the physical VNA ports to a device of balanced and single-ended logical ports for multi-port systems with greater than 4 ports.)

These topologies can be used in the reverse (⇐⇒) direction to measure:

- **Balanced / Single-ended** topology
- **Balanced / Single-ended - Single-ended** topology

For example, to measure a **Balanced / Single-ended** topology, measure the S12 (reverse direction) of a **Single-ended / Balanced** topology.

A balanced port can be any one of four physical port combinations:
Balanced Port  Number of rows equals number of VNA ports. User may select each port as SE or BAL or Unused. Unused ports are always forced to the bottom of the list and some Unused port selectors may be grayed-out when all the VNA ports are used.

VNA Port  Displays physical port numbers. But will display logical port numbers if logical ports are used. Balanced port requires (+) and (-) VNA port definitions; SE port only requires a single VNA port definition.

True Mode  Define true mode independently for each BAL port. SE ports cannot be defined as true mode. See this dialog for Integrated True Mode Stimulus Application (iTMSA).

Port Z Tab
Provide an enable for both Common and Differential Conversion and SE Port Z Conversion.

**Balanced Port** Shows all ports defined on the balanced topology page.

**Default Z** Shows the default impedances that will be applied if the port Z conversions are not enabled. The SE Default Z always equals the System Zo defined for the VNA. The Differential and Common Default Z will display values calculated from the single-ended port impedances.

**Converted Z** User may enter the real or imaginary component of the impedance.

**Power Waves** (default) This was legacy method used. If a load is the complex conjugate of the system impedance, then it will be displayed as a perfect match.

**Traveling Waves** This is newly added. If a load equals the complex system impedance, then it will be displayed as a perfect match.
Warning Dialog  The dialog is displayed if "Conversion" is enabled and "Apply Fixtures" is currently disabled.

See Also

- Learn about Logical Port mapping when using an External Test Set.
- Learn more about Balanced Measurements
- Balanced parameters can be saved to SNP files. Learn more.
Frequency Range

Frequency range is the span of frequencies you specify for making a device measurement.

- How to Set Frequency Range
- Zoom
- CW Frequencies
- Frequency Resolution
- Frequency Band Crossings
- Auto Tune (Option S9x09xxA/B, S9x090A/B only)
- Span All Meas Bands (Option S93070xB, S9x070A/B Modulation Distortion only)

See other ‘Setup Measurements’ topics

How to set Frequency Range

You can also make these settings and more from the Sweep Type dialog.

See the frequency ranges of all analyzer models.

Using Hardkey/SoftTab/Softkey

1. Press Freq> Main > Start, Stop, Center, or Span.
2. Enter desired frequency value.

Using a mouse

1. Right click on the X-axis label or stimulus range area under grid line.
2. Click on Start/Stop/Center....
Either of the following pairs of settings determine the frequency range. The last value that you enter determines the X-Axis labels. For example, if you enter the Start and Span values, the X-Axis will show Center and Span labels.

**Start /Stop** - Specifies the beginning and end frequency of the swept measurement range.

*Note:* The start frequency at preset is not the minimum value in full range.

**Center /Span** - Specifies the value at the center and frequency range.

Either of the following settings determine the number of evenly-spaced data points across the frequency range.

**Points** - Specifies the number of evenly-spaced data points across the frequency range. Learn more about Data Points.

**Step** - Available ONLY in Linear sweep type. Specifies the frequency step size between evenly-spaced data points. Changes to this setting will cause the Points setting to adjust to the closest integer. Any 'remainder' will adjust either the Stop value or Span value depending on which is displayed on the X-Axis label.

**Zoom**

Zoom allows you to easily change the start and stop frequencies or start and stop power levels in a power sweep.

Zoom operates on the Active Trace and all traces in the same channel as the active trace, regardless of the window in which they appear.
# How to Zoom in a measurement window

1. Left-click the mouse or use a finger, then drag across a portion of a trace.
2. Release the mouse or lift the finger and the following menu appears:
3. Select from the following:
   - **Zoom** - changes the channel stimulus settings to the left and right border values of the Zoom selection
   - **Zoom xy** - changes the channel stimulus settings as above. In addition, the Y-axis scale of the active trace changes to the approximate scale of the Zoom selection.
   - **Zoom Full Out** - changes the channel stimulus settings to the full span of the current calibration. If no calibration is ON, then the stimulus settings are changed to the full span of the VNA model.

## Notes

- The stimulus settings are changed for **ALL** traces in the active channel, regardless of the window in which they appear.
- If markers are in the selected area, they remain in place.
- If markers are in the unselected area, they are moved to the right or left edge of the new span. When Zoom Full Out is selected, the markers are moved back to their original location.

Zoom is NOT available for the following:

- Smith Chart or Polar display formats
- CW Time and Segment sweep type
- Frequency Offset Measurements
- FCA Opt S93083A/B Measurements

## CW Frequencies

Measurements with a **CW Time sweep** or **Power sweep** are made at a single frequency rather than over a range of frequencies.
How to set CW Frequency

Using Hardkey/SoftTab/Softkey

1. Set Sweep Type to CW Time or Power Sweep.
2. Press Freq > Main > CW.
3. Enter desired CW frequency.

Using a mouse

1. Set Sweep Type to CW Time or Power Sweep.
2. Right click on the stimulus range area under grid box.
3. Click on CW....

Frequency Resolution

The resolution for setting frequency is 1 Hz.

Frequency Band Crossings

The frequency range of the VNA covers several internal frequency bands. The higher the frequency range of the VNA, the larger the number of bands. The source power to your DUT turns off as the stimulus frequency is swept through these band crossings. To learn more, see Power ON and OFF during Sweep and Retrace.

The listed frequencies in the following tables are the stop frequency of the specified band, and the start frequency of the following band.

You can download a VNA Band Structure Readout utility that lists the band crossings for your VNA.

Frequency band crossings are different for the following models:
- PNA-X Models (with synthesizer revision greater than 6)
- PNA-X Models (with synthesizer revision 6 or earlier)
- N522xB Models (with synthesizer revision greater than 6)
- N522xB Models (with synthesizer revision 6 or earlier)
- N523xA Models

**PNA-X Models Band Stop Frequencies (with synthesizer revision greater than 6)**

**N5241B and N5242B**

<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>.396</td>
<td>24</td>
<td>12.00</td>
</tr>
<tr>
<td>1</td>
<td>.010</td>
<td>13</td>
<td>.500</td>
<td>25</td>
<td>12.80</td>
</tr>
<tr>
<td>2</td>
<td>.014</td>
<td>14</td>
<td>.628</td>
<td>26</td>
<td>13.51</td>
</tr>
<tr>
<td>3</td>
<td>.019</td>
<td>15</td>
<td>1.00</td>
<td>27</td>
<td>15.20</td>
</tr>
<tr>
<td>4</td>
<td>.027</td>
<td>16</td>
<td>1.50</td>
<td>28</td>
<td>16.00</td>
</tr>
<tr>
<td>5</td>
<td>.038</td>
<td>17</td>
<td>2.00</td>
<td>29</td>
<td>18.00</td>
</tr>
<tr>
<td>6</td>
<td>.053</td>
<td>18</td>
<td>3.20</td>
<td>30</td>
<td>19.04</td>
</tr>
<tr>
<td>7</td>
<td>.075</td>
<td>19</td>
<td>5.332</td>
<td>31</td>
<td>20.00</td>
</tr>
<tr>
<td>8</td>
<td>.105</td>
<td>20</td>
<td>6.00</td>
<td>32</td>
<td>21.30</td>
</tr>
<tr>
<td>9</td>
<td>.146</td>
<td>21</td>
<td>7.6</td>
<td>33</td>
<td>24.00</td>
</tr>
<tr>
<td>10</td>
<td>.205</td>
<td>22</td>
<td>8.5</td>
<td>34</td>
<td>26.50</td>
</tr>
<tr>
<td>11</td>
<td>.250</td>
<td>23</td>
<td>9.52</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**N5244B and N5245B**


<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>.396</td>
<td>24</td>
<td>12.00</td>
<td>36</td>
<td>30.40</td>
</tr>
<tr>
<td>1</td>
<td>.010</td>
<td>13</td>
<td>.500</td>
<td>25</td>
<td>12.80</td>
<td>37</td>
<td>32.00</td>
</tr>
<tr>
<td>2</td>
<td>.014</td>
<td>14</td>
<td>.628</td>
<td>26</td>
<td>13.51</td>
<td>38</td>
<td>36.10</td>
</tr>
<tr>
<td>3</td>
<td>.019</td>
<td>15</td>
<td>1.00</td>
<td>27</td>
<td>15.20</td>
<td>39</td>
<td>38.08</td>
</tr>
<tr>
<td>4</td>
<td>.027</td>
<td>16</td>
<td>1.50</td>
<td>28</td>
<td>16.00</td>
<td>40</td>
<td>40.53</td>
</tr>
<tr>
<td>5</td>
<td>.038</td>
<td>17</td>
<td>2.00</td>
<td>29</td>
<td>19.04</td>
<td>41</td>
<td>42.00</td>
</tr>
<tr>
<td>6</td>
<td>.053</td>
<td>18</td>
<td>3.20</td>
<td>30</td>
<td>20.00</td>
<td>42</td>
<td>43.50</td>
</tr>
<tr>
<td>7</td>
<td>.075</td>
<td>19</td>
<td>5.332</td>
<td>31</td>
<td>21.30</td>
<td>43</td>
<td>45.60</td>
</tr>
<tr>
<td>8</td>
<td>.105</td>
<td>20</td>
<td>6.00</td>
<td>32</td>
<td>24.00</td>
<td>44</td>
<td>48.00</td>
</tr>
<tr>
<td>9</td>
<td>.146</td>
<td>21</td>
<td>7.60</td>
<td>33</td>
<td>26.50</td>
<td>45</td>
<td>50.00</td>
</tr>
<tr>
<td>10</td>
<td>.205</td>
<td>22</td>
<td>8.50</td>
<td>34</td>
<td>27.02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>.250</td>
<td>23</td>
<td>9.52</td>
<td>35</td>
<td>28.50</td>
<td>47</td>
<td>57.30</td>
</tr>
</tbody>
</table>

**N5247B**

<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>.396</td>
<td>24</td>
<td>12.00</td>
<td>36</td>
<td>30.40</td>
</tr>
<tr>
<td>1</td>
<td>.010</td>
<td>13</td>
<td>.500</td>
<td>25</td>
<td>12.80</td>
<td>37</td>
<td>32.00</td>
</tr>
<tr>
<td>2</td>
<td>.014</td>
<td>14</td>
<td>.628</td>
<td>26</td>
<td>13.51</td>
<td>38</td>
<td>36.10</td>
</tr>
<tr>
<td>3</td>
<td>.019</td>
<td>15</td>
<td>1.00</td>
<td>27</td>
<td>15.20</td>
<td>39</td>
<td>38.08</td>
</tr>
<tr>
<td>4</td>
<td>.027</td>
<td>16</td>
<td>1.50</td>
<td>28</td>
<td>16.00</td>
<td>40</td>
<td>40.53</td>
</tr>
<tr>
<td>5</td>
<td>.038</td>
<td>17</td>
<td>2.00</td>
<td>29</td>
<td>19.04</td>
<td>41</td>
<td>42.00</td>
</tr>
<tr>
<td>6</td>
<td>.053</td>
<td>18</td>
<td>3.20</td>
<td>30</td>
<td>20.00</td>
<td>42</td>
<td>43.50</td>
</tr>
<tr>
<td>7</td>
<td>.075</td>
<td>19</td>
<td>5.332</td>
<td>31</td>
<td>21.30</td>
<td>43</td>
<td>45.60</td>
</tr>
<tr>
<td>8</td>
<td>.105</td>
<td>20</td>
<td>6.00</td>
<td>32</td>
<td>24.00</td>
<td>44</td>
<td>48.00</td>
</tr>
<tr>
<td>9</td>
<td>.146</td>
<td>21</td>
<td>7.60</td>
<td>33</td>
<td>26.50</td>
<td>45</td>
<td>50.00</td>
</tr>
<tr>
<td>10</td>
<td>.205</td>
<td>22</td>
<td>8.50</td>
<td>34</td>
<td>27.02</td>
<td>46</td>
<td>54.04</td>
</tr>
<tr>
<td>11</td>
<td>.250</td>
<td>23</td>
<td>9.52</td>
<td>35</td>
<td>28.50</td>
<td>47</td>
<td>57.30</td>
</tr>
</tbody>
</table>

**PNA-X Models Band Stop Frequencies (with synthesizer revision 6 or earlier)**

**N5241B** and **N5242B**
<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>.396</td>
<td>24</td>
<td>8.50</td>
<td>36</td>
<td>27.008</td>
</tr>
<tr>
<td>1</td>
<td>Reserved</td>
<td>13</td>
<td>.500</td>
<td>25</td>
<td>10.664</td>
<td>37</td>
<td>32.00</td>
</tr>
<tr>
<td>2</td>
<td>.014</td>
<td>14</td>
<td>.628</td>
<td>26</td>
<td>12.00</td>
<td>38</td>
<td>36.50</td>
</tr>
<tr>
<td>3</td>
<td>.019</td>
<td>15</td>
<td>1.00</td>
<td>27</td>
<td>12.80</td>
<td>39</td>
<td>40.50</td>
</tr>
<tr>
<td>4</td>
<td>.027</td>
<td>16</td>
<td>1.50</td>
<td>28</td>
<td>13.51</td>
<td>40</td>
<td>42.656</td>
</tr>
<tr>
<td>5</td>
<td>.038</td>
<td>17</td>
<td>2.00</td>
<td>29</td>
<td>15.40</td>
<td>41</td>
<td>43.50</td>
</tr>
<tr>
<td>6</td>
<td>.053</td>
<td>18</td>
<td>3.00</td>
<td>30</td>
<td>16.00</td>
<td>42</td>
<td>46.20</td>
</tr>
<tr>
<td>7</td>
<td>.075</td>
<td>19</td>
<td>3.20</td>
<td>31</td>
<td>19.00</td>
<td>43</td>
<td>48.00</td>
</tr>
<tr>
<td>8</td>
<td>.105</td>
<td>20</td>
<td>4.00</td>
<td>32</td>
<td>20.00</td>
<td>44</td>
<td>50.20</td>
</tr>
<tr>
<td>9</td>
<td>.146</td>
<td>21</td>
<td>5.332</td>
<td>33</td>
<td>21.328</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>.205</td>
<td>22</td>
<td>6.752</td>
<td>34</td>
<td>24.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>.250</td>
<td>23</td>
<td>8.00</td>
<td>35</td>
<td>26.50</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

N524B and N5245B

<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>.396</td>
<td>24</td>
<td>8.50</td>
<td>36</td>
<td>27.008</td>
</tr>
<tr>
<td>1</td>
<td>Reserved</td>
<td>13</td>
<td>.500</td>
<td>25</td>
<td>10.664</td>
<td>37</td>
<td>32.00</td>
</tr>
<tr>
<td>2</td>
<td>.014</td>
<td>14</td>
<td>.628</td>
<td>26</td>
<td>12.00</td>
<td>38</td>
<td>36.50</td>
</tr>
<tr>
<td>3</td>
<td>.019</td>
<td>15</td>
<td>1.00</td>
<td>27</td>
<td>12.80</td>
<td>39</td>
<td>40.50</td>
</tr>
<tr>
<td>4</td>
<td>.027</td>
<td>16</td>
<td>1.50</td>
<td>28</td>
<td>13.51</td>
<td>40</td>
<td>42.656</td>
</tr>
<tr>
<td>5</td>
<td>.038</td>
<td>17</td>
<td>2.00</td>
<td>29</td>
<td>15.40</td>
<td>41</td>
<td>43.50</td>
</tr>
<tr>
<td>6</td>
<td>.053</td>
<td>18</td>
<td>3.00</td>
<td>30</td>
<td>16.00</td>
<td>42</td>
<td>46.20</td>
</tr>
<tr>
<td>7</td>
<td>.075</td>
<td>19</td>
<td>3.20</td>
<td>31</td>
<td>19.00</td>
<td>43</td>
<td>48.00</td>
</tr>
<tr>
<td>8</td>
<td>.105</td>
<td>20</td>
<td>4.00</td>
<td>32</td>
<td>20.00</td>
<td>44</td>
<td>50.20</td>
</tr>
<tr>
<td>9</td>
<td>.146</td>
<td>21</td>
<td>5.332</td>
<td>33</td>
<td>21.328</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>.205</td>
<td>22</td>
<td>6.752</td>
<td>34</td>
<td>24.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>.250</td>
<td>23</td>
<td>8.00</td>
<td>35</td>
<td>26.50</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

N5247B
<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>6.75</td>
<td>24</td>
<td>32.00</td>
</tr>
<tr>
<td>1</td>
<td>Reserved</td>
<td>13</td>
<td>8.00</td>
<td>25</td>
<td>40.00</td>
</tr>
<tr>
<td>2</td>
<td>.053</td>
<td>14</td>
<td>10.70</td>
<td>26</td>
<td>40.50</td>
</tr>
<tr>
<td>3</td>
<td>.175</td>
<td>15</td>
<td>13.50</td>
<td>27</td>
<td>42.70</td>
</tr>
<tr>
<td>4</td>
<td>.250</td>
<td>16</td>
<td>15.40</td>
<td>28</td>
<td>46.20</td>
</tr>
<tr>
<td>5</td>
<td>.500</td>
<td>17</td>
<td>16.00</td>
<td>29</td>
<td>48.00</td>
</tr>
<tr>
<td>6</td>
<td>1.000</td>
<td>18</td>
<td>19.00</td>
<td>30</td>
<td>50.00</td>
</tr>
<tr>
<td>7</td>
<td>2.000</td>
<td>19</td>
<td>20.00</td>
<td>31</td>
<td>54.00</td>
</tr>
<tr>
<td>8</td>
<td>3.000</td>
<td>20</td>
<td>21.30</td>
<td>32</td>
<td>60.00</td>
</tr>
<tr>
<td>9</td>
<td>3.200</td>
<td>21</td>
<td>24.00</td>
<td>33</td>
<td>64.00</td>
</tr>
<tr>
<td>10</td>
<td>4.000</td>
<td>22</td>
<td>26.50</td>
<td>34</td>
<td>67.00</td>
</tr>
<tr>
<td>11</td>
<td>5.330</td>
<td>23</td>
<td>27.00</td>
<td>35</td>
<td>70.00</td>
</tr>
</tbody>
</table>

N522xB Models Band Stop Frequencies (with synthesizer revision greater than 6)

**N5221B and N5222B**

<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>15.20</td>
</tr>
<tr>
<td>1</td>
<td>.010</td>
<td>13</td>
<td>16.00</td>
</tr>
<tr>
<td>2</td>
<td>.053</td>
<td>14</td>
<td>18.00</td>
</tr>
<tr>
<td>3</td>
<td>.175</td>
<td>15</td>
<td>19.04</td>
</tr>
<tr>
<td>4</td>
<td>.250</td>
<td>16</td>
<td>20.00</td>
</tr>
<tr>
<td>5</td>
<td>.500</td>
<td>17</td>
<td>21.30</td>
</tr>
<tr>
<td>6</td>
<td>3.200</td>
<td>18</td>
<td>24.00</td>
</tr>
<tr>
<td>7</td>
<td>6.000</td>
<td>19</td>
<td>26.50</td>
</tr>
<tr>
<td>8</td>
<td>7.600</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9.52</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>12.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>13.51</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**N5224B and N5225B**
<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>15.20</td>
<td>24</td>
<td>38.08</td>
</tr>
<tr>
<td>1</td>
<td>.010</td>
<td>13</td>
<td>16.00</td>
<td>25</td>
<td>40.53</td>
</tr>
<tr>
<td>2</td>
<td>.053</td>
<td>14</td>
<td>19.04</td>
<td>26</td>
<td>42.00</td>
</tr>
<tr>
<td>3</td>
<td>.175</td>
<td>15</td>
<td>20.00</td>
<td>27</td>
<td>43.50</td>
</tr>
<tr>
<td>4</td>
<td>.250</td>
<td>16</td>
<td>21.30</td>
<td>28</td>
<td>45.60</td>
</tr>
<tr>
<td>5</td>
<td>.500</td>
<td>17</td>
<td>24.00</td>
<td>29</td>
<td>48.00</td>
</tr>
<tr>
<td>6</td>
<td>3.20</td>
<td>18</td>
<td>26.50</td>
<td>30</td>
<td>50.00</td>
</tr>
<tr>
<td>7</td>
<td>6.00</td>
<td>19</td>
<td>27.02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>7.60</td>
<td>20</td>
<td>28.50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9.52</td>
<td>21</td>
<td>30.40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>12.00</td>
<td>22</td>
<td>32.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>13.51</td>
<td>23</td>
<td>36.10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**N5227B**

<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>15.20</td>
<td>24</td>
<td>38.08</td>
</tr>
<tr>
<td>1</td>
<td>.010</td>
<td>13</td>
<td>16.00</td>
<td>25</td>
<td>40.00</td>
</tr>
<tr>
<td>2</td>
<td>.053</td>
<td>14</td>
<td>19.04</td>
<td>26</td>
<td>40.53</td>
</tr>
<tr>
<td>3</td>
<td>.175</td>
<td>15</td>
<td>20.00</td>
<td>27</td>
<td>45.82</td>
</tr>
<tr>
<td>4</td>
<td>.250</td>
<td>16</td>
<td>21.30</td>
<td>28</td>
<td>48.00</td>
</tr>
<tr>
<td>5</td>
<td>.500</td>
<td>17</td>
<td>24.00</td>
<td>29</td>
<td>50.00</td>
</tr>
<tr>
<td>6</td>
<td>3.20</td>
<td>18</td>
<td>26.50</td>
<td>30</td>
<td>53.00</td>
</tr>
<tr>
<td>7</td>
<td>6.00</td>
<td>19</td>
<td>27.02</td>
<td>31</td>
<td>54.04</td>
</tr>
<tr>
<td>8</td>
<td>7.60</td>
<td>20</td>
<td>28.50</td>
<td>32</td>
<td>57.30</td>
</tr>
<tr>
<td>9</td>
<td>9.52</td>
<td>21</td>
<td>30.40</td>
<td>33</td>
<td>60.10</td>
</tr>
<tr>
<td>10</td>
<td>12.00</td>
<td>22</td>
<td>32.00</td>
<td>34</td>
<td>60.80</td>
</tr>
<tr>
<td>11</td>
<td>13.51</td>
<td>23</td>
<td>36.10</td>
<td>35</td>
<td>64.00</td>
</tr>
</tbody>
</table>

**N522xB Models Band Stop Frequencies (with synthesizer revision 6 or earlier)**

**N5221B and N5222B**
<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>6.752</td>
</tr>
<tr>
<td>1</td>
<td>.010</td>
<td>13</td>
<td>8.0</td>
</tr>
<tr>
<td>2</td>
<td>.053</td>
<td>14</td>
<td>10.664</td>
</tr>
<tr>
<td>3</td>
<td>.175</td>
<td>15</td>
<td>13.51</td>
</tr>
<tr>
<td>4</td>
<td>.250</td>
<td>16</td>
<td>15.40</td>
</tr>
<tr>
<td>5</td>
<td>.500</td>
<td>17</td>
<td>16.0</td>
</tr>
<tr>
<td>6</td>
<td>1.0</td>
<td>18</td>
<td>20.0</td>
</tr>
<tr>
<td>7</td>
<td>2.0</td>
<td>19</td>
<td>21.328</td>
</tr>
<tr>
<td>8</td>
<td>3.0</td>
<td>20</td>
<td>24.0</td>
</tr>
<tr>
<td>9</td>
<td>3.2</td>
<td>21</td>
<td>27.0</td>
</tr>
<tr>
<td>10</td>
<td>4.0</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>5.332</td>
<td>23</td>
<td></td>
</tr>
</tbody>
</table>

N5224B and N5225B

<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>6.752</td>
</tr>
<tr>
<td>1</td>
<td>Reserved</td>
<td>13</td>
<td>8.0</td>
</tr>
<tr>
<td>2</td>
<td>.053</td>
<td>14</td>
<td>10.664</td>
</tr>
<tr>
<td>3</td>
<td>.175</td>
<td>15</td>
<td>13.51</td>
</tr>
<tr>
<td>4</td>
<td>.250</td>
<td>16</td>
<td>15.40</td>
</tr>
<tr>
<td>5</td>
<td>.500</td>
<td>17</td>
<td>16.0</td>
</tr>
<tr>
<td>6</td>
<td>1.000</td>
<td>18</td>
<td>19.00</td>
</tr>
<tr>
<td>7</td>
<td>2.000</td>
<td>19</td>
<td>20.0</td>
</tr>
<tr>
<td>8</td>
<td>3.000</td>
<td>20</td>
<td>21.328</td>
</tr>
<tr>
<td>9</td>
<td>3.200</td>
<td>21</td>
<td>24.0</td>
</tr>
<tr>
<td>10</td>
<td>4.000</td>
<td>22</td>
<td>26.50</td>
</tr>
<tr>
<td>11</td>
<td>5.332</td>
<td>23</td>
<td>27.008</td>
</tr>
</tbody>
</table>

N5227B
<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>6.752</td>
<td>24</td>
<td>32.00</td>
</tr>
<tr>
<td>1</td>
<td>.010</td>
<td>13</td>
<td>8.00</td>
<td>25</td>
<td>40.00</td>
</tr>
<tr>
<td>2</td>
<td>.053</td>
<td>14</td>
<td>10.664</td>
<td>26</td>
<td>40.50</td>
</tr>
<tr>
<td>3</td>
<td>.175</td>
<td>15</td>
<td>13.51</td>
<td>27</td>
<td>42.656</td>
</tr>
<tr>
<td>4</td>
<td>.250</td>
<td>16</td>
<td>15.40</td>
<td>28</td>
<td>46.20</td>
</tr>
<tr>
<td>5</td>
<td>.500</td>
<td>17</td>
<td>16.00</td>
<td>29</td>
<td>48.00</td>
</tr>
<tr>
<td>6</td>
<td>1.000</td>
<td>18</td>
<td>19.00</td>
<td>30</td>
<td>50.00</td>
</tr>
<tr>
<td>7</td>
<td>2.000</td>
<td>19</td>
<td>20.00</td>
<td>31</td>
<td>54.00</td>
</tr>
<tr>
<td>8</td>
<td>3.000</td>
<td>20</td>
<td>21.328</td>
<td>32</td>
<td>60.00</td>
</tr>
<tr>
<td>9</td>
<td>3.200</td>
<td>21</td>
<td>24.00</td>
<td>33</td>
<td>64.00</td>
</tr>
<tr>
<td>10</td>
<td>4.000</td>
<td>22</td>
<td>26.50</td>
<td>34</td>
<td>67.00</td>
</tr>
<tr>
<td>11</td>
<td>5.332</td>
<td>23</td>
<td>27.008</td>
<td>35</td>
<td>70.00</td>
</tr>
</tbody>
</table>

N523x Models  See N523x Freq ranges.

N5231B, N5232B, N5239B

<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>10</td>
<td>5.33</td>
</tr>
<tr>
<td>1</td>
<td>.000425</td>
<td>11</td>
<td>6.75</td>
</tr>
<tr>
<td>2</td>
<td>.0012</td>
<td>12</td>
<td>8.00</td>
</tr>
<tr>
<td>3</td>
<td>.0075</td>
<td>13</td>
<td>10.07</td>
</tr>
<tr>
<td>4</td>
<td>.025</td>
<td>14</td>
<td>13.50</td>
</tr>
<tr>
<td>5</td>
<td>.050</td>
<td>15</td>
<td>16.00</td>
</tr>
<tr>
<td>6</td>
<td>1.00</td>
<td>16</td>
<td>16.70</td>
</tr>
<tr>
<td>7</td>
<td>2.00</td>
<td>17</td>
<td>20.01</td>
</tr>
<tr>
<td>8</td>
<td>3.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>4.00</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

N5234B, N5235B Models
<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.010</td>
<td>11</td>
<td>10.66</td>
<td>21</td>
<td>30.80</td>
</tr>
<tr>
<td>1</td>
<td>13.51</td>
<td>12</td>
<td>13.51</td>
<td>22</td>
<td>31.50</td>
</tr>
<tr>
<td>2</td>
<td>15.40</td>
<td>13</td>
<td>15.40</td>
<td>23</td>
<td>32.00</td>
</tr>
<tr>
<td>3</td>
<td>16.00</td>
<td>14</td>
<td>16.00</td>
<td>24</td>
<td>40.00</td>
</tr>
<tr>
<td>4</td>
<td>20.00</td>
<td>15</td>
<td>20.00</td>
<td>25</td>
<td>40.56</td>
</tr>
<tr>
<td>5</td>
<td>20.26</td>
<td>16</td>
<td>20.26</td>
<td>26</td>
<td>42.00</td>
</tr>
<tr>
<td>6</td>
<td>21.33</td>
<td>17</td>
<td>21.33</td>
<td>27</td>
<td>48.00</td>
</tr>
<tr>
<td>7</td>
<td>24.00</td>
<td>18</td>
<td>24.00</td>
<td>28</td>
<td>50.00</td>
</tr>
<tr>
<td>8</td>
<td>26.00</td>
<td>19</td>
<td>26.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>27.01</td>
<td>20</td>
<td>27.01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>8.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Auto Tune**

The **Auto Tune** softkey is a feature of the Spectrum Analyzer measurement class (Option S9x09xxA/B, S9x090A/B). Each time this softkey is pressed, all defined distortion measurement bands for the modulation channel are automatically tuned.

**Span All Meas Bands**

The **Span All Meas Bands** softkey is a feature of the Modulation Distortion measurement class (Option S93070xB, S9x070A/B only). Each time this softkey is pressed, all defined distortion measurement bands for the modulation channel are automatically tuned.
Power Level

Power level is the power of the source at the test ports.

- How to make Power Settings
- Power Dialog
- Power and Attenuator Dialog
- Source Unleveled
- Setting Independent Port Power
- Optimum Attenuation Value
- Receiver Attenuation
- Power ON and OFF during Save / Recall and Preset
- Power ON and OFF during Sweep and Retrace

See other ‘Setup Measurements’ topics

Power Settings

The test port output power is specified over frequency.

See the Power Range specifications for your analyzer.
How to make Power settings

Use one of the following methods to set port power.

Using Hardkey/SoftTab/Softkey

1. Press **Power** > **Main** > **Power Level / Start Power / Stop Power** to enter desired power level.
2. Press **Power** > **Main** > **RF Power** to turn ON or OFF the RF power.
3. Press **Power** > **Leveling & Offsets** > click left side **Slope** small button to turn ON or OFF the slope (Green color means the slope is turned ON; Grey color means the slope is turned OFF).

Using a mouse

1. Right click on the X-axis label or stimulus range area under grid line.
2. Click **Power....** and then **Power: Channel N** dialog box appears.

**Power** dialog box help

![Power dialog box](image)

This dialog provides basic control of source power for a specific port.

See [Power and Attenuators dialog box](#).

**Power On (All Channels)** Check to enable source power for all channels. Only turns power ON if channel power setting is ON or Auto.

**Port 'n’** Active source port for which power is being set.
**Port Power**  Sets the power level for the specified port.

To accurately set the power level at any point after the test port, perform a Source Power Calibration.

**Power Sweep**

**Start / Stop Power**  Set the start and stop power values of a power sweep.

- These settings are only available when Sweep Type is set to Power Sweep.
- Uncoupled power sweep power can be set from the Advanced Power dialog.
- You can Zoom to easily change the start and stop power levels in a power sweep. Learn how.
- Learn more about Power Sweep.

**Power Slope**

Helps compensate for cable and test fixture power losses at increased frequency.

**Slope**  Select to set the power slope. Clear to set power slope OFF. Learn more about power slope.

---

**How to make Power settings**

Use one of the following methods to set port power.

**Using Hardkey/SoftTab/Softkey**

1. Press **Power > Main > Power and Attenuators** and then the **Power and Attenuators** dialog box appears.

**Detailed settings for Power and Attenuators:**

1. Press **Power > Main > RF Power** to turn ON or OFF the RF power.

2. Press **Power > Port Power > Select Port x** to active the selected port.

3. Press **Power > Port Power > Power Level / Start Power / Stop Power** to enter desired power level for selected port.

4. Press **Power > Port Power > Source State** to choose the source state either Auto, On or Off.

5. Press **Power > Port Power > Coupling** to turn ON or OFF Power Coupling.

6. Press **Power > Leveling & Offsets** > click left side Slope small button to turn ON or OFF the slope (Green color means the slope is turned ON; Grey color means the slope is turned OFF).
**Power and Attenuators** dialog box help

Defines and controls the source power and attenuation for the active channel.

**Note:** External sources can be controlled from this dialog. Learn more.

**Power On (All Channels)** Check to enable source power for all channels (same function as **RF Power**). Only turns power ON if channel power setting is ON or Auto.

**Port Powers Coupled**

- **Coupled** (checked) The power levels are the same at each test port. Set power at any test port and all test ports change to the same power level.

- **Uncoupled** (cleared) The power levels are set independently for each test port. Uncouple power, for example, if you want to measure the gain and reverse-isolation of a high-gain amplifier. The power required for the input port of the amplifier is much lower than the power required for the output port. A power sweep can also be performed with uncoupled power. Learn more about Setting Independent Port Power

**Name** Lists the analyzer test ports.

**State**

- **Auto** Source power is turned ON at the specified test port when required by the measurement. This is the most common (default) setting. See also Power ON and OFF during Save / Recall, User Preset, and Preset.

- **ON** Source power is ALWAYS ON, regardless of measurements that are in process. Use this setting to
supply source power to a DUT port that always requires power, such as an LO port. This could turn OFF power at another test port. Learn about internal second source restrictions. See also Setting Port Power On/Off for the Second source and Multi Module/Unit configuration.

- **OFF** Source power is never ON, regardless of the measurement requirements. Use this setting to prevent damage to a sensitive DUT test port.

**Port Power** Sets the power level at the output of the source.

- To accurately set the power level at any point after the test port, perform a Source Power Calibration.
- See specified power range of VNA model.
- See ECal Module Compression Level

**Start / Stop Power** Available ONLY when sweep type is set to Power Sweep. Set the start and stop power values of a power sweep. Learn how to set Power Sweep.

- You can specify whether to maintain source power at either the start power or stop power level at the end of a power sweep. Learn more.
- A power sweep can be performed with uncoupled power. Different power ranges can be swept in the forward and reverse directions.

**Auto Range** Check to allow the analyzer to select the optimum attenuation value to achieve the specified test port power.

Clear to manually set the attenuation for each port. Type or select the attenuation value in the adjacent Attenuator Control box.

When using manual attenuation (Auto Range cleared), Port Power can be set within a 60 dB range. For example:

- With 0 dB of manual attenuation, Port Power can be set from -30 dBm to +30 dBm.
- With 10 dB of manual attenuation, Port Power can be set from -40 dBm to +20 dBm, and so forth.

**Important Note:** The available power range can also be adjusted AUTOMATICALLY by a Source Power Calibration, Guided Power Cal, or Power Compensation. If you are NOT seeing the range that you expect, or the correct power level at your DUT, view the Power Offset column in the Power Limits and Offsets dialog.

**Source Attenuator** When Port Powers are Uncoupled, manual attenuator control allows you to
set a wide range of power levels by setting the attenuation. See Setting Independent Port Power. Also use manual attenuation control when a measurement requires a very good impedance match with the source, such as with oscillators or conditionally unstable amplifiers. Choose an attenuation level of 10 dB or more to ensure the best source match.

- Attenuators or other mechanical switches are NOT allowed to switch continuously. Learn more.

- When Port Powers are Coupled, changing one port Attenuation Control value changes all port values.

**Note:** Because the reference receiver is not in the attenuation path, there is more power at the reference receiver than at the test port by the amount of source attenuation.

By default, ALL VNA models mathematically offset the reported power at the reference receivers by the amount of source attenuation. See Block diagram.

A preference can be set to NOT mathematically offset the reported power of the reference receiver by the amount of source attenuation.

Learn how to set the preference.

**Leveling Mode (ALC Hardware Softkey)**- Refer to the following diagram:

- **Internal** - ALC leveling. Power level within an attenuator setting is limited to the ALC Range. See Source Unleveled.

- **Open Loop** - No ALC and NO Receiver Leveling. (Used during pulse conditions with the internal source modulators). NOT available on N523x models. No leveling is used in setting the source power. The lowest settable power, without attenuation, is limited to -30dBm. The source power level accuracy is very compromised. Use a source power calibration to make the source power somewhat more accurate.

- **Receiver Rx** - Receiver Leveling. Select a receiver to use for leveling the source. Learn more.
Note: Receiver Leveling can be used with EITHER Internal ALC or Open Loop. See Enable ALC Hardware on the Receiver Leveling dialog.

Channel Power Slope

Helps compensate for cable and test fixture power losses at increased frequency. With power slope enabled, the port output power increases (enter positive value) or decreases (enter negative value) as the sweep frequency increases.

Slope  Select to set the power slope. Clear to set power slope OFF.

Power slope is computed and applied from 0 GHz – not from the measurement start frequency.

For example, with the following measurement settings:

- Start / Stop Freq: 10 GHz to 20 GHz
- Power level: 0 dBm
- Slope: 1 dB/GHz

The power into the DUT from 10 GHz to 20 GHz is 10 dBm sloping to 20 dBm

Offset and Limits  Launches the Power Offset and Limits dialog.

Receiver Leveling  Launches the Receiver Leveling dialog.

Receiver Attenuator  Launches the Receiver Attenuator dialog.

Path Configurator  Launches the Path Configurator dialog.

Source Unleveled

When the power level that is required at a test port is higher than can be supplied, a Source Unleveled error message appears on the screen and the letters LVL appear on the status bar.

To perform a power sweep, the range of power is usually limited to the range of the Automatic Leveling Control (ALC) loop. (The PNA-X allows a very wide power range using Open Loop).

Specifications guarantee the ALC power range over which power can be supplied without an unleveled indication. However, the actual achievable power range on your analyzer is probably greater than the specified range.

How to calculate the specified achievable power range
From the specifications for a frequency span from 15 GHz to 20 GHz:

- Max Leveled Power = -8 dBm
- Power Sweep Range (ALC) = -17 dB

For this frequency range the specified power range is calculated as:

- Max = -8 dBm
- Min = (-8)-(17) = -25 dBm

When using Source Attenuators:

- with 10dB of attenuation, this becomes -18 dBm to -35 dBm
- with 20dB of attenuation, this becomes -28 dBm to -45 dBm, and so forth.

To resolve an unleveled condition, change either the Test Port Power or Attenuator setting.

**Important Note:** The available power range can also be adjusted AUTOMATICALLY by a Source Power Calibration, Guided Power Cal, or Power Compensation. If you are NOT seeing the range that you expect, or the correct power level at your DUT, view the Power Offset column in the Power Limits and Offsets dialog.

Setting Independent Port Power

You can uncouple port power and specify different power levels at each test port. There are a few things to consider when setting independent port powers.

- Does your required high and low power levels fall within the specified Min and Max power range of the analyzer? See Unleveled Indicator. If they do not, you may need to use the internal Source Attenuators.

- Does the analyzer have source attenuators? If so, how many source attenuators? Some VNA models have one attenuator for each port. In most multiport VNA systems, the attenuators are shared by at least two test ports. See VNA Options to see the availability and range of source attenuation on your VNA.
Note: To prevent premature wear, attenuators or other mechanical switches are NOT allowed to switch continuously.

These mechanical devices are set for the entire channel. When more than one channel is used, and a mechanical device setting is NOT the same for all channels, only the ACTIVE channel is allowed to sweep. All other channels are NOT allowed to sweep (Blocked).

Learn how to view the settings of all mechanical devices.

Setting Port Power On/Off for the Second source and Multi Module/Unit configuration

When the unit has the second source capability or multi module/unit configuration in PXI/USB VNA, Source power ALWAYS ON can be applied for each source group.

The following dialog shows an example of three module configuration of six, four and two port PXI VNAs. In this case, there are three independent source group, port 1 to 6, 7 to 10 and 11 to 12. You can set one port in each group at ON state.

Optimum Attenuation Value

The attenuator has different positions, allowing a wide range of power levels. The number of power ranges available is determined by the source attenuation installed in your VNA. See VNA Options to see the availability and range of source attenuation on your VNA.

- Each range has a total specified span (50 dB in the following Attenuation Values graphic).
- The optimum setting is the middle of the range. This range provides the best accuracy and performance of the source leveling system. The optimum ranges are the blue regions in the following graphic.
- An attenuator setting can be selected manually or automatically. If automatic is selected, the blue optimum ranges (shown in the following graphic) are used.
**Note:** Error correction is fully accurate only for the power level at which a measurement calibration was performed. However, when changing power within the same attenuator range at which the measurement calibration was performed, ratioed measurements can be made with nearly full accuracy (non-ratioed measurements with less accuracy).

**Receiver Attenuators** dialog box help

Receiver Attenuators are offered as an option. Learn more.

Type or select independent attenuation values for each receiver.
Receiver A is at Test Port 1
Receiver B is at Test Port 2
Receiver C is at Test Port 3
Receiver D is at Test Port 4

Receiver Attenuation is used to protect the VNA test port receivers from damage or compression. Receiver attenuation causes the reported power at the receiver to be less than the power at the test port by the specified amount of attenuation.

When an external test set is connected, Receiver Attenuation control is only available for the physical receivers in the VNA. Switching receiver attenuation using logical receiver notation is NOT allowed.

Note: A preference can be set to mathematically offset (or NOT) the reported power at the test port receivers by the amount of receiver attenuation. By default, All VNA models offset the display.

Learn how to set the preference.

CAUTION! You can damage the analyzer receivers if the power levels exceed the maximum values.

- See Technical Specifications for the maximum input power to a receiver and receiver compression.
- See Receiver attenuation values for your VNA model.

Power ON and OFF during Save / Recall, User Preset, and Preset

To protect your DUT from being inadvertently powered ON, the following RF Power ON/OFF settings occur:

**Instrument State Save/Recall**

If power is OFF when an instrument state is saved, then power will always be OFF after the instrument state is recalled.

If power is ON when an instrument state is saved, and the current power setting is OFF, then power will be OFF after the instrument state is recalled.
User Preset

If power is OFF when a User Preset is saved, then power will always be OFF after a User Preset.

If power is ON when a User Preset is saved, and the current power setting is OFF, then power will be OFF after a User Preset.

Preset

Instrument Preset sets power ON by default.

This can be changed with a Preference setting so that, if the current power setting is OFF, then power will be OFF after Preset.

Power ON and OFF during Sweep and Retrace

The frequency range of the VNA covers several internal frequency bands. The higher the frequency range of the VNA, the larger the number of bands. For example, a 9 GHz VNA has 6 frequency bands, a 50 GHz VNA has 25 frequency bands. See the frequency band crossings.

Power to the DUT is turned OFF during band changes to avoid causing power spikes to the DUT.

Retrace occurs when the source gets to the end of your selected frequency span and moves back to the start frequency. Power to the DUT is again turned OFF when retracing across frequency bands.

The following power ON/OFF states occur for various stimulus settings:

1. **Single band sweep** - The power ON/OFF state to the DUT during retrace is determined by a preference setting.

2. **Multi-band sweep** - The power to the DUT is turned OFF while sweeping across a band crossing. It is turned OFF again during retrace.

3. **Power sweep** - Power sweep is always done at a single frequency, Therefore, like item 1, the power ON/OFF state to the DUT during retrace is determined by a preference setting

4. **Single sweep**:

   - Manual trigger mode - At the end of a multiband sweep, power is turned OFF during retrace, and then power is turned back ON before arming for the next trigger.

   - Hold mode - Power can be ON or OFF depending on when and how Hold mode is entered. However, power can be immediately turned OFF manually or remotely.

**Caution**: Avoid expensive repairs to your analyzer. Read Electrostatic Discharge Protection.
Receiver Leveling

**Note:** The M937xA/P937xA does not support this function.

Receiver Leveling adjusts the source power until the measured receiver power is equal to the Port Power.

In this topic:

- Overview
- Receiver Leveling Process
- Features and Limitations
- How to make Receiver Leveling settings
  - Support for Pre-Sweep Mode, Point Mode, and Prior Sweep Mode
  - Receiver Leveling dialog box help
  - Initial Power Selection

**See other 'Setup Measurements' topics**

**Overview**

Receiver Leveling uses receiver measurements to adjust the source power level across a frequency or power sweep. There are three receiver leveling modes: pre-sweep mode, point mode, and prior sweep mode.

**Note:** Enabling Safe Mode when using receiver leveling may be necessary to ensure stable results.

**Receiver Leveling Process**

**Pre-Sweep Mode**

Before each measurement sweep, a variable number of background sweeps are performed to measure and adjust power at the target port. Those power adjustments are then used to achieve greater source power level accuracy on the final sweep.
This is similar to a Source Power Calibration but because Receiver Leveling is updated for every measurement sweep, it provides more accurate source power levels, but also takes longer to perform each measurement sweep.

1. For each leveling pre-sweep, port power is measured by the specified receiver. Learn how the initial power level is selected.
2. The deviation is calculated between the measured power and the port power setting.
3. The deviation is applied as a power offset on the final sweep.
4. This pre-sweep process continues until the receiver power at each data point has achieved the port power within the specified tolerance value, or until the specified number of leveling sweeps (iterations) has been reached.

**Point Mode (if supported by the hardware)**

This mode does not use a pre-sweep. Instead, leveling is performed at the same time each point is acquired.

1. When the sweep starts, the source is set to the first point and power is read.
2. If the power is outside the target power tolerance, the source power is adjusted and the point is read again.
3. If max iterations is reached or the target power is within the tolerance, display the point and move to the next point.
4. This process repeats for each point.

**Prior Sweep Mode**

This mode avoids any extra data acquisition. There is no pre-sweep or extra point acquisitions.

This mode uses the data acquired during the prior sweep to compute a power offset for the following sweep. The power continues to be adjusted in this way until the tolerance is met.

1. Acquire data from the prior sweep.
2. The deviation is calculated between the measured power and the port power.
3. The deviation is applied as a power offset on the final sweep.
4. This process repeats for each successive sweep until the tolerance is reached.

**Features and Limitations**
Receiver Leveling can be used with most sweep types, including Segment sweep and Power sweep. See Wide Power Sweep with Receiver Leveling.

Receiver Leveling is ALWAYS enabled for the controlled source when Phase Control (Opt S93088A/B) is enabled.

Receiver Leveling is available for standard S-parameter measurements and with FCA, GCA, and IMD applications.

Turn ON Receiver Leveling before or after doing a Calibration. When turned ON before calibrating, it is turned OFF during the calibration, then back ON after calibration.

Power Offset on the Offsets and Limits dialog can be used when there exists an additional attenuator or booster amplifier in the source path. An offset should be set to improve the leveling speed. This power offset is automatically used to set the port power.

Use Receiver Leveling for the following:

- Correcting for short term drift when using an external component, such as a booster amplifier. The booster amplifier must be connected to the front-panel jumpers, in front of the reference receiver. See the Block diagram for your VNA, located at the end of every Specifications document.
- Extending the accuracy of power leveling at very low powers where the internal detector may be too noisy.
- Providing controlled power during Pulsed measurements in an open loop mode.
- Controlling the power at the outputs of MM-Wave heads.

**Note:** Increase the sweep delay if output power is not accurate when Receiver Leveling is applied to two or more ports.

### How to make Receiver Leveling settings

Start the Power and Attenuators dialog box as follows:

<table>
<thead>
<tr>
<th><strong>Using Hardkey/SoftTab/Softkey</strong></th>
<th><strong>Using a mouse</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Power &gt; Main &gt; Power and Attenuators</strong>...</td>
<td></td>
</tr>
<tr>
<td>2. On the Power and Attenuators dialog, click <strong>Receiver Leveling</strong> button.</td>
<td></td>
</tr>
<tr>
<td>1. Click <strong>Stimulus</strong>, <strong>Power</strong>, then <strong>Power and Attenuators</strong>.</td>
<td></td>
</tr>
<tr>
<td>2. On the Power and Attenuators dialog, click <strong>Receiver Leveling</strong> button.</td>
<td></td>
</tr>
</tbody>
</table>
**Receiver Leveling** dialog box help

Learn about Receiver Leveling (scroll up).

**Support for Pre-Sweep Mode, Point Mode, and Prior Sweep Mode**

The following dialog and descriptions apply to VNAs that support pre-sweep mode, point mode, and prior sweep mode receiver leveling.

![Receiver Leveling Setup: Channel 1](image-url)

**Controlled Source (Port)**

Each source port to be leveled is configured individually. Select a source to be configured for receiver leveling. Choose from: Port 1, Port 2, Port 3, Port 4, or any active external source. Learn more about External Devices.
**Note:** A Modulation Distortion channel only allows one modulated source. Therefore, only one row will be displayed showing the name of the modulated source.

**Enable Leveling** - Checkbox enables/disables receiver leveling. The default is disabled (unchecked).

**Leveling Receiver** - Select a receiver to be used to level the specified source by clicking on the **Leveling Receiver** pull down menu. Choose from a VNA Receiver or Ext. Device (PMAR).

![Leveling Receiver](image)

For a VNA Receiver, choose from any VNA receiver using standard or receiver notation.

To level power at the source output or DUT input choose the reference receiver for the source port. For example, to level the source power at port 1, then choose "R1". To level power at the DUT output, choose the receiver that is used to measure the DUT output. If the DUT output is connected to port 2, then select "B" or 'b2'. [Learn about Receiver Notation](#).

When **Phase Control** is enabled, the ratioed receivers used in Phase Control are selected and can NOT be changed. However, the Reference Source CAN also be selected for Receiver Leveling.

For Ext Device type, choose a configured PMAR device.

**Leveling Type - Sets the receiver leveling type:**

- **Pre-Sweep** - Leveling sweeps are performed in the background (not visible) before every measurement sweep to measure and apply source correction data.

- **Point** - Leveling is performed per point. If the point is outside the tolerance, the source power is adjusted and the point is read again. This process is repeated until the leveling receiver reports that the target power is within tolerance (or the maximum iteration setting is reached). When the iteration is done, it moves immediately to the next point.

- **Prior Sweep** - Disables the receiver leveling search function. After the first sweep, receiver leveling reads the data and computes the correction for each point in an attempt to reach the target power level. The calculated offset is used for the next sweep. This process is repeated
to improve receiver leveling for successive sweeps.

**Controlled Source Setup**

**Max Power** Always limits the maximum power out of the source to this value. The message: **Power set to Max Power** appears when this limit is reached.

If the maximum port power out of the VNA is reached at any time during the leveling sweeps, the following message appears: **Power set to user power limit.**

**Min Power** Always limits the minimum power out of the source to this value. The message: **Power set to Min Power** appears when this limit is reached. When Safe Mode is enabled, this value is used as the initial power level for the leveling loop process.

**Note:** The MAX/MIN limit is always used regardless of the safe mode state. In addition, the MAX/MIN limit is for port power and related to power offset. If the power offset is not set correctly, the MAX/MIN limit is not correct and it may impact the leveling. Ensure that the power offset in the channel is the same as power offset during calibration. If the exact power offset is not known, choose a limit for source and then it will not be related to power offset.

**Enable Safe Mode Leveling Using Max Step Size**

To protect your DUT and ensure stable results, these settings control the extent to which the source power will be changed to achieve the port power as measured at the reference receiver. These settings could be necessary when using external components with a large variation in frequency response (flatness).

When checked:

- The Min source output is used as the initial power level for the leveling loop process.
- The controlled source is never stepped more than the Max source step size.

When cleared:

- The initial power for the leveling loop may be determined by the Min source output, the Max source output, the last setting of the leveling loop, or the target value of the leveling loop.  **See Initial Power below.**
- The Max source step size is ignored.

**Max source step size**

When Safe Mode is enabled, the change in source power at each data point from one sweep to the
next is limited to this value. For example, assume Safe Mode is enabled, and Max Power Step is set to 1 dB. On the first leveling sweep, the first data point measures 3 dB lower than the port power, then source power for data point 1 will be increased by 1 dB for the next sweep, and likely for the following two sweeps.

**Update Source Power Calibration with Leveling Data**  Available only when using an RF Source and VNA receiver.

- When checked, the latest correction data is copied to the Source Power Cal correction array. When Leveling Mode is switched back to Internal (on the Power and Attenuators dialog), Source Power Cal is automatically turned ON using this correction data.
- When cleared, Source Power Cal is NOT turned ON when Leveling Mode is switched back to Internal.

**Source ALC Hardware**

NOT available with External sources.

- **Internal** - Internal ALC leveling and Receiver Leveling (Recommended).
- **Open Loop** - NO ALC leveling; Receiver Leveling ONLY. NOT available on N523x models.

**Leveling Receiver Setup**

**Leveling Tolerance**  The source is considered leveled when each stimulus data point has achieved the power level +/- (plus or minus) this tolerance value.

**Leveling Max Iterations**  If every stimulus data point does NOT achieve the port power after this number of leveling sweeps, the measurement sweep occurs using the correction values obtained from the last leveling sweep. The message: **Not settled, noisy trace** appears when the Max Iterations is reached. If you see this message, you can increase the Max Iterations, reduce the IFBW, or increase the Tolerance setting.

**Note:** Max Iterations can no longer be set to zero. Instead, select Prior Sweep mode to disable the receiver leveling search function. In this way, there will be no pre-sweep for the receiver
leveling, but the value of the receiver data will be used to correct the next sweep. This provides a post sweep correction and can be useful for correcting slow drift in a system where a booster amplifier or open loop ALC is used, without adding pre-sweeps to the sweep-acquisitions.

**Leveling Receiver Frequency** - Available ONLY when the selected receiver is a VNA Receiver or power meter. This setting determines which receiver frequencies are measured. Choose from:

- **Auto** - always uses the frequency range that is assigned to the measurement receiver.
- **FOM Receiver** - FOM Receiver frequency range. Learn more about Frequency Offset Mode.
- **FOM Source** - FOM Source frequency range.
- **DUT Input** - Mixer/Converter input frequency range.
- **DUT Output** - Mixer/Converter output frequency range.
- **FOM Primary** - Current Active Channel settings.

**Leveling IFBW**  Available only for VNA receivers. By default, the IFBW for the leveling sweeps is set to 100 kHz. Learn more about IFBW.

- Increase this value to make faster, but noisier leveling sweeps.
- Decrease this value to make slower, more repeatable leveling sweeps.
- Uncheck the box to use the same IFBW as the measurement sweeps.

**Leveling Noise BW** - (SA multitone and Modulation Distortion channel only) Sets the receiver noise bandwidth value for leveling at pre-sweep. Noise bandwidth is equal to the Resolution bandwidth divided by the Vector Average factor.

**Carrier Aperture Span/Offset** - (SA multitone Modulation Distortion channel only) Enabling the aperture settings will measure the power more quickly by reducing the span of the measurement during the pre-sweep. Enter the span and offset of the frequency aperture used to measure the signal power. Since the ideal modulation signal is known, the total power is calculated from this value.

**Initial Power Selection**

For each displayed data point, the leveling algorithm must select an initial power to begin the
iteration process. This value is chosen as follows:

Where:

\( P_{\text{Init}} \) = the initial power for the iteration process.

\( P_{\text{Final}} \) = the final power setting from the previous leveled sweep.

\( P_{\text{Min}} \) = the minimum controlled source output level as specified in the Receiver leveling setup.

\( P_{\text{Max}} \) = the maximum controlled source output level as specified in the Receiver leveling setup.

\( P_{\text{Target}} \) = the target power level for the selected leveling receiver.
A sweep is a series of consecutive data point measurements taken over a specified sequence of stimulus values. You can make the following sweep settings:

- **Number of Points** *(Separate topic)*

- **Sweep Type**
  - Linear / Log
  - Power Sweep
  - CW Time
  - Segment Sweep
  - Phase

- **Frequency Range**: Start/Stop *(Separate topic)*

- **Power Sweep**

- **Segment Sweep**
  - How to make segment sweep settings
  - Segment Table dialog

- **X-Axis Point Spacing - Segment Sweep ONLY**

- **Arbitrary Segment Sweep**

- **Sweep Time**

- **Sweep Setup**
  - Fast Sweep
  - Auto vs Stepped
  - Dwell and Delay
  - Standard vs Point Sweep
See Triggering and other 'Setup Measurements' topics

How to set Sweep Type

Using Hardkey/SoftTab/Softkey

1. Press **Sweep > Main > Sweep Type**.

Using a mouse

1. Right click on the **stimulus range** area under grid box.
2. Click **Sweep Setup....**
3. Select **Sweep Type** tab and select type.
4. Click **Apply** to implement the setting changes.

**Sweep Type** dialog box help

![Sweep Type dialog box](image)

**Note:** Sweep Settings are not applied until either **OK** or **Apply** is pressed.

**Channel**  The active channel when Sweep Type was selected. Sweep settings will be applied to this channel.

**Sweep Type**

**Linear Frequency**  Sets a linear frequency sweep that is displayed on a standard grid with ten equal horizontal divisions.
- **Start**  Sets the beginning value of the frequency sweep.
- **Stop**  Sets the end value of the frequency sweep.
- **Points**  Sets the number of data points that the analyzer measures during a sweep (Default is 201). Learn more.
- **Power** - Sets the power level or the source. Learn more.
- **IF Bandwidth** - Sets the IF (Receiver) bandwidth. Learn more.

**Log Frequency**  The source is stepped in logarithmic increments and the data is displayed on a logarithmic x-axis. This is usually slower than a continuous sweep with the same number of points.

- **Start**  Sets the beginning value of the frequency sweep.
- **Stop**  Sets the end value of the frequency sweep.
- **Points**  Sets the number of data points that the analyzer measures during a sweep (Default is 201). Learn more.
- **Power** - Sets the power level or the source. Learn more.
- **IF Bandwidth** - Sets the IF (Receiver) bandwidth. Learn more.

**Power Sweep**  Activates a power sweep at a single frequency that you specify. Learn about power sweep

- **Start Power**  Sets the beginning value of the power sweep.
- **Stop Power**  Sets the end value of the power sweep.
- **CW Freq**  Sets the single frequency where the analyzer remains during the measurement sweep.
- **Points**  Sets the number of data points that the analyzer measures during a sweep (Default is 201). Learn more.
- **IF Bandwidth** - Sets the IF (Receiver) bandwidth. Learn more.

**CW Time**  Sets the analyzer to a single frequency, and the data is displayed versus time. Learn more.

- **CW Freq**  Sets the frequency where the analyzer remains during the measurement.
- **Power** - Sets the power level or the source. Learn more.
- **Sweep Time**  Sets the duration of the measurement, which is displayed on the X-axis.
- **Points**  Sets the number of data points that the analyzer measures during a sweep (Default is 201). [Learn more.](#)
- **IF Bandwidth**  Sets the IF (Receiver) bandwidth. [Learn more.](#)

**Segment Sweep**  Sets the analyzer to sweep through user-defined sweep segments. [Learn how to make these settings.](#)

**Phase Sweep**  Sweeps the phase of one or more sources relative to another source. Choose values between -360° and +360°. [Learn more.](#)

  - **Start Phase**  Sets the beginning value of the phase sweep.
  - **Stop Phase**  Sets the end value of the phase sweep.
  - **CW Freq**  Sets the single frequency where the analyzer remains during the sweep.

**OK**  Applies setting changes and closes the dialog box.

**Apply**  Applies setting changes and leaves the dialog box open to make more setting changes.

**Cancel**  Closes the dialog. Setting changes that have been made since the last Apply button click are NOT applied.

**Help** - Display the **Sweep Type** dialog box help.

---

**Power Sweep**

A power sweep either increases or decreases source power in discrete steps. Power sweep is used to characterize power-sensitive circuits, with measurements such as gain compression.

In the Sweep Type dialog, specify Start power, Stop power, and CW Frequency. Power can be swept over any attainable range within the analyzer ALC range.

**Note:** If the VNA has source attenuators, and the attenuation must be changed in order to achieve the requested start and stop power, click **Stimulus**, then **Power** then **Power and Attenuators**.

The VNA does NOT allow power to sweep over a range that requires attenuator switching. However, two power sweeps can be performed in different channels. The attenuators will not be allowed to switch continuously, but triggering can be performed using single or group triggering. [Learn more.](#)

The remaining power settings apply in power sweep mode:
- Test Port Power setting is not available.
- Port Power can be coupled or uncoupled.
- Attenuator Control is always Manual.
- Power Slope (dB/GHz) is ignored (output frequency is CW).
- Press **Sweep > Main > Number of Points** to change the step size of the power sweep.

**Notes:**

- Using a **preference setting**, you can specify whether to maintain source power at either the start power or stop power level at the end of a power sweep.
- Power Sweep is optimized for speed. For highest measurement accuracy during a power sweep, it may be necessary to increase the **Dwell Time** to allow the source more time to settle.
- You may be able to perform a **60 dB power sweep** with Receiver Leveling. Learn how.

### Segment Sweep

Segment Sweep activates a sweep which consists of frequency sub-sweeps, called segments. For each segment you can define independent power levels, IF bandwidth, and sweep time.

Once a measurement calibration is performed on the entire sweep or across all segments, you can make calibrated measurements for one or more segments.

In segment sweep type, the analyzer does the following:

- Sorts all the defined segments in order of increasing frequency
- Measures each point
- Displays a single trace that is a composite of all data taken

Restrictions for segment sweep:

- The frequency range of a segment is not allowed to overlap the frequency range of any other segment.

**Note:** Segment sweeps can be overlapped for converter measurement types.
The number of segments is limited only by the combined number of data points for all segments in a sweep.

The combined number of data points for all segments in a sweep cannot exceed the max number of data points per trace.

All segments are FORCED to have power levels within the same attenuator range to avoid premature wear of the mechanical step attenuator. See Power Level.

### How to make segment sweep settings

#### Using Hardkey/SoftTab/Softkey

1. Press **Sweep > Segment Table > Segment Table...**

#### Using a mouse

1. Right click on the X-axis label or stimulus display area under grid line.

2. Click **Sweep Setup...** then select **Segment Sweep** under sweep type.

3. Click **Segment Table...**.

### Segment Table dialog box help

**Segment Table Softkeys**

- **Add Segment** - adds a sweep segment at last segment.
- **Insert Segment** - adds a sweep segment before the selected segment. You can also click the "down" arrow on your keyboard to quickly add many segments.
- **Delete Segment** - removes the selected segment.
- **Delete All Segments** - removes all segments.

**Note:** At least ONE segment must be ON or **Sweep Type** is automatically set to **Linear**.

### Segment Table dialog box

**X-Axis Point Spacing** - Check to scale the X-axis to include only the segments. [Learn more.](#)

**Allow Arbitrary Segments** - Check to allow arbitrary frequencies (overlapped or reverse sweep). [Learn more.](#)

**Display Center/Span Freq** - Check to display the center/span frequency.
Independent Setting Per Segment

Power Level - Sets the Power level for the segment. Also, the test port power can UNCOUPLE. See Power Coupling.

IF Bandwidth - Sets the IF Bandwidth for the segment.

IF Bandwidth Per Port - Sets the different bandwidth with different port for the segment.

Sweep Time - Sets the Sweep time for the segment.

Dwell Time - Specifies the time the source stays at each measurement point before the analyzer takes the data.

Vector Averaging - Enables SA vector average for the segment.

Video Bandwidth - Enables SA video bandwidth for the segment.

Delay - Sets the time to wait just before acquisition begins for each segment.

Sweep Mode - Sets the sweep mode to auto or stepped.

Shift LO - Sets the state of Shift LO.

Receiver Atten Per Port - Enables receiver attenuation per port for the segment.

Reference Tone - Enables SA multitone reference for the segment.

SA Data Threshold - Enables SA data threshold for the segment.

Save Table - Saves the setting changes in segment table.

Load Table - Apply the setting changes in segment table.

To Modify an Existing Segment

To make the following menu settings available, the segment table must be displayed first. (Press Sweep > Segment Table).

State - Click the box on the segment to be modified. Then, use the up/down arrow to turn the segment ON or OFF.

Start - Sets start frequency for the segment. Click the box and type a value and the first letter of a suffix (KHz, Mhz, GHz). Or double-click the box to select a value.
**Stop** - Sets stop frequency for the segment. Click the box and type a value and the first letter of a suffix (KHz, Mhz, GHz). Or double-click the box to select a value.

**Note:** The segment table truncates the frequency resolution. To verify the frequency resolution that you input, create a marker at the start or stop frequency settings.

**Points** - Sets number of data points for this segment. Insert a value or double-click the box to select a value.

To set **Power Level, IF Bandwidth/IF Bandwidth Per Port, Sweep Time, Delay, Receiver Attenuator Per Port, Sweep Mode** and **Shift LO** independently for each segment:

1. Press **Sweep > Main > Sweep Type > Segment Sweep.**
2. Click on **Segment Table > Segment Table...**
3. Check the box corresponding to the segment setting to set then click **OK.**
4. Click in the box at the bottom of the display and use the up/down arrows to enter a value or double-click the box and select a value with the numeric keypad.

**Note:** If the following are NOT set, the entire sweep uses the channel IFBW, Power, and Time settings.

**X-Axis Point Spacing - Segment Sweep ONLY**

This feature affects how a segment trace is drawn on the screen.

**How to select X-Axis Point Spacing**

**Using Hardkey/SoftTab/Softkey**

1. Press **Sweep > Segment Table. > Segment Table....**
2. Check **X-Axis Point Spacing.**

**Using a mouse**

1. Right click on the X-axis label or stimulus display area under grid line.
2. Click **Sweep Setup...** then select **Segment Sweep** under sweep type.
3. Click **Segment Table....**
4. Check **X-Axis Point Spacing.**

**Without X-axis point spacing**, a multi-segment sweep trace can sometimes result in squeezing many measurement points into a narrow portion of the x-axis.
- **With X-axis point spacing**, the x-axis position of each point is chosen so that all measurement points are evenly spaced along the x-axis.

For example, given the following two segments:

<table>
<thead>
<tr>
<th>STATE</th>
<th>START</th>
<th>STOP</th>
<th>POINTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ON</td>
<td>45.000000 MHz</td>
<td>50.000000 MHz</td>
</tr>
<tr>
<td>2</td>
<td>ON</td>
<td>170.000000 MHz</td>
<td>180.000000 MHz</td>
</tr>
</tbody>
</table>

**Without X-Axis Point Spacing**

**With X-Axis Point Spacing**

**Arbitrary Segment Sweep**

This feature allows arbitrary frequencies to be entered into the segment sweep table. With this capability, segments can have:
- overlapping frequencies.
- the stop frequency less than the start frequency (reverse sweep).

However, there are several limitations:

- Sweep mode: Stepped only.
- Sweep time: When reverse sweep is performed, the sweep time and the wait time before the measurement point becomes longer.

### How to enable Arbitrary Segment Sweep

**Using Hardkey/SoftTab/Softkey**

1. Press **Sweep > Segment Table. > Segment Table...**
2. Check **Allow Arbitrary Segments**.

**Using a mouse**

1. Right click on the X-axis label or stimulus display area under grid line.
2. Click **Sweep Setup...** then select **Segment Sweep** under sweep type.
3. Click **Segment Table...**
4. Check **Allow Arbitrary Segments**.

#### Notes:

- Unusual results may occur when using arbitrary sweep segments with markers, display settings, limit lines, formatting, and some calibration features.
- When **Allow Arbitrary Segments** is checked, a dialog is displayed indicating that X-axis point spacing is recommended.
- When the segment table has both forward and reverse frequency sweep, the correction interpolation may not work properly. Use the same segment table setting for both correction and measurement. (Make a measurement at the condition where "Cor" is displayed.)
- When the measurement data size of segment table exceeds its limitation, an error occurs. In this case, reduce the measurement data size (i.e., NOP, number of channels).

#### Sweep Time

The analyzer automatically maintains the fastest sweep time possible with the selected measurement settings. However, you can increase the sweep time to perform a slower sweep.
How to set Sweep Time

Using Hardkey/SoftTab/Softkey

1. Press **Sweep > Sweep Timing > Sweep Time**.
2. Input the desired sweep time.

Using a mouse

1. Right click on the X-axis label or stimulus range area under grid line.
2. Click on **Sweep Setup**....
3. Select **Timing** tab.
4. Deselect the **Auto Sweep Time** box.
5. Input the desired sweep time.

**Sweep Time** dialog box help

*Sweep Time* Specifies the time the analyzer takes to acquire RF data for a sweep. The maximum sweep time of the analyzer is 86400 seconds or 1 day. Learn about other settings that affect sweep speed.

**Note:** When a channel has an Aux In or SMU trace, the measurement time for DC voltage/current will be added.

**Note:** If sweep time accuracy is critical, use ONLY the up and down arrows next to the *Sweep Time* entry box in the *Sweep Setup* dialog to select a value that has been calculated by the analyzer. Do NOT type a sweep time value as it will probably be rounded up to the closest calculated value. This rounded value will not be updated in the dialog box.

- The actual sweep time includes this acquisition time plus some "overhead" time.
- The analyzer automatically maintains the fastest sweep time possible with the selected measurement settings. However, you can increase the sweep time using this setting.
- Enter 0 seconds to return the analyzer to the fastest possible sweep time.
- The Sweep Time setting is applied to the active channel.
- The sweep time is per sweep. A full 2-port cal requires two sweeps, both using the specified sweep time. Learn more.
- **A Sweep Indicator** appears on the data trace when the Sweep Time is 0.3 seconds or greater, or if trigger is set to **Point**. The indicator is located on the last data point that was measured by the receiver. If the indicator is stopped (point sweep mode) the source has already stepped to the next data point.
Sweep Setup

How to make Sweep Setup settings

Using Hardkey/SoftTab/Softkey

1. Press Sweep > Sweep Timing.

Using a mouse

1. Right click on the X-axis label or stimulus range area under grid line.
2. Click on Sweep Setup....

Sweep Setup dialog box help

Time

Sweep Time - same as Sweep Time Softtab Help.

Dwell Time - Specifies the time the source stays at each measurement point before the analyzer takes the data. Only applies to stepped sweep. The maximum dwell time is 20 seconds. See also Electrically Long Devices.

Sweep Delay - Specifies the time to wait just before acquisition begins for each sweep. This delay is in addition to Dwell Time (per point) and External Trigger delay if enabled.

Fast Sweep - NOT available on N5264B and N523xB models. When checked, in Analog Sweep mode the analyzer source settling times are shortened in both frequency and power-control (ALC) circuits. In Stepped Sweep mode, the settling time at ALL data points are shortened. This nearly doubles the sweep speed at preset conditions, but at the expense of frequency accuracy and a few dB of amplitude variation. For ratioed measurements, such as S-Parameters, these errors substantially ratio out.

- By default, Fast Sweep is always OFF to provide maximum accuracy and stability.
- Fast Sweep is NOT allowed with Power Limit enabled.
- Note: Performance specifications do NOT apply in Fast Sweep.

Sweep Mode

Auto - Automatic adjusts the sweep time and dwell time. When "Auto Sweep Time" is checked,
the sweep time and dwell time will disable to edit.

**Stepped** When checked (Stepped Sweep) the analyzer source is tuned, then waits the specified Dwell time, then takes response data, then tunes the source to the next frequency point. This is slower than Analog Sweep, but is more accurate when testing electrically-long devices.

When cleared (Analog Sweep) the analyzer takes response data AS the source is sweeping. The sweep time is faster than Stepped, but could cause measurement errors when testing electrically-long devices.

When the dialog check box is cleared, the analyzer could be in either Analog or Step mode. The mode can change from sweep to sweep. There is **NO way** to determine whether the analyzer is in Analog or Stepped Sweep. If you want to be sure what the current sweep mode is, then switch it to Stepped.

Stepped sweep is automatically selected for a number of reasons. Here are some of the reasons:

- **IF Bandwidth** is at, or below, 1 kHz.
- **Source Power Correction** is ON unless doing CW measurement.
- When more than one source is turned ON (multisource VNA models).
- When step mode is a faster way to take the data.
- For all **FOM and FCA** measurements.
- For all **ADC** measurements.
- For all **MMwave** measurements.

**Sweep Sequence**

**Standard Sweep** When checked, the analyzer sweeps all data points for each source port in turn. For a 2-port analyzer, this means that all data points are swept in the forward direction, then all data points are swept in the reverse direction. Even when NO reverse parameters are displayed (S22 or S12), reverse measurements are necessary when a full 2-port calibration is correcting the channel. This is the default behavior.

**Point Sweep** Available ONLY on standard S-parameter channels. When checked, the analyzer measures all parameters at each frequency point before stepping to the next frequency. The display trace is updated as each data point is measured.

- Point sweep usually results in slower sweeps and is useful only in rare circumstances.
- Point sweep is the same as stepped sweep mode on the 8510 and 8530.
- **PMAR** is not compatible with Point Sweep mode.
A trigger is a signal that causes the analyzer to make a measurement sweep. The analyzer offers great flexibility in configuring the trigger function.

View the interactive Trigger Model animation to see how triggering works.

- How to Set Trigger
- Source
- Scope
- Channel Settings
- Restart
- External and Auxiliary Triggering (separate topic)

How to set Triggering

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press Trigger &gt; Main &gt; Trigger....</td>
<td>1. Right click on the Trig or Meas icons on the status bar.</td>
</tr>
<tr>
<td></td>
<td>2. Select Trigger....</td>
</tr>
</tbody>
</table>

Note: The Continuous, Single, and Hold settings apply ONLY to the active channel. These settings are available from the Trigger menu, Active Entry keys, and softkeys.

Trigger Setup dialog box help
View the interactive Trigger Model animation to see how triggering works.

**Trigger Source**

These settings determine where the trigger signals originate for all existing channels. A valid trigger signal can be generated only when the analyzer is not sweeping.

**Internal** Continuous trigger signals are sent by the analyzer as soon as the previous measurement is complete.

**Manual** One trigger signal is sent when invoked by the Trigger button, the active tool bar, or a programming command.

**External** Trigger signals received from the trigger source. The trigger source can be defined in External and AUX Triggering.

**Pulse3** Trigger signal using internal Pulse3 to Meas Trig In bypass. Pulse3 can be defined in the Pulse Generators Setup dialog.

**Manual Trigger!** - Manually sends one trigger signal to the analyzer. Available ONLY when
Manual trigger is selected.

**Trigger Scope**

These settings determine what is triggered.

**Global**  All channels not in Hold receive the trigger signal [Default setting]

**Channel**  Only the next channel that is not in Hold receives the trigger signal. This is not obvious or useful unless Trigger Source is set to Manual. This setting enables Point Sweep mode.

**Active Channel** - Trigger are sent only to the active channel. The active channel does not change.

**Channel Trigger State**

These settings determine how many trigger signals the channel will accept.

**Continuous**  The channel accepts an infinite number of trigger signals.

**Groups**  The channel accepts only the number of trigger signals that is specified in the Number of Groups text box, then goes into Hold. Before selecting groups you must first increment the Number of Groups text box to greater than one.

**Number of Groups**  Specify the number of triggers the channel accepts before going into Hold. If in Point Sweep, an entire sweep is considered one group.

First increment to desired number, then select 'Groups'.

**Single**  The channel accepts ONE trigger signal, then goes into Hold.

Another way to trigger a single measurement is to set Trigger Source to Manual, then send a Manual trigger. However, ALL channels are single triggered.

**Hold**  The channel accepts NO trigger signals.

**Trigger Mode**

These settings determine what EACH signal will trigger.

Sweep and Point modes are available ONLY when both Trigger Source = MANUAL or EXTERNAL AND Trigger Scope = CHANNEL.

- **Channel**  Each trigger signal causes ALL traces in that channel to be swept in the order specified below.

- **Point**  Each Manual or External trigger signal causes one data point to be measured.
Subsequent triggers go to the same trace until it is complete, then other traces in the same channel are swept in the order specified below. When in Groups or Single trigger, the count is decremented by one after ALL data points on ALL traces in the channel are measured. See Also, the (point) Sweep Indicator and SCPI Triggering example for use with External.

For gated SA measurements, **Point** does not refer to a data point. Instead, it refers to the next LO acquisition. For SA, each time the LO is shifted an acquisition is captured. The time for each LO acquisition is based on the ADC Record Size and time per ADC sample (10 nsec).

- **Trace** Available ONLY when **Point Sweep** is selected. Each trigger signal causes two identical measurements to be triggered separately - one trigger signal is required for each measurement. Other trigger mode settings cause two identical parameters to be measured simultaneously. Trace triggering is NOT permitted when a channel is using a 2 port (or more) S-Parameter calibration.

- **Sweep** Each Manual or External trigger signal causes **ALL traces that share a source port** to be swept in the order specified below. When in Groups or Single trigger, the count is decremented by one after ALL traces in ALL directions are swept.

  When multiport correction is ON, which requires sweeps in more than one direction, traces on the screen will not update until all of the relevant directions have been swept. For example, with all four 2-port S-Parameters displayed:

  - When Full 2-port correction is ON, trigger 1 causes NO traces to update; trigger 2 causes ALL S-Parameters to update. Learn more about sweeps with correction ON.
  - When correction is OFF, trigger 1 causes S11 and S21 to update; trigger 2 causes S22 and S12 to update.
Trace Sweep Order

For ALL Trigger Modes, trigger signals continue in the same channel until all traces in that channel are complete. Triggering then continues to the next channel that is not in HOLD.

Traces within each channel are always swept in the following order:

- Traces are swept sequentially in source-port order. For example, in a channel with all four 2-port S-parameters, first the source port 1 traces (S11 and S21) are swept simultaneously. Then the source port 2 traces (S22 and S12) are swept simultaneously.

- In addition, when Alternate sweep is selected, traces are swept sequentially in source-port / receiver-port order. In the above example, first the S11 trace is swept, then S21, then S12, then S22.

Restart  (Available only from the Trigger menu) Restart stops the currently sweeping Channel, and continues on to the next Channel that can sweep.

Examples:

1. Single channel in Continuous Trigger mode: The current sweep on the channel is stopped, and the channel starts sweeping the first sweep of the Channel.

2. Single channel in Hold mode: The Restart softkey does not do anything.

3. Multiple Channels in Continuous mode: The current sweep on the current channel is stopped, and the next channel that can sweep is started.

See Also

- External and AUX Triggering
- Interactive Trigger Model animation
External and Auxiliary Triggering

External and auxiliary triggering is used to synchronize the triggering of the analyzer with other equipment.

**Note:** When an External Source is configured as an External Device, the PNA automatically controls all trigger settings. Do NOT make additional trigger settings. Learn more.

- Overview
  - How to make Trigger Settings:
    - Auxiliary Triggering
    - Meas Trig (IN) Dialog
    - Pulse Triggering

See Also

- Controlling a Handler
- Synchronizing an External Source
- Internal Triggering
- Pulse Triggering (separate topic)

Overview

**Ready Signals versus Trigger Signals**

A 'Ready for Trigger' signal is different from a Trigger signal. The ready signal indicates that the instrument sending the signal is ready for measurement. The instrument receiving the ready signal would then send a trigger signal, indicating that the measurement will be, or has been, made. Usually the slower instrument sends the trigger signal.

Learn more about each type of triggering signal:

- Meas Trig RDY and Meas Trig IN - This pair of signals is easy to use and limited in ability to configure.
• **AUX TRIG OUT** and **AUX TRIG IN** - These two pair of connectors and signals are highly configurable. Use them to synchronize with any number of devices and equipment.

**Meas (External) Trigger** dialog box help

![Trigger Dialog](image)

**Meas Trig RDY and Meas Trig IN**

The **MEAS TRIG** connectors are located on the VNA rear-panel.

These signals can be used when the VNA is communicating with a slow mechanical device. A material handler is very mechanical and takes a relatively long time to load and discharge parts. Here is how these signals work together to communicate:

1. The VNA sends a 'Ready' signal when it is ready to make a measurement.
2. The external device sends a trigger signal to the VNA when it is ready for a measurement.
3. Additional signals are available on the VNA Handler I/O to indicate that the VNA sweep has ended, and that the handler can setup for the next measurement. See Material Handler I/O description Material Handler I/O description

**See how to access the Trigger Dialog**

**Trigger Ready and Trigger IN**
The Trigger connectors are located on the rear-panel.

These signals can be used when the VNA is communicating with a slow mechanical device. A material handler is very mechanical and takes a relatively long time to load and discharge parts. Here is how these signals work together to communicate:

1. The VNA sends a 'Ready' signal when it is ready to make a measurement.
2. The external device sends a trigger signal to the VNA when it is ready for a measurement.
3. Additional signals are available on the VNA Handler I/O to indicate that the VNA sweep has ended, and that the handler can setup for the next measurement. See Material Handler I/O description.

**Dialog Settings**

To cause the VNA to respond to Meas Trig IN or Handler I/O signals, select **External** on the Trigger Setup tab, Source setting.

**Note:** You must select **External** when you use any external triggers.

Also on the Trigger Setup tab, Scope setting, choose whether one external trigger signal will apply to ALL channels (Global) or one trigger signal per Channel. The following settings apply accordingly.

**Main Trigger Input**

**Global / Channel Trigger Delay** After an external trigger is received, the start of the sweep is held off for this specified amount of time plus any inherent latency.

- When Trigger Scope = Channel, the delay value is applied to the specified channel.
- When Trigger Scope = Global, the same delay value is applied to ALL channels.

**Source** The VNA accepts Trigger IN signals through the following connectors:

- Meas Trig IN BNC
- Handler I/O Pin 18
- Pulse3

**Level / Edge**
**High Level**  The VNA is triggered when it is armed (ready for trigger) and the TTL signal at the select input is HIGH.

**Low Level**  The VNA is triggered when it is armed (ready for trigger) and the TTL signal at the select input is LOW.

**Positive Edge**  After the VNA arms, it will trigger on the next positive edge. If is set, the VNA will trigger as soon as it arms if a positive edge was received since the last data was taken.

**Negative Edge**  After the VNA arms, it will trigger on the next negative edge. If is set, the VNA will trigger as soon as it arms if a negative edge was received since the last data was taken.

**Accept Trigger Before Armed**  When checked, as the VNA becomes armed (ready to be triggered), the VNA will immediately trigger if any triggers were received since the last taking of data. The VNA remembers only one trigger signal. All others are ignored.

- When this checkbox is cleared, any trigger signal received before VNA is armed is ignored.
- This feature is only available when positive or negative EDGE triggering is selected.
- Configure this setting remotely using CONTrol:SIGNal (SCPI) or ExternalTriggerConnectionBehavior (COM).

**Ready for Trigger Indicator (Trigger Ready)**

On the VNA, when External is selected on the Trigger Setup tab, then both Meas Trig IN and Meas Trig Ready are enabled.

**Note:** The Ready for trigger is available only when the trigger source is set at External.

Choose a connector to send the VNA Ready OUT signal:

- **Meas Trig RDY**
- **Handler I/O p21**

Choose Polarity of the 'Ready OUT' signal.

- **Ready High** - TTL High indicates the VNA is ready for trigger.
- **Ready Low** - TTL Low indicates the VNA is ready for trigger (default setting).

See Also
Pulse Trigger Tab

Learn how to External Trigger during Calibration

**Aux Trig 1 - Aux Trig 2 dialog box help**

![Trigger dialog box](image)

See how to access the Trigger Dialog

Note: When an External Source is configured as an External Device, the VNA automatically controls all trigger settings. Do NOT make additional trigger settings. Learn more.

**AUX TRIG OUT and AUX TRIG IN**

See the AUX TRIG (1&2) connectors on the VNA rear-panel.

These signals are highly configurable. They can be used with all types of external devices to send and receive signals. However, it is important to note that either Aux Trig INPUT does NOT trigger the VNA. That signal must be selected. See step 2 in the following procedure.
1. An external source sends a 'Ready' signal to the VNA (at the Aux Trig IN connector) when it is settled at a frequency.

2. After receiving the Ready signal, the VNA begins the measurement when it receives a Trigger signal from the specified Trigger Source:

   - **Internal** - Measurement begins immediately.
   - **Manual** - Measurement begins when the VNA Trigger button is pressed.
   - **External** - Measurement begins when Meas Trig In signal is received from an external device. This must be configured independently.

   - The Aux Trig OUT signal can be configured to be sent either just BEFORE the measurement is made or AFTER the measurement is complete. When communicating ONLY with an external source, the Aux Trig OUT signal should be sent AFTER the measurement is complete to indicate that the external source can setup for the next measurement.

**Dialog Settings**

The Aux Trig 1 and Aux Trig 2 tabs are identical. Two pair of connectors are available to allow two external devices to be controlled simultaneously.

**Enable** Check to use the Aux1 or Aux2 connectors to output signals to an external device.

**Channel:** This setting is controlled by a VNA Preference setting.

   - **Global** - ALL Aux Trig settings apply to ALL channels. The Per Point setting (see below) is made on the Trigger Setup tab which also applies to ALL channels.
   - **Channel** - ALL Aux Trig settings apply to the specified channel. Each channel can be configured independently.

**AUX TRIG OUT (To Device)**

The following settings control the properties of the signals sent out the rear panel AUX TRIG OUT (1&2) connectors:

**Polarity**

   - **Positive Pulse**  Outgoing pulse is positive.
   - **Negative Pulse**  Outgoing pulse is negative.

**Position**
**Before Acquisition**  Pulse is sent immediately before data acquisition begins.

**After Acquisition**  Pulse is sent immediately after data acquisition is complete.

**Per Point**  Check to cause a trigger output to be sent for each data point. Clear to send a trigger output for each sweep.

When the Aux Trig - "Global" VNA Preference is selected, then the Point setting is made on the Trigger Setup tab. It then applies to ALL channels. When more than one channel is present, the channel setting that was made last is used.

**Pulse Duration**  Specifies the duration of the positive or negative output trigger pulse.

**Backplane**: (M980xA, M937xA, P937xA) Select PXI backplane trigger No.

**Rear SMB**: (USB VNAs except P937xA) Select the trigger port number on rear panel. None: Neither port 1 nor 2 is selected.

**Enable Wait-for-Device Handshake**

When checked, the VNA waits indefinitely for the input line at the rear panel AUX TRIG IN (1&2) connectors to change to the specified level before acquiring data. This signal indicates that the external device is ready for VNA data acquisition. If the signal arrives before the VNA is ready to acquire data, it is latched (remembered).

When NOT checked, the VNA does not wait, but outputs trigger signals when the VNA is ready. This signal does NOT trigger the VNA. The trigger signal is generated from Trigger Sources: Internal, Manual, or External.

**IN (READY)**

**Positive Edge**  VNA responds to the leading edge of a pulse on the Aux1 or Aux2 In connector.

**Negative Edge**  VNA responds to the trailing edge of a pulse on the Aux1 or Aux2 In connector.

**Delay**  Time that the VNA waits after receiving the Handshake input before data acquisition begins.

**See Also**

- See how to use these connectors to synchronize with External Sources.
- Pulse Triggering
Note: Guided and Unguided Calibration CAN be performed in External Trigger mode. With this optional behavior, while Trigger Source is set to External, trigger signals must be sent for Calibration sweeps. This behavior does not apply to FCA calibrations.

You can set a Preference to calibrate using Internal trigger signals while Trigger Source is set to External.

The following dialog box appears on the screen while waiting for an External trigger signal.

Click Abort to cancel the wait for a trigger signal.
Reference

Note: This feature applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above.

The frequency reference can be set to use the internal reference in the VNA or use an external reference connected to the 10 MHz REF IN connector on the rear panel.

The 10 MHz REF IN on the rear panel accepts an external frequency reference of 10 MHz, 20 MHz, or 100 MHz. The 10 MHz REF OUT connector on the rear panel is used to connect the frequency reference to other test equipment.

Accessing Reference Settings

Using Hardkey/SoftTab/Softkey

1. Press Setup > Internal Hardware > Reference...

Using a mouse

1. Click Instrument.
2. Select Setup.
3. Select Internal Hardware.
4. Select Reference.
Auto-select Reference (10 MHz Only) - When selected (default), the frequency reference selects between the internal 10 MHz reference or external 10 MHz reference automatically. The external reference will be selected if a signal is detected at the 10 MHz REF IN connector on the rear panel.

Reference In

- **Internal** - Selects the 10 MHz internal frequency reference.
- **External 10 MHz** - Selects an external 10 MHz frequency reference connected to the 10 MHz REF IN connector on the rear panel.
- **External 100 MHz** - Selects an external 100 MHz frequency reference connected to the 10 MHz REF IN connector on the rear panel.
- **External 20 MHz** - Selects an external 20 MHz frequency reference connected to the 10 MHz REF IN connector on the rear panel.

Reference Out

- **10 MHz** - Selects 10 MHz frequency reference.
- **100 MHz** - Selects 100 MHz frequency reference.
About the trigger model

Read Text description of triggering behaviors.

This model does not include Sweep trigger mode.
Data Format

A data format is the way the analyzer presents measurement data graphically. Pick a data format appropriate to the information you want to learn about the test device.

- How to set Format
- Rectangular (Cartesian) Display Formats
- Polar
- Smith Chart
- Temperature

See other 'Setup Measurements' topics

How to set the Display Format

Using **Hardkey/SoftTab/Softkey**

1. Press **Format > Format 1 or Format 2.**

Using a mouse

1. Right-click on the trace status area above the grid box.
2. Click **Format.**
3. Select the desired format.
Format dialog box help

Click a link to learn about that format:

<table>
<thead>
<tr>
<th>Log Mag</th>
<th>Polar</th>
<th>Complex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase / Unwrapped Phase</td>
<td>Linear Mag</td>
<td>Temperature</td>
</tr>
<tr>
<td>Group Delay</td>
<td>SWR</td>
<td>Kelvin</td>
</tr>
<tr>
<td>Smith / Inverse Smith Chart</td>
<td>Real</td>
<td>°F</td>
</tr>
<tr>
<td>Positive Phase</td>
<td>Imaginary</td>
<td>°C</td>
</tr>
</tbody>
</table>

Format Unit

Only the following Formats allow a Unit selections:

**Log Mag** - Choose from:

- dB (Decibel)
- dBm (Power)
- dBmV (dB milli Volts) - used for unratioed receiver measurements.
- dBuV (dB micro Volts) - used for unratioed receiver measurements.
- dBmA (dB milli Amps) - used for unratioed receiver measurements.

**Lin Mag** - Choose from:

- U (no units), W (Watts), V, (volts), A (amps)

Rectangular Display Formats

Seven of the nine available data formats use a rectangular display to present measurement data. This display is also known as Cartesian, X/Y, or rectilinear. The rectangular display is especially useful for clearly displaying frequency response information of your test device.

- Stimulus data (frequency, power, or time) appears on the X-axis, scaled linearly
- Measured data appears on the Y-Axis.

**Log Mag (Logarithmic Magnitude) Format**

- Displays Magnitude (no phase)
- Y-axis: dB
- Typical measurements:
  - Return Loss
  - Insertion Loss or Gain

**Phase Format**

Measures the phase of a signal relative to the calibration reference plane with a range of +/- 180 degrees.

- Displays Phase (no magnitude)
- Y-axis: Phase (degrees)
- The trace 'wraps' every 180 degrees for easier scaling.
- Typical Measurements:
  - Deviation from Linear Phase

**Unwrapped Phase**

- Same as Phase, but without 180 degree wrapping.

**Note:** Phase is unwrapped by comparing the phase from one data point to the next. If the phase difference between two points is greater than 180 degrees, or if the phase of the first data point is greater than 180 degrees from DC, than the phase measurement is probably NOT accurate.

**Positive Phase**

Displays the phase wrapped between 0 to +360 degrees.

**Group Delay Format**

- Displays signal transmission (propagation) time through a device
- Y-axis: Time (seconds)
- Typical Measurements:
  - Group Delay
See Also:

- Group Delay (Measurement)
- Comparing the analyzer Delay Functions.
- Phase Measurement Accuracy

**Linear Magnitude Format**

- Displays positive values only

- Y-axis: Unitless (U) for ratioed measurements
  Watts (W) for unratioed measurements.

- Typical Measurements:
  - reflection and transmission coefficients (magnitude)
  - time domain transfer

**SWR Format**

- Displays reflection measurement data calculated from the formula \((1+\rho)/(1-\rho)\) where \(\rho\) is reflection coefficient.

- Valid only for reflection measurements.

- Y axis: Unitless

- Typical Measurements:
  - SWR

**Real Format**

- Displays only the real (resistive) portion of the measured complex data.

- Can show both positive and negative values.

- Y axis: Unitless

- Typical Measurements:
  - time domain
  - auxiliary input voltage signal for service purposes
**Imaginary Format**

- Displays only the imaginary (reactive) portion of the measured data.
- Y-axis: Unitless
- Typical Measurements:
  - impedance for designing matching network

**Polar Format**

Polar format is used to view the magnitude and phase of the reflection coefficient ($\Gamma$) from your $S_{11}$ or $S_{22}$ measurement.

You can use Markers to display the following:

- Linear magnitude (in units) or log magnitude (in dB)
- Phase (in degrees)

- The dashed circles represent reflection coefficient. The outermost circle represents a reflection coefficient ($\Gamma$) of 1, or total reflected signal. The center of the circle represents a reflection coefficient ($\Gamma$) of 0, or no reflected signal.
- The radial lines show the phase angle of reflected signal. The right-most position corresponds to zero phase angle, (that is, the reflected signal is at the same phase as the incident signal). Phase differences of $90^\circ$, $\pm 180^\circ$, and $-90^\circ$ correspond to the top, left-most, and bottom positions on the polar display, respectively.
Smith Chart Format

The Smith chart is a tool that maps the complex reflection coefficient ($\Gamma$) to the test device's impedance.

In a Smith chart, the rectilinear impedance plane is reshaped to form a circular grid, from which the series resistance and reactance can be read ($R + jX$).

You can use Markers to display the following:

- Resistance (in units of ohms)
- Reactance as an equivalent capacitance (in units of farads) or inductance (in units of henrys)

Inverse Smith Chart (also known as Admittance)

Same as standard Smith Chart, except:

- The plot graticule is reversed right-to-left.
- Admittance (in units of siemens) instead of resistance.
Interpreting the Smith Chart

- Every point on the Smith Chart represents a complex impedance made up of a real resistance (r) and an imaginary reactance (r+jX).

- The horizontal axis (the solid line) is the real portion of the impedance - the resistance. The center of the horizontal axis always represents the system impedance. To the far right, the value is infinite ohms (open). To the far left, the value is zero ohms (short).

- The dashed circles that intersect the horizontal axis represent constant resistance.

- The dashed arcs that are tangent to the horizontal axis represent constant reactance.

- The upper half of the Smith chart is the area where the reactive component is positive and therefore inductive.

- The lower half is the area where the reactive component is negative and therefore capacitive.

Complex

- Displays the real (resistive) portion of the measured complex data on the X axis and the imaginary (reactive) portion of the measured complex data on the Y axis.

X axis annotation
There is X-axis annotation for 2-D plot, and it takes window area for all traces in the window. The X-axis annotation area is reserved when there is at least one trace that the format is complex.

The annotation area can be turned off with the SCPI command :DISPlay:WINDow:ANNotation:X:STATe.

When the rectangular trace is selected, the X axis annotation area gets blank, while the complex trace still showing.
When there is no trace with complex format, the X axis annotation area is collapsed.

When :DISPlay:WINDow:ANNotation:X:STATe is set to off, the X axis annotation area is also collapsed even if there are some traces with complex format.

**Kelvin, °F, and °C**

Used to display temperature, primarily with the Noise Figure application. [Learn more.](#)
The Scale, Reference Level and Reference Position settings (along with Format) determine how the data trace appears on the VNA screen.

- Scale, Reference Level and Position
- Scale Coupling
- Magnify Mode and Zoom Preference
- Electrical Delay (Separate topic)
- Magnitude Offset & Magnitude Slope
- Y-Axis Graph (Lin/Log Scale), Top, Bottom, Ref X/Y Level, Ref X/Y Position
- Phase Offset (Separate topic)

See other ‘Setup Measurements’ topics

Scale, Reference Level and Position

The Scale, Reference Level and Reference Position settings (along with format) determine how the data trace appears on the VNA screen.

How to set Scale, Reference Level, and Position

Using Hardkey/SoftTab/Softkey

1. Press Scale > Main > Scale / Reference Level / Reference Position.
2. Input the desired value.

Using a mouse

1. Right-click on Y-axis annotation or the trace status label above the grid box.
2. Select Scale....

Scale dialog box help
Note: The scale settings are set to couple with other traces in each window. The following settings assume that Scale Coupling is set to OFF. [Learn more about Scale Coupling.]

**Scale**

**Per Division** Sets the value of the vertical divisions of a rectangular display format. In Polar and Smith Chart formats, scale sets the value of the outer circumference. Range: 0.001dB/div to 500 dB/div.

**Tip:** Click on the Y-axis labels, then use a mouse scroll wheel to change scale in preset increments. Or Right-click on **Y-axis** annotation to change Scale.

**Autoscale** - Automatically sets value of the vertical divisions and reference value to fit the ACTIVE data trace within the grid area of the screen. The stimulus values and reference position are not affected.

The analyzer determines the smallest possible scale factor that will allow all the displayed data to fit onto 80 percent of the vertical grid.

The reference value is chosen to center the trace on the screen.

**Tip:** Double click on the Y-axis labels to autoscale the active trace.

**Autoscale All**  Automatically scales ALL data traces in the ACTIVE WINDOW to fit vertically within the grid area of the screen.

**Reference**

**Level** In rectangular formats, sets the value of the reference line, denoted by 0.00 dB on the
screen. Range: -500 dB to 500 dB.

In Polar and Smith chart formats, reference level is not applicable.

**Tip:** Click on the Y-axis labels, then drag up or down to change the reference level in preset increments.

**Position** In rectangular formats, sets the position of the reference line. Zero is the bottom line of the screen and ten is the top line. Default position is five (middle).

In Polar and Smith chart formats, reference position is not applicable.

**Tip:** Click on the triangle, then drag up or down to change the reference position in preset increments.

---

**Scale Coupling**

With Scale Coupling enabled, traces that have the same format will have the same Scale, Reference Level, and Reference Position. You can choose to couple the scale of traces that are in the same window, couple the scale of all traces in all windows, or to have NO coupling.

**How to set Scale Coupling**

**Using Hardkey/SofTab/Softkey**

1. Press **Scale > Main > Scale Coupling...**

**Using a mouse**

1. Right-click on **Y-axis** annotation.
2. Select **Scale Coupling...**

---

**Scale Coupling** dialog box help
Allows traces that share the same format to have the same Scale, Reference Level and Reference Position.

**Coupling Method**

- **Off** - No coupling. Traces are scaled individually. Default setting.
- **Window** - All traces with the same format in each selected window share the same scale settings.
- **All** - All traces in ALL selected windows with the same format share the same scale settings.

- When **Window** or **All** coupling is enabled, the scale settings for the active trace are assumed by other coupled traces with the same format.
- When there are traces with a different format present, all traces with that format assume the trace settings of the lowest-numbered trace of that format.
- Once enabled, scale settings for all coupled traces with the same format can be changed with any coupled trace being active.

**Selected Windows**

Available when either the **Window** or **All** method is selected. Selected windows will participate in scale coupling. All windows are selected by default. Clear a checkbox to 'Opt-out' of scale coupling for that window.
About Autoscale and Scale Coupling

**Autoscale** (not Autoscale All) affects the active trace in the active window. All traces that are coupled to this trace assume the new scale settings of the active trace. This could cause some traces to NOT show on the screen.

**Autoscale All** with Coupling Method...

- **Off** - All traces in the active window are autoscaled independently.
- **Window** - All traces in each selected window are autoscaled to fit within a common set of scaling factors.
- **All** - All traces in all selected windows are autoscaled to fit within a common set of scaling factors.

Y-Axis Graph (Lin/Log Scale), Top, Bottom, Ref X/Y Level, Ref X/Y Position

**How to set Y-Axis Spacing**

Using **Hardkey/SoftTab/Softkey**

1. Press **Scale > Main > Y-Axis Spacing / Top / Bottom / Ref Y Level / Ref Y Position / Ref X Level / Ref X Position**.

**Y Axis Spacing** - Selects either a linear or log scale Y-axis format.

**Top** - Sets the maximum scale value for the Log Y-axis.

**Bottom** - Sets the minimum scale value for the Log Y-axis.

**Ref Y Level** - Sets the Y axis Reference Level of the specified trace in the specified window.

**Ref Y Position** - Sets the Y axis Reference Position of the specified trace in the specified window.

**Ref X Level** - Sets the X axis Reference Level of the specified trace in the specified window.

**Ref X Position** - Sets the X axis Reference Position of the specified trace in the specified window.
**Graph Area Graticules**

In the graph area, multiple traces can be drawn overlaid. Different graticule is drawn based on the active trace.

The following is an example of a case that these 3 traces are shown:

Trace 1 = Log (10k to 1M)

Trace 2 = Log (5k to 500k)

Trace 3 = Linear

When trace 1 is active.
When Trace 2 is active (graticule is still log, but different from Trace 1)

When Trace 3 is active.

**Magnify Mode and Zoom Preference**

The magnify feature allows to magnify all traces in the active window. It allows to zoom into a portion of the display to see the response in detail.
How to magnify the trace

Using Hardkey/SoftTab/Softkey
None

Using a mouse

1. Left-click and select the area you want to magnify
2. Select Magnify from the pop-up menu.

Other methods to set the magnify mode includes:

- When the Zoom Preference is set to Magnify, a 2-finger spread gesture will turn ON the Magnify mode
- Click on the Magnify icon at the top of the screen
- Enable the zoom box select mode and drag the zoom box to select an area, and then choose the Magnify option in the popup menu.

When the Magnify Mode is turned on:
2-finger pinch on the display changes the x and y magnification. If magnification is pinched down to 1:1 scaling, the magnify mode will be turned off.

1-finger drag across display changes the reference of the magnified x and y.

Zoom Preference Dialog Box Help
Zoom Preference allows you to decide the default operation of the 2-finger pinch gesture, from these 4 choices:

2-finger pinch automatically turns on magnify mode so that x and y-axis can be magnified/ This is the default setting.

2-finger pinch changes y-axis scale.

2-finger pinch changes y-axis scale and x-axis stimulus.

2-finger pinch does nothing.
Magnitude Offset allows you to offset the magnitude (not phase) data by a fixed and / or sloped value in dB. If the display format is Linear Magnitude or Real (unitless), the conversion from dB is performed and the correct amount of offset is implemented.

How to set Magnitude Offset

Using Hardkey/SoftTab/Softkey

1. Press Scale > Constants > Mag Offset / Mag Slope.

Magnitude Offset dialog box help

Magnitude Offset allows you to offset the magnitude (not phase) data by a fixed and / or sloped value in dB. If the display format is Linear Magnitude or Real (unitless), the conversion from dB is performed and the correct amount of offset is implemented.

The Magnitude offset setting affects only the active trace.

**Mag Offset** Offsets the entire data trace by the specified value.

**Mag Slope** Offsets the data trace by a value that changes with frequency. The offset slope begins at 0 Hz.

For your convenience, the offset value at the start frequency is calculated and displayed.

See where this operation is performed in the data processing chain.
PNA-X RF Path Configurator

Allows you to configure hardware components that are available with selected PNA-X options.

N522xB and N523x models do NOT have the RF Path Configurator.

How to access Path Configurator

Using Hardkey/SoftTab/Softkey

1. Press Setup > Internal Hardware > RF Path Config....

Using Menus:

1. Click Instrument.
2. Select Setup.
3. Select Internal Hardware.
4. Select RF Path Config.

The following image shows configuration with PNA-X Opt 423 (4-port, internal 2nd source, combiner, and mechanical switches). Your PNA-X may not include these options.

See Also

IF Path Configuration

PNA-X specs for block diagrams

Internal Second Source limitations

VNA Configurations and Options
**Path Configuration** dialog box help

See Noise Figure tab of the Path Configuration.

**Note:** With selected PNA-X options, pulse modulation is available ONLY on OUT1 of Src 1 and Src 2. See block diagram.

Different paths can be configured for each channel.

**Configuration**

Select, store, and delete factory configurations or user-defined configurations. Configurations are stored on the VNA hard drive.

Any configuration can be saved, and later recalled, from this dialog. Click **Store**, type a configuration name, then click **OK**.

**Text area** Displays text describing the physical connections required to complete the configuration. The text for factory configurations can NOT be edited. Text is saved as part of the configuration.

**Cancel** Closes the dialog and returns the configuration settings to the state they were in when the dialog was opened. Cancel does NOT undo **Store** and **Delete** actions that were performed while the dialog was open.
Notes

- Click or touch anywhere within a box on the screen to cycle through the available settings.
- Some switch settings alter graphics in areas other that where the switch is thrown.

- **If you don’t hear switches clicking**, this could be why:
  - Electronic switches are **orange** on the path configuration dialog. These switches do not make noise when being thrown. Mechanical switches are **blue**.
  - The channel is not sweeping.
  - The following selections do NOT throw switches, but simply indicate how to connect jumper cables by drawing red lines on the dialog:
    - Combiner (Normal/Reversed)
    - Port 2 Source (Src2 OUT1/Src1 OUT2)

**Note:** To prevent premature wear, the VNA does not allow attenuators or other mechanical switches to switch continuously.

These mechanical devices are set for the entire channel. When more than one channel is used, and a mechanical device setting is NOT the same for all channels, only the ACTIVE channel is allowed to sweep. All other channels are NOT allowed to sweep (Blocked).

Learn how to view the settings of all mechanical devices in the VNA.

- Red lines are jumpers on the front or rear panel.

- **Src1 / Src2 Settings:**
  - **Low Bnd Filtered** (default setting) reduces harmonics below 3.2 GHz on OUT1 of both **Src1** and **Src2**.
  - **Low Bnd Hi Pwr** setting does not use the filtering which causes higher power below 3.2 GHz. See Max Leveled Power in the specifications for your PNA-X model.
  - Each source optionally has pulse modulation capability.

**Note:** Pulse modulation is available ONLY on OUT1 of Src 1 and Src 2. See block diagram. Your VNA must also have Pulse options. Learn more.
- Copy channel feature also copies path configuration settings.

- Path Configuration is saved and recalled as part of an instrument state.

**Factory Configurations**

**Note:** Recalling a stored configuration will over-write MANY RF and IF path configuration settings. Make your measurement settings AFTER recalling a stored configuration, NOT before.

**Port 1 2-tone** Routes Source 2 through the internal combiner to create a two-tone signal out port 1. The standard jumper configuration is used. This is the configuration that is used to make IMD measurements.

**Src 2 Out Port 2** (Option 423 ONLY) Routes Source 2 (OUT1) to port 2 using an external cable attached to the rear-panel J8 to J1. This allows source power at port 1 AND port 2 simultaneously. Source power is NOT available at port 3. Learn more about Internal Second Source capability.

**2 port Dual Source** (Option 224 ONLY) Routes Source 2 (OUT1) to port 2 using an external cable attached to the rear-panel J8 to J1. This allows source power at port 1 AND port 2 simultaneously. This also allows pulsed measurements to be performed on both ports 1 and 2.

**Hot S-parameters** The rear panel jumpers must be manually reconfigured to allow the source 2 signal to be routed through the thru path of the internal combiner. Source 2 is used to drive the AUT into compression, so the highest possible output power is required. Source 1, routed through the coupled path of the combiner, is used for S-parameter measurements, so a small signal is sufficient. These two signals are combined but the frequencies are usually offset.

**See Also**

- Configuration for High-power measurements
Source Phase Control (Opt S9x088A/B)

Option S9x088A/B allows you to control the phase of a VNA source or an external source. Two sources are required.

- Overview
- Features and Limitations
- Phase Control Use-Cases
- How to make Phase Settings
- Calibration and Phase Control
- Active Load Pull Examples
- Phase Control with N Sources

See other 'Setup Measurements' topics

Overview

The Source Phase Control feature provides a specific phase difference between two sources. The phase difference can be fixed (for example, at 90 degrees), or swept between two arbitrary phase values (for example, from 0 to 360 degrees).

This feature is allowed on ALL PNA-X models, any N522x, E5080B, M980xA and P50xxA model with either two internal sources or an external source and configurable test set.

Any combination of VNA internal or external sources may be used. One source is selected as the controlled source and the other source is the reference source. You select the two sources by selecting the ports at which the sources are available. The choice of ports is limited for you on the Phase Control dialog. Learn about these limitations.

In addition to selecting source ports, you also select the receivers to be used to measure the phase for the sources. This can be test port receivers or the reference receivers for the specified source ports. The receivers measure the relative phase of the sources, then adjust the phase of the controlled source, then remeasure until the phase difference is within the tolerance that you specify.

Phase can also be set without using any receivers. This is called 'Open Loop' mode. In this mode, the phase of the controlled source is set once, and iterations are not done, resulting in phase that is less
accurate and stable compared to using receivers to measure and set phase. Use Open Loop mode when you need to use the receivers to measure other parameters.

The phase of the controlled source can be swept relative to the reference source. The phase difference between the controlled and reference source is incremented and iterated on consecutive data points. Before starting the Phase Control dialog, select Sweep Type = Phase Sweep. Learn how.

When the phase of a source is controlled, the power of that source is also controlled using Receiver Leveling. Instead of the normal receiver-leveling mode where only one receiver is used, when phase control is active the ratio of two receivers is used to level the power of the controlled source. This is useful for making active load-pull measurements as described below. In Open Loop mode, neither the phase nor the power of the source is controlled.

Features and Limitations

- Phase Control is allowed ONLY in a standard S-parameter channel.
- Phase Control can be used with Wideband pulse measurements - NOT in narrow-band pulse mode.
- Point Averaging is NOT allowed.
- External sources are supported. Learn how to Configure an External Device. Phase can be controlled on Keysight MXG, PSG, ESG and EXG sources. The external source must be routed through the rear panel so that a reference receiver can measure its phase. Use the Path Configuration dialog to make switch settings and enable FOM mode.
- Remote commands are available that allow the phase and power of each point to be set individually, much like in source power calibration. Use these commands if you need to create a specific pattern of amplitude/phase states, such as characterizing the load-pull of an amplifier.

Phase Control Use-Cases

Phase control and phase sweep is useful in the following applications:

Active load control

Provide a controlled, electronically-settable impedance to the output port of a DUT under fixed or swept-frequency conditions. Some examples are: measuring the gain and output power of an amplifier with a known load, and measuring the output from a directional detector with a known load.

The reference source is applied to the DUT input port, and the controlled source is applied to the DUT output port as a reverse input wave. The phase and power level of the controlled source is set relative to the forward output wave of the DUT (which is determined by the reference source), so that any arbitrary load impedance (gamma) can be set.
Optionally, the phase of the controlled source can be swept with a constant frequency, so that the phase of gamma rotates while the magnitude of gamma remains constant. The ratio of reverse input wave and forward output wave as viewed on a Smith chart or polar display would appear as a circle. This capability can be combined with external load-pull software to create traditional load-pull power contours.

**Phase-controlled sources**

Set the phase and magnitude of one source relative to a reference source, to provide differential, quadrature, or arbitrary phase-offset signals at a fixed or swept frequency. Typically, another instrument, receiver, or detector would be required to measure the response of the DUT.

**How to make Phase Control settings:**

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Sweep &gt; Main &gt; Sweep Setup...</strong></td>
<td>1. Click <strong>Stimulus</strong>.</td>
</tr>
<tr>
<td>2. In the Sweep Setup dialog select <strong>Phase Sweep</strong>.</td>
<td>2. Select <strong>Sweep</strong>.</td>
</tr>
<tr>
<td></td>
<td>3. Select <strong>Sweep Setup...</strong>.</td>
</tr>
<tr>
<td></td>
<td>4. In the Sweep Setup dialog select <strong>Phase Sweep</strong>.</td>
</tr>
</tbody>
</table>

**To perform a Phase Sweep:**

| 1. Press **Sweep > Main > Sweep Type**. | 1. Click **Stimulus**. |
| 2. Select **Phase Sweep**. | 2. Select **Sweep**. |

**To select (view) a Phase format: (Learn more)**

| 1. Press **Format > Format 1 > Phase**. | 1. Click **Response**. |
| | 2. Select **Format**. |
| | 3. Select **Phase, Unwrapped Phase**, or **Smith**. |

**To access the Phase Control dialog:**

| 1. Press **Sweep > Source Control > Phase Control**. | 1. Click **Stimulus**. |
| | 2. Select **Sweep**. |
3. Select **Source Control**.
4. Select **Phase Control**.

**Phase Control** dialog box help

**Sweep Type**  See **Sweep Type** for information.

**Port N**  Select a port to set its phase control configuration. This is the 'controlled' port.

**Phase**  Specify any Fixed Phase setting.

**Start / Stop Phase**  Available when Sweep Type is set to Phase Sweep. **Learn how**. Enter the Start and Stop phase values in degrees.

**Phase Control**  Click in the port cell to be controlled, then choose from the following:
- **Off** - Phase is NOT set or controlled.

- **Open Loop** - Phase is set, but receivers are NOT used to measure and iterate the phase or power of the source. Therefore, the setting of phase is not as accurate or stable. Open Loop mode can be used with phase sweep (for example, from 0 to 360 degrees). However, each sweep may not start at an absolute phase of 0 degrees. Settings made for phase control do not apply to an open loop sweep.

  **Note:** Open loop mode works in AUTO mode using the [Power and Attenuators dialog](#). Therefore, the source does not have to be set to ON.

- **<rec/rec>, reference port (Controlled)** - Phase and power is measured and iterated to within the specified tolerance. The receivers, reference port, and iteration properties are specified below.

- **Reference** is displayed to indicate the specified reference port. You can also perform receiver leveling on the reference source. [Learn how](#).

**Reference Port**  Select a source port to be used as a phase reference for the controlled port.

The two internal VNA sources are available ONLY at specific ports. These choices are limited for you on the Phase Control dialog. For example on a 4-port PNA-X, the possible port pairings are: 1/3, 1/4, 2/3, or 2/4. Port 1 can NOT be paired with Port 2, and Port 3 can NOT be paired with Port 4. [Learn more about these limitations](#).

**Ext Source Port**  Selects the source port that the external source is routed through. Displayed when an external source is selected as the active port.

**Control Parameter**  Select the receivers to be used to measure the phase and power of the sources.

The swept phase or phase offset will be the difference between these two receivers. Use either standard notation (R1, A) or logical receiver notation (a1,b1). [Learn more](#).

Select the receivers based on your application. You are responsible to make sure that your DUT configuration routes the signals of interest to the correct receivers. Otherwise, the phase will not be properly controlled. For example, if you select a configuration where Port 4 is the controlled source (measured by a4) and Port 1 is the reference source (measured by b4), both Port 4 and Port 1 must be connected and measured by the Port 4 (a4 and b4) receivers. This would typically be at your DUT output. See the [Active Load Pull Example](#) below.

**Phase Leveling Tolerance**  When consecutive phase measurements of the same data point are within this value of each other, then the phase measurement is considered settled.

**Leveling Max Iterations**  Sets the maximum number of background phase measurements to perform in order to achieve settling. If the phase is not sufficiently settled after these measurements, then the closest value is used.
**Power Leveling Tolerance** The source is considered leveled when each stimulus data point has achieved the power level +/- (plus or minus) this tolerance value.

**Leveling IFBW** Available only for VNA receivers. By default, the IFBW for the leveling sweeps is set to 100 kHz. [Learn more about IFBW.](#)

**Power Ratio** Power ratio value. This is the power offset between the two receivers in the control parameter. Displayed for all sweep types except **Power Sweep**.

**Start/Stop Power** Sets the start and stop power ratio values during a power sweep. Displayed when the **Sweep Type** is set to **Power Sweep**. This is the power offset between the two receivers in the control parameter.

---

**Phase Control Setup** dialog box help

![Phase Control Setup dialog box](image)

**Select a port to set its phase control configuration** This is the 'controlled' port.

**Referenced to** Select a source port to be used as a phase reference for the controlled port.

The two internal VNA sources are available ONLY at specific ports. These choices are limited for you on the Phase Control dialog. For example on a 4-port PNA-X, the possible port pairings are: 1/3, 1/4, 2/3, or 2/4. Port 1 can NOT be paired with Port 2, and Port 3 can NOT be paired with Port 4. [Learn more about these limitations.](#)

**Control Parameter** Select the receivers to be used to measure the phase and power of the sources.

- The LEFT receiver (a1 in the above image) measures the controlled source.
- The RIGHT receiver (a3 in the above image) measures the reference source.
The swept phase or phase offset will be the difference between these two receivers. Use either standard notation (R1, A) or logical receiver notation (a1,b1). Learn more.

Select the receivers based on your application. You are responsible to make sure that your DUT configuration routes the signals of interest to the correct receivers. Otherwise, the phase will not be properly controlled. For example, if you select a configuration where Port 4 is the controlled source (measured by a4) and Port 1 is the reference source (measured by b4), both Port 4 and Port 1 must be connected and measured by the Port 4 (a4 and b4) receivers. This would typically be at your DUT output. See the Active Load Pull Example below.

**Background Sweep Properties**

Background sweeps are phase and power measurements that are made, but the results are not displayed. For each data point, when subsequent measurements are within the specified tolerance, that point is considered settled. If consecutive phase or power measurements of the same data point are NOT within the specified tolerance before the Max Iterations is reached, then one of the following messages are displayed:

- Phase leveling warning: phase not settled.
- Phase leveling warning: power not settled.
- Phase leveling warning: phase and power not settled.

**Apply Settings to All Ports** When checked, the specified settings are used for all background sweeps for all phase-controlled ports. When cleared, the following three settings are specified independently for each port pair.

**Use Leveling IFBW** Available only for VNA receivers. By default, the IFBW for the leveling sweeps is set to 100 kHz. Learn more about IFBW.

**Tolerance** When consecutive phase measurements of the same data point are within this value of each other, then the phase measurement is considered settled.

**Max Iterations** Sets the maximum number of background phase measurements to perform in order to achieve settling. If the phase is not sufficiently settled after these measurements, then the closest value is used.

**Calibration and Phase Control**

Calibrate only those ports that are used for a phase control measurement. For example, if using ports 1 and 3 for phase control, then do NOT calibrate all four ports. If other ports are calibrated, then even ports 1 and 3 may not yield acceptable results.
Important

During the Guided Calibration, at the Select Ports dialog, check the Calibrate source and receiver power checkbox. Otherwise, the calibration may not be accurate.

After performing calibration, the phase is aligned and the power is accurate at the calibration plane.

Active Load Control Example (4-port PNA-X)

See setup using a 2-port 2-source PNA-X.

1. Select the Phase Control parameters in the above Phase Control and Phase Control Setup dialogs.

2. Setup a measurement with the same receivers that are selected on the Phase Control Setup dialog. In this example, on the 'Receivers' measurement tab, select \textit{a4/ b4} as in the following image:
3. Select **Format**, then either **Phase** or **Smith** Chart.

![Smith Chart](image)

4. If you continually see Phase leveling warning: **power** not settled, then on the **Power and Attenuators dialog**, clear Port Powers Coupled (independent port power), then add attenuation to the controlled port. This happens because of additional power being measured from both sources.

![Power and Attenuators dialog](image)

**Active Load Control with 2-Port 2-Source PNA-X Models**

1. Connect a rear-panel jumpers cable from J8 to J1.

2. On the Phase Control Setup dialog, select **Port 2** as the controlled source and **Port1 Src 2** as the reference source.

3. On the **Path Configuration dialog**, on the Configuration box, select **2-port dual source**.

4. Select the Control Parameter (receivers) **a2 / b2**.
Phase-Coherent Measurements

Coherent phase means that the measurement does NOT have random phase. Coherent-phase measurements require the comparison of phase between two sources that are sweeping simultaneously. It also requires that two receivers are measuring the two sources simultaneously.

This capability is available in the PNA-X using several applications:

- iTMSA (Opt S93460A/B) provides phase-coherent Balanced measurements.
- Phase Control (Opt S93088A/B) provides phase-coherent CW or Phase Sweep measurements.
- **R/R measurements** can provide simple phase-coherent measurements.

**How to make phase-coherent R/R (R over R) measurements**

To make a phase-coherent R/R measurements, both sources must be ON simultaneously, and the receiver measurements must be made on the same sweep. On a 4-port PNA-X, the two sources are NOT available at all VNA ports simultaneously. It is important to **learn these restrictions**. Also, both receiver measurements must be made on the same sweep.

1. Create a ratioed receiver measurement using two reference receivers. **Learn how**. For example, you might create an R1/R3 measurement, specifying the source port as either 1 or 3.

2. The source port that is selected above is turned on automatically. The other source port (port 3 in this case) must be turned ON manually using the **Power and Attenuators dialog**.

3. Select a phase format. **Learn how**.
Customize the Analyzer Screen

You can customize your analyzer screen by showing or hiding the following display elements. All of these selections are made from the **Response > Display** menu.

- Windows (Separate topic)
- Display Labels
  - Trace Status
  - Y-axis Labels
  - X-axis Labels
  - Trace Annotation for All Traces
- Marker Display (Separate topic)
- Tables
- Toolbars
  - Softkey
  - Hardkey
  - Port Extension
  - Transform
  - Marker
  - Cal Set Viewer
  - Title Bars
  - Active Entry
  - Status bars
  - System Date and Time
- Display Colors (Separate topic)
- Grid: SOLID | Dotted
Labels

You can display different labels for traces status, Y-axis and X-axis labels.

**How to display labels**

**Using Hardkey /SoftTab /Softkey**

1. Press **Display > Display Setup > Customize Display...**

2. Select **Labels** tab.

OR

1. Press **Marker > Marker Setup > Marker Display...**

2. Select **Labels** tab.

**Using a mouse**

1. Right click on any window area.

2. Click **Customize Display...**

3. Select **Labels** tab.
Show Labels

Trace Status

Trace status is annotated at the top of each window.

The highlighted trace number indicates **Active Trace**.

Click the title to select a trace.

Trace Status shows the following:

- Trace number (Tr x). This is the trace number of the channel; NOT the window trace number which is used in many programming commands.
- Measurement parameter. This can be replaced with a custom Trace Title.
- Format
- Scaling factor
- Reference level

How to show/hide Trace Status.

**Y-axis Labels**

"**Y-axis Labels**" - allows user to show or hide the y-axis labels.

How to show/hide Y-axis Labels.

**X-axis Labels**

"**X-axis Freq Resolution**" - allows user to choose the resolution of the frequency display. The pull down selects: 6-digit, GHz, MHz, kHz, Hz. It shows 1Hz resolution, but only shows significant digits.
How to show/hide X-axis Labels.

Trace Annotation for All Traces
Select to display trace annotation for all traces.

Grid & Tables

How to set VNA Grid and display tables.
Each window can display only one table at a time.

Using Hardkey / SoftTab / Softkey

1. Press Display > Display Setup > Customize Display...
2. Select Grid tab.

OR

1. Press Marker > Marker Setup > Marker Display...
2. Select Grid tab.

Using a mouse

1. Right click on any window area.
2. Click Customize Display...
3. Select Grid tab.
Grid Lines: Solid | Dotted - Set whether to display ALL open window grid lines in solid or dotted lines. The selected setting is shown in CAPS. Once set, new windows are created using this setting. Grid lines return to SOLID when the VNA is Preset.

Set the color of the grid using Display Colors.

How to display grid settings

Y-axis Divisions - Set the desired rows of Y-axis, it can shows 2 to 30.

Show Table

None - Turn OFF the table.
Marker Table

You can display a table of marker settings. These settings include the:

- Marker number
- Marker reference (for delta measurements)
- Frequency
- Time and Distance (for Time Domain measurements)
- Response

Learn more about Markers

Limit Line Table

You can display, set up, and modify a table of limit test settings. These include:

- Type (MIN, MAX, or OFF)
- Beginning and ending stimulus values
- Beginning and ending response values

Learn more about Limit Lines.

Ripple Table

You can display, set up, and modify a table of maximum ripple limit over frequency range settings. These include:

- Type (ON or OFF)
- Beginning stimulus value
- Ending stimulus value
- Maximum ripple

Learn more about Ripple limits.
Segment Sweep Table

You can display, set up, and modify a table of segment sweep settings. These include:

- State (On/Off)
- Start and Stop frequencies
- Number of Points
- IF Bandwidth (if independent levels)
- Power Level (if independent levels)
- Sweep Time (if independent levels)

Learn more about Segment Sweep.

Distortion Table

You can display, set up, and modify a table of modulation distortion measurement parameters. Each row represents a measurement band. Each column represents a measurement parameter.

Learn more about the Distortion Table.

Spurious, Integrated Noise, Spot Noise Tables

You can display, set up, and modify a table of phase noise measurement parameters.

Learn more about the Phase Noise.
### Toolbars

You can display different toolbars to allow you to easily set up and modify measurements.

#### How to display Toolbars

**Using Hardkey /SoftTab /Softkey**

1. Press **Display > Display Setup > Customize Display...**
2. Select **Toolbars** tab.

**OR**

1. Press **Marker > Marker Setup > Marker Display...**
2. Select **Toolbars** tab.

#### Using a mouse

1. Right click on any window area.
2. Click **Customize Display...**
3. Select **Toolbars** tab.
Show Toolbar

Note: There is also a Cal Set toolbar available for Monitoring Error Terms

Softkey
Softkey is a combination of softkeys and SoftTabs. Softkeys are automatically turned ON when one of the 'function' hardkeys is pressed. This setting allows you to turn the softkeys OFF to show more measurement space on the screen. The softkeys will reappear when another function hardkey is pressed.

**Hardkey**
These keys also known as Front Keys, perform interface operations that are equivalent to those of keys in the INSTRUMENT keys, RESPONSE keys, STIMULUS keys and UTILITY keys on the front panel of VNA. Learn more.

**Port Extensions Toolbar**
The Port Extension toolbar allows you to set Port Extensions while viewing the measurement trace. Learn more about Port Extensions.

**Transform (Time Domain) Toolbar**

The Time Domain toolbar allows you to do the following:

- Turn **Transform** and **Gating** ON/OFF.
- Change the Start/Stop times for both Transform and Gating.
- **More...** - Launches the Time Domain Transform dialog box.
- ? - Display the help file.
- X - Closes the toolbar.

**Markers Toolbar**

The markers toolbar allows you to set up and modify markers. It shows:

- Marker number
- Stimulation value
- Marker functions:
  - Delta
  - Max/Min
  - Start/Stop
  - Center/Span
Tip: To use the Front Panel Knob to change marker position, first click the Stimulus field of the marker toolbar and then turn the knob.

Learn more about Markers

Cal Set Viewer Toolbar

Learn more about Cal Set Viewer.

All Off (NOT on softkeys)

This allows you to hide all toolbars with a single selection. NOT available on sofkeys.

Other Bars

Main Title

The Main Title shows the title of VNA window and Minimize / Maximize icons.

- Checked - Title bars for all VNA window are shown.
- Cleared - Title bars for all VNA window are hidden. This allows more room to display measurement results.

Active Entry Toolbar

When used with softkeys, this area allows numeric values to be entered for settings. From the keyboard, enter G for Giga, M for Mega or milli, K for kilo and so forth.

Status Bar

2 Bars
When enabled, the status bar is displayed along the bottom of the VNA screen. The primary status bar shows the following:

**Tip:** Right-click on many of these items in the status bar for quick access to settings.

- Active trace
- Active channel
- Trigger source
- Channel Trigger State (Hold, Single, Continuous)
- IF Bandwidth
- Error correction for the active trace to the Basic cal, Smart cal and Calibrate All Channels.
  - F: Full Port Calibration, R: Response Calibration, -: Nothing
- Reference (Internal/ External )
- Source Power Calibration
- Service (shows when the measurement result is not guaranteed such as hardware failure. The advance mode in TDR also indicates the Service.)
- RF power
- Display Update
- Error messages
- GPIB status : Local (LCL), Remote Talker Listener (RMT), or System Controller (CTL).
- System Date and Time - Can be set ON or OFF. How to show/hide the VNA clock.
- Trace Deviation

**Note:** A second level status bar appears when using External Test Set Control or Interface control.

The status bar state (ON or OFF) will not change when the VNA is Preset.

**Clock (System Date and Time)**
The VNA system date and time can be shown in the far right corner of the status bar. The format is: year-month-day hr:min and can NOT be changed.

To hide the clock, right click the mouse on the clock and then click **Hide Clock**.

Learn how to set the VNA time settings.

**Display Sheet Tabs**

**Top** - Display sheet tabs above display.

**Bottom** - Display sheet tabs below display.

---

**Tools**

**How to set Tools settings**

<table>
<thead>
<tr>
<th>Using Hardkey / SoftTab / Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Display &gt; Display Setup &gt; Customize Display</strong>...</td>
<td>1. Right click on any window area.</td>
</tr>
<tr>
<td>2. Select <strong>Tools</strong> tab.</td>
<td>2. Click <strong>Customize Display</strong>... .</td>
</tr>
</tbody>
</table>

OR

1. Press **Marker > Marker Setup > Marker Display**... .

2. Select **Tools** tab.

---

**Programming Commands**

**Tools** Dialog Box Help
Tools function to create a shortcut icon to display on Active Entry or Softkey Toolbar. The maximum icons can display on Active Entry is 9, while Softkey Toolbar is 12.

**New Window** - Create a new window.

**New Channel** - Create a new channel on active window.

**New Channel + Window** - Create a new trace and channel to a new window.

**New Trace** - Create a new trace on active window.
New Trace + Window - Create a new trace to a new window, but the channel is remain.

New Sheet - Create a new sheet.

New Trace + Sheet - Create a new trace to a new sheet, but the channel is remain.

New Channel + Sheet - Create a new trace and channel to a new sheet.

Home - Display VNA Home softkeys.

Favorite - Set favorite application. To Add a Favorite, press and hold any softkey for three seconds and select the desired Favorite number (Favorite 1 to 3).

Help - Shows Help file.

Show Task Bar - Shows Window bar.

Preference - Display preference dialog box.
Tile Windows

Manages Files - Use to manage the saved files in the "D:" drive folder. Learn more.

Minimize Application - Restore VNA screen. Learn more.

About Network Analyzer - Display About Network Analyzer dialog box.

Undo - Recover to previous version. Learn more.

Redo - Set to latest version. Learn more.

Screenshot - Save screen figure to "D:" drive (D:\).

Delete - Delete the active window.
Colors

How to set colors

Using Hardkey / SoftTab / Softkey

1. Press Display > Display Setup > Customize Display...
2. Select Colors tab.

Using a mouse

1. Right click on any window area.
2. Click Customize Display...
3. Select Colors tab.

Colors Dialog Box Help

Menu Color Theme - Select color theme

Display Colors... - See Display Colors

Print Colors... - See Print Preview
**Window Title**

You can create and display a title for each **window**.

- The limit is set by the number of windows that are displayed.
- The title (My Window) is annotated in the upper-left of the window as follows:

```
Tr 1  S11 LogM 10.00dB/ 0.00dB
   Window 1
```

**How to enter a Window Title**

**Using Hardkey /SoftTab /Softkey**

1. Press **Display > Window Setup > Window Title...**

**Using a mouse**

1. Move a cursor in the grid and then right click.
2. Select **Title...**

**Window Title Dialog Box Help**

1. Click **Enable**, then type the window title. Click **Keyboard** to type with a mouse.
2. To remove the window title, clear the **Enable** checkbox or delete the text from the dialog entry.

**Trace Title**

A Trace Title overwrites the Measurement Parameter in the Trace Status area, the Status Bar and hardcopy prints.

- This title has priority over Equation Editor titles.
The practical limit is about 70 characters if there is only one trace.

Spaces are accepted but not displayed; use underscores.

The title is annotated as follows:

![Trace Title Example](image)

**How to enter a Trace Title**

**Using Hardkey / SoftTab / Softkey**

1. Press Trace > Trace Setup > Trace Title...

**Using a mouse**

1. Move a cursor in the grid and then right click.
   
   2. Select Trace Title...

**Trace Title Dialog Box Help**

1. Click **Enable**, then type the window title. Click **Keyboard** to type with a mouse.

2. To remove the window title, clear the **Enable** checkbox or delete the text from the dialog entry.

**Frequency/Stimulus**

![Frequency/Stimulus Example](image)

Frequency/stimulus information is displayed at the bottom of each window on the screen. It shows:

- Channel number
- Start value
- Stop value
Minimize Application

The Network Analyzer application can be minimized to show the desktop and Windows taskbar.

1. Click **System > Main > Minimize Application**.

To restore the VNA application, double-click the VNA application on the desktop.
Copy Channels

Copy channels allows you to make a duplicate channel of the same Measurement Class and with the same stimulus conditions as an existing channel.

- Why Copy Channels
- How to Copy Channels
- List of Channel Settings

Note: Copy Channels CAN be used with PNA Applications, such as FCA, Gain Compression, or Noise Figure.

Other Setup Measurements Topics

Why Copy Channels

Copy channel settings if you need to create several channels that have slightly different settings.

For example, if you have an amplifier that you want to characterize over a frequency span with several different input power levels.

Follow these steps:

1. Create one measurement with your optimized channel settings.
2. Copy that channel to new channels.
3. Change the power level on the new channels.

The alternative to using Copy Channels is to create new default measurements on new channels. Then change every channel setting to your new requirement. This is very time consuming and thus shows the benefit of the Copy Channels feature.
How to Copy Channels

Using Hardkey/SoftTab/Softkey

1. Press **Channel > Channel Setup > Copy Channel.**

2. Click **Copy to Active Window/Copy to New Window/Copy Channel...**

![Programming Commands](image)

**Copy Channel** dialog box help

Copies an existing channel's settings to another channel. Measurement traces from the source channel are **NOT** copied.

**Copy channel** (also known as 'Source' channel): Select a channel to copy.

**to** (also known as 'Destination' channel): Scroll to select a channel to copy settings to. Compatible channel numbers that are currently being used are highlighted. They can be selected and overwritten.

The following are compatible destination channels:

- A channel that does not yet exist. The new channel is created with the channel's default measurement.

- A channel of the **same Measurement Class** as the source. The existing measurements remain on the destination channel.

- A channel of any Measurement Class that contains no measurements. Again, the destination channel is created with the channel's default measurement.

**Notes:**

- You can copy channel settings to ONLY one new or existing channel. Repeat this operation to copy to more than one channel.
The source channel is ALWAYS copied to the Active window. If you want the destination channel in a separate window, first create a compatible new measurement in a new window. Then make sure it is the Active window before you copy the channel into it.

The measurement in the destination channel becomes the active measurement.

For example:

1. **Source** channel 1: Standard S21 measurement
2. **Destination** NEW channel 2
3. **Result**: Source channel 1, S21 Measurement AND channel 2, S11 measurement. Both with same stimulus settings and in the same window. Channel 2, S11 measurement is the active measurement.

For more information see [Traces, Channels, and Windows](#)

**List of Channel Settings**

- Frequency Span
- Power
- Cal Set usage
- Source Power Cal data
- IF Bandwidth
- Number of Points
- Sweep Settings
- Average
- Trigger (some settings)
DC Source Control

When a DC Source (power supply) is configured as an external device, the new DC source can be controlled from the VNA using this dialog. Internal DC Sources are also controlled from this dialog.

- How to start the DC Source Control dialog
- The DC Source Control dialog box
- The DC Limits dialog box

See Also

Configure the DC Source as an External Device
Internal DC Sources - ADC Measurements

Other Setup Measurements Topics

How to start the DC Source Control dialog

Using Hardkey/SoftTab/Softkey

1. Press **Sweep > Source Control > DC Source**....

Using a mouse

1. Click **Stimulus**.
2. Select **Sweep**.
3. Select **Source Control**.
4. Select **DC Source**.

DC Source Control dialog box help
Note: The 9-pin PWR I/O (Power I/O) D connector on the rear-panel replaces much of the functionality of the AUX I/O connector on older VNA models. The Power I/O voltages can be set using the following methods:

- CONTrol:AUXiliary:OUTPut:VOLTage or put_OutputVoltage Method (no GUI equivalent, global scoped, and settings not saved as part of the instrument state)
- SOURce:DC:START and SOURce:DC:STOP (DC Source dialog is the GUI equivalent, channel scoped, and settings saved as part of the instrument state)
- Interface Control dialog (no remote equivalent, channel scoped, and settings saved as part of the instrument state)

To avoid unexpected behavior, choose one method only to set the Power I/O voltages.

Name  Lists the names of the configured DC Sources. In the above image:

- **AuxOut1** and **AuxOut2** are internal VNA DC sources that are available
  - Rear-panel Power I/O connector (Pins 3 and 4). Learn more.
- **MyDCSupply** is the name of an external DC Source. Learn how to setup and configure an External DC Source and DC Meter.

State  Set the state of the DC source.

- **ON**  DC Source is always ON.
- **OFF**  DC source is always OFF.
- **Per Port**  The Name selection for that DC source expands to allow an Port <n> / N/A setting for each
When the RF source for that port is ON, then the DC source for port is also ON. Select 'N/A' to turn the DC Source OFF for that port.

**Start / Stop DC**  Set the start and stop voltages of the DC source. The VNA will step the voltage of the DC source from Start to Stop in increments = (Stop - Start)/Number of data points.

**Buttons**

**Limits**  Click to start the DC Limits dialog.

**Receiver Leveling**  For future use.

---

**DC Limits dialog box help**

Select the Minimum and Maximum voltages to which the specified DC sources can be set by the VNA. When the DC source level exceeds the limit, DC source is turned off and the measurement sweep stops.
ADC Measurements

The PNA is equipped with two ADC (Analog to Digital Converter) inputs on the rear-panel Power I/O connector (Pins 7 and 8). These ADC inputs can be used as measurement receivers and display measurements on the VNA screen.

- Analog Inputs (AI1 and AI2) can be used for measuring from -10V to +10V. These inputs can be considered auxiliary receivers and used in a similar way as S-Parameter receivers.
- Analog Output Sense inputs (AOS1 and AOS2) can be used to measure the corresponding DAC outputs.
- Analog Ground input (AIG) can be used to measure the instruments analog ground (PNA-X only).

Two DAC outputs are supplied on the Power I/O connector (Pins 3 and 4). These DAC outputs are controlled from the DC Source Control dialog.

Learn how to configure an external DC Source or DC Meter.

**New Trace (ADC) dialog box help**

**Note:** Sweep speed slows dramatically when measuring more than two ADC receivers.

On the **New Trace dialog**, click the **Receivers** tab.

**Activate** - check any empty line to create a trace.

**Numerator** - select from the following:

- **AIx** - Input x:  x is Input No.
- **AOSx** - Output sense x:  x is Output No.
- **AIG** - Analog ground  (PNA-X only)

**Denominator** - NOT available (ONLY unratioed measurements)

**Source Port** - The ADC receiver is measured when the specified source port is sweeping. Select None to always measure the ADC receiver.
ADC receiver traces are labeled as shown in the following images:

- The ADC1 input is being measured, with 2 as the source port.
- The Y axis is U (unitless).
- The default trace format is Real (linear).

ADC Traces and other useful VNA functions

Although most VNA functions work with ADC traces, the following may be especially useful.

- **Equation Editor** can be used with the trace data. Although the VNA ADC is measuring voltage (-10V to +10V range in 14 bits), by using a trace formula, this voltage can represent other types of measurement parameters (such as current, temperature, or a scaled voltage). See PAE example.

- **Trace averaging** and **Trace Smoothing** can be used to remove trace noise.

- **Dwell time** can be used to allow for settling.

VNA Functions Not Supported

- Calibration for ADC receivers is NOT supported.

- Not supported in **Noise Figure application**
Integrated Pulse Measurements

The Pulse Setup dialogs shown in this topic are now integrated in the VNA firmware and are available with S9x026A/B or Opt H08.

**Note:** For information about pulsed measurements using a PNA-X, refer to Active-Device Characterization in Pulsed Operation Using the PNA-X.

Previously, setup was performed with the Narrowband or Wideband pulse programs. With the appropriate hardware/software options (Opt 021, 022, S93025A/B) these commands are still available without Opt S93026A/B or H08. Learn more about VNA Options.

External pulse generators can be used along with the VNA internal pulse generators. Learn more.

In this topic

- Pulse Setup
- Pulse Setup (Modulation Distortion Channel)
- Pulse Generator Setup
- Pulse Trigger Tab
- Pulse Gens and IF Block Diagram
- Calibration in Pulse

**See Also (separate topics)**

- Configure and Use External Pulse Generators
- IF Path Configuration
- Programming commands
- Narrowband Pulsed Application (Opt H08)
- WB Pulsed App
- See Swept IMD note regarding IF Filter setting

**App Note:** Active-Device Characterization in Pulsed Operation Using the PNA-X (1408-21)
How to start the Pulse Setup dialog

Using Hardkey /SoftTab/Softkey

1. Press **Sweep > Source Control > Pulse Setup...**

Using a mouse

1. Click **Stimulus**
2. Select **Sweep**
3. Select **Sweep Control**
4. Select **Pulse Setup...**

Pulse Setup dialog box help

Note: The M937xA does not support this function.

The Basic controls allow simple pulse measurements using the default (Autoselect) settings in the Advanced section of the dialog.

Pulsed measurements are performed in a Standard channel. See Measurement Class. However, several VNA measurement settings are controlled by the Pulse setup, such as sweep type, number of points, and so forth.
**Pulse Measurement**

**Off** - Source and Receivers are NOT pulsed

**Standard Pulse** - With pulsed RF, the VNA can be configured to sweep in frequency, power sweep, and CW time.

- To make 'Point-in-Pulse' measurements, narrow the receiver pulse width and enter delay.
- To make 'Pulse-to-Pulse' measurements, disable **Autoselect Pulse Detection Method**, then select **Wideband**. select **Stimulus**, then **Sweep**, then **Sweep type = CW** in the VNA menu.

**Pulse Profile** - Pulse profile measurements provides a time domain (CW frequency) view of the pulse envelope. Profiling is performed using a measurement technique that "walks" a narrow receiver "snapshot" across the width of the pulse. This is analogous to using a camera to take many small snapshots of a wide image, then piecing them together to form a single, panoramic view.

![Pulse Profile measurement using default settings and R1 receiver.](image)

- Pulse Profiling can be performed using ratioed or unratioed measurements. You can preview the pulse on port 1 by using an R1 receiver measurement.
- Pulse Profiling is performed at a single CW frequency in either Narrowband or Wideband mode.
- To select the CW Frequency, click **Stimulus**, then **Sweep Type**.
- In **Narrowband** mode, the delay increment value, which is responsible for "walking" the receiver snapshot across the pulse, is selected by the VNA and is accessible only with a programming command.
- In **Wideband** mode, the receiver is walked across the pulse by making a sequence of closely-spaced measurements in real-time.
In SW Gating mode, software gating sensitivity is improved. When unchecked, all data outside the measurement band is zeroed. SW Gating mode is used for troubleshooting purposes.

**Pulse Timing**

**Pulse Width** - Sets the width of the source pulse. See measurement timing to learn how to control the receiver width and delay.

**Pulse Period**  The time to make one complete pulse.

**Pulse Frequency (PRF)** The reciprocal of Period (1/Period). See Internal Pulse Generators to learn more.

By default, these settings configure Pulse Gen 1 to drive Source Modulators 1 and 2. This can be changed from the Advanced Settings Pulse Generator Setup dialog.

------- Advanced Settings -------

The following settings allow maximum control of a Pulse measurement.

**Note:** When the "Auto" check boxes are cleared, it is possible to configure settings to make an invalid measurement.

**Properties**

**Autoselect pulse detection method** - check to automatically switch between Narrowband and Wideband based on the Pulse Width.

In Standard Pulse:

- **Wideband** - used when the (source) Pulse Width is WIDER than the fastest receiver acquisition time. This allows the receiver to measure all pulse ON time - no pulse OFF time. The VNA will select Wideband whenever possible.

- **Narrowband** - used when the (source) Pulse Width is NARROWER than the fastest receiver acquisition time (267 ns). This measurement requires a spectral nulling technique to measure the pulse response through the DUT.

In Pulse Profile:

- **Wideband** - used when the (source) Pulse Width is greater than 1.600 us. This allows the receiver make several sequential measurements to measure the entire pulse.

- **Narrowband** - used when the (source) Pulse Width is less than or equal to 1.600 us.

**Autoselect IF Path Gain and Loss** - For future use.
**IF Path** - Click to launch the IF Path dialog.

**Optimize Pulse Frequency** - Automatically selects the Pulse Frequency and Pulse Period.

- In Narrowband, the pulse frequency is adjusted slightly to get the best spectral-nulling filtering possible.
- In Wideband, this checkbox is ignored.

**Autoselect Profile Sweep Time** - In Pulse Profile mode, adjusts the default X-axis start time to zero and the stop time double the Pulse Width. This allows you to see one complete pulse. If unchecked, the Sweep Time will not be changed.

To adjust the X-axis manually, click OK to close the dialog. Then press Sweep > Main, then change the Start Time and Stop Time.

**Sweep Time** - Sets the time the analyzer takes to complete one sweep.

**Number of Points** - Sets the number of data points for the measurement.

**IFBW** - Select the IFBW for the measurement.

- In Narrowband, an IFBW as close as possible to the entered value will be used.
- In Wideband, this setting determines the receiver acquisition time - approximately 1/IFBW.

**Measurement Timing**

**Port n, Rcvr** - Used as RF Source Modulation Drive.

- **Width** - source pulse width.
- **Delay** - source pulse delay relative to the pulse generator clock.
- **Pulse Gen** - Pulse generator used to modulate the source. Select CW to have NO source modulation.

The receiver settings in this table change depending on whether the VNA is in Narrowband or Wideband mode.

- In Narrowband, for each IF receiver path, configure the Pulse Width, Delay, and Pulse Generator to be used to drive the receiver gate.
- In Wideband, all receiver paths are the same.
**Pulse Trigger Source**  Refer to Block Diagram  Choose from:

- **Internal** - Default setting. The pulse generator is triggered by an internal pulse clock.
- **External** - An external pulse generator is selected but not controlled by the VNA. Use this setting to make manual pulse measurements.
- **<External Pulse Gen name>** - Available when a 81110A is configured as an External Device and **Master Mode** is checked on the pulse generator properties dialog. See how to make this setting using SCPI and COM.

**Primary Clock**

The Primary Clock is controlled by the internal or external pulse generator and is the primary pulse clock. The **Internal** and **External** selections are not the same as the Trigger Source **Internal** and **External** selections found in the Pulse Trigger tab of the Trigger dialog. However, they are inter-related as follows:

<table>
<thead>
<tr>
<th>Primary Clock</th>
<th>Trigger Source (from Pulse Trigger tab of Trigger dialog)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary pulse clock controlled by selected pulse generator.</td>
<td>Trigger source for the internal pulse generator.</td>
</tr>
<tr>
<td><strong>Internal</strong></td>
<td></td>
</tr>
<tr>
<td>- Primary pulse clock is controlled by the internal pulse generator.</td>
<td>- Trigger is self-generated in the internal pulse generator.</td>
</tr>
<tr>
<td>- PRF setting is applied to the internal pulse generator.</td>
<td>- The trigger source for the internal pulse generator trigger is set to Internal and is reflected in the Trigger Source selection in the Pulse Trigger tab of the Trigger dialog.</td>
</tr>
<tr>
<td><strong>External</strong></td>
<td></td>
</tr>
<tr>
<td>- The primary pulse clock is an uncontrolled external pulse generator.</td>
<td>- Trigger comes from the PULSE I/O connector (PulseSyncIn) on the rear panel.</td>
</tr>
<tr>
<td>- PRF settings are determined by the external pulse generator, which the VNA firmware does not control. The user must set up the external system clock. Entries</td>
<td>- The trigger source for the internal pulse generator</td>
</tr>
</tbody>
</table>
Autoselect Width and Delay - When checked, for Wideband mode and Pulse Gen = Pulse Trigger, the default setting for the receiver is adjusted to approximately 75% of the source pulse width, with 20% delay. This leaves approximately 5% of the source pulse ON after acquisition is complete.

When checked for Narrowband mode and Pulse Gen <> CW, then Delay and Width matches the RF Source.

Autoselect Pulse Generators - When checked:

- Pulse1 is selected for Modulator Drive.
- Pulse2 is CW (OFF).
- For Wideband, Pulse Trigger is selected to gate the ADC.
- For Narrowband, Pulse2 is selected.

Pulse Generators... Click to launch the Pulse Generators Setup dialog.
Pulse Measurement

**Off** - Source and Receivers are NOT pulsed

**Standard Pulse** - With pulsed RF, the VNA can be configured to sweep in frequency, power sweep, and CW time.

Pulse Timing

**RF Pulse Width** - Sets the width of the RF Source pulse.

**Pulse Period** pulldown- Allows the selection of the following:

- **Pulse Period** - The time to make one complete pulse.
- **Pulse Frequency** - The reciprocal of Period (1/Period).
- **Pulse Duty Cycle** - **Pulse Width** divided by the **Pulse Period**.

Labels

- **Sweep** - Complete cycle time of measurement including background sweeps.
- **Acquire** - The ADC acquisition time required to measure the data for a single FFT.
- **Pulse/Swp** - Number of pulses which will occur during the sweep, including pulses used for background sweeps.
Pulse Details

**Generator** - Pulse generator outputs in numerical order.

**Device** - Indicates the device being controlled by the pulse generator output.

- **Pulse0** - Always set to **Receiver**. Sets the amount of time to wait before triggering the ADC to begin acquisition and is always selected for Pulse0 and cannot be changed. Pulse0 adds (ADC Delay) + (Modulator Delay). The ADC will begin measuring data 250 ns before the rising edge of Pulse0. This delay is indicated in the **Fixed ADC Delay = 250 ns** annotation.

- **Pulse1** through **Pulse4** - Pulse outputs can be set to the following:
  - **RF** - Selecting **RF** indicates that the pulse signal is used to drive the RF modulator. Only one pulse generator output can be used to drive an RF source. If you try to set more than one pulse generator output to **RF**, then the other one will be set to **User N** (where "N" is the pulse generator number).
  - **User 1, User 2, User 3, User 4** - Labels for user convenience. These labels do not connect the pulse generator to any specific hardware. These selections may be used to control a DUT, DC biases, or other signals.
  - **Pulse4 ADC Activity** - (Pulse4 only) Pulse4 can also be set to ADC. This selection outputs a signal on Pulse4 when the ADC is active. This is the same as Pulse4 Output Indicates ADC Activity on the Pulse Generators Setup dialog. If ADC is selected for Pulse4, then the pulse width and delay entries are grayed out because Pulse4 is no longer a pulse output.

**Width** - RF Source pulse width. This setting is the same as **Pulse Width** under RF Pulse.

**Delay** - If the **Offset Pulses using ADC Delay** check box is disabled, then the delays are the pulse delays relative to the trigger. If the **Offset Pulses using ADC Delay** check box is enabled, then the delays are defined for the pulses relative to each other. In this case, you can enter negative delays, and the delay from the trigger will be adjusted to correct for these values.

**Invert** - Check to cause the pulse ON time to be active low and OFF be active high.

**Enable** Check to enable individual pulse generators.

**Autoselect Receiver Timing** - Computes the width to 80% of the RF Source Width and the delay will be 10% of the RF Source Width. With this setting selected, the Width and Delay columns in the table are grayed out.

**Offset Pulses using ADC Delay Check Box**
If checked (default), adds delays to the Pulse Generator:

- Pulse0 adds (ADC Delay) + (Modulator Delay).
- Pulse Output used as the Modulator Drive adds no delay.
- All other Pulse Outputs add Modulator Delay.

**RF Modulator Delay**

- Defines the RF delay of the source modulator. This is the time lag between the pulse drive signal and the actual RF output. This may indicate the lag for either an internal or external source.
- The default is 40 ns, which is the average delay of the internal RF modulators. The internal modulator below 3.2 GHz is slower than the internal modulator above 3.2 GHz. Therefore, the average value is chosen.

**ADC Delay**

- The ADC starts on the rising edge of Pulse0. Due to the data pipeline, the ADC begins measuring data 210 ns before the rising edge of Pulse0 occurred. Since the ADC measurement leads Pulse0, Pulse0 is delayed by this amount of time. This value cannot be changed.

**Timing Example**

Assume Pulse1 is used to modulate the RF signal, all Pulse outputs are enabled, all are set to zero delay, and all are set to the same width. The first timing diagram below is with offset off and the second timing diagram is with offset on.
Buttons

Plot Pulse Timing - Accesses pulse timing display to evaluate the pulse timing setup.

Right-click in the display area to access the following menu:
**Autoscale** - Automatically scales the data to fit vertically within the display grid area.

**Display marker annotation** - Select to display marker annotation in the top-right of the display.

**Show graticule** - Select to display graticules.

**Add marker to:** - Select to add a marker to a displayed pulse trace. When a selection is made, the mouse pointer changes to a "+ ". Click in the display area and the marker will appear. Drag the marker to the desired position. Each time this selection is made, a new marker will be added.

**Copy to Clipboard** - Copies a bitmap of the trace control (Display) to the clipboard. It can then be pasted into any document that accepts bitmaps.

**Print...** - Prints the displayed data.

**Scale properties...** - Accesses the following dialog:
Stimulus - Sets the Begin and End displayed on the X-axis in seconds.

Response - Sets the Reference level in the center of the Y-axis and sets the scale per division.

Pulse Generators - Accesses the Pulse Generators Setup dialog. See below.

**Pulse Generators Setup** dialog box help

This dialog is available with Option S9x025A/B (pulse generators).

To see this dialog, press **Pulse Generators...** on the Pulse Setup dialog.

**Pulse Generators**

Configure the Pulse Generators to be used for your measurement. The pulse 0 is for the receiver. The pulse 1 is for source and P1 output and pulse 2 to 4 are for P2 to P4 output signal, respectively.

- **D** = Delay; the time before each pulse begins
- **W** = Width; the time the pulse is ON
Duty Cycle = W/P

P = Period; one complete pulse cycle

Pulse Frequency (PRF) = 1 / Period

**Important:** If D + W is greater than P, then undefined VNA behavior results. There is NO error message or warning.

**Invert**  Check to cause the pulse ON time to be active low and OFF be active high.

**Enable**  Check to enable individual pulse generators.

**Trigger**  Choose from: (When ONE of these is changed, they ALL change. The internal Pulse Generators can NOT be triggered individually).

- Internal - Pulse generators are triggered by the internal pulse clock.
- External - Pulse generators are triggered by an external pulse generator though Pulse Sync IN.

**Frequency**  - Set the pulse frequency of each generator.

- Pulse Frequency (PRF) = 1 / Period
- P = Period; one complete pulse cycle

**Period**  - Set the period of each generator.

Learn more about the Pulse Generators.

**Pulsed Sources**

Check to enable the required internal source ports.

These are switches 8 and 9 in the Block Diagram

**Important**: When internally modulating the sources, source leveling is automatically set to Open-loop (**ALC Open Loop** box will be checked automatically).

**Modulator Drive**  - Choose the pulse generator to modulate the specified source. Choose from CW (NO pulse), Pulse 1, 2, 3, 4, External.  This is switch 7 Block Diagram.

**Offset Pulses**

**Offset Pulses using ADC Delay Check Box**
If checked (default), adds delays to the Pulse Generator:

- Pulse0 adds (ADC Delay) + (Modulator Delay).
- Pulse Output used as the Modulator Drive adds no delay.
- All other Pulse Outputs add Modulator Delay.

RF Modulator Delay

- Defines the RF delay of the source modulator. This is the time lag between the pulse drive signal and the actual RF output. This may indicate the lag for either an internal or external source.
- The default is 40 ns, which is the average delay of the internal RF modulators. The internal modulator below 3.2 GHz is slower than the internal modulator above 3.2 GHz. Therefore, the average value is chosen.

ADC Delay

- The ADC starts on the rising edge of Pulse0. Due to the data pipeline, the ADC begins measuring data 210 ns before the rising edge of Pulse0 occurred. Since the ADC measurement leads Pulse0, Pulse0 is delayed by this amount of time. This value cannot be changed.

Offset Pulse Example

Assume Pulse1 is used to modulate the RF signal, all Pulse outputs are enabled, all are set to zero delay, and all are set to the same width. The first timing diagram below is with offsets off and the second timing diagram is with offsets on.
**Pulsed Receivers**

**Synchronize ADCs using Pulse Trigger** - Check to enable triggering used to gate the ADC for wideband receiver measurements. This is the same as **Pulse0 Enable**. The Width can NOT be configured.

**Pulse4 Output Indicates** - Check to use an oscilloscope connected to the pulse 4 (pin 13 of the PULSE I/O connector on the rear panel of the VNA) to display when the ADC is making measurements. There are two selections:

- **All ADC Activity** - When selected, all ADC activity can be monitored, including ADC activity that may not be displayed on a trace. An example is background measurements that are used for receiver leveling, but are not actually displayed on a trace.

- **Trace ADC Activity** - When selected, Pulse4 will be active only during measurements that will be displayed on a trace.

**Pulse Trigger...** - Click to start the Pulse Trigger dialog.

**Trigger...** - Accesses the Trigger dialog for setting up triggering. Learn more.

---

**Pulse Trigger Tab** - Trigger dialog box help
To see this dialog, press **Pulse Trigger** on the Pulse Generator Setup dialog or select **Stimulus**, then **Trigger** from the VNA Menu.

**Trigger Source**

Select **Internal** or **External** to provide sync capability for the internal pulse generators.

- **Internal** - The pulse generator is internally triggered and puts out a periodic pulse train with a period defined by the Pulse Generator Setup dialog.

- **External** - The internal pulse generator puts out one set of pulses (P0-P4) per external trigger (Pulse Sync In). All five pulse outputs have unique delay and pulse width settings.

The external trigger input is on the Pulse I/O connector pin 7 (PulseSyncIn). The PulseSyncIn line provides a configurable trigger signal into the Pulse Generators. If the trigger mode is set to "level", and if the trigger is still valid when the first pulse set is finished, another set will be generated. Only one set of pulses is emitted when edge triggering is used.

**Trigger Level/Edge**

Sets the edge or level of the trigger signal to which the internal pulse generators will respond when being externally triggered at the PulseSyncIn pin.
Positive = rising edge; Negative = falling edge.

These selections are available ONLY with DSP version: 4.0 FPGA: 34 or higher. Learn more. Otherwise, the pulse generators respond only to positive, level input trigger signals.

**Receiver synchronization**

**Synchronize ADCs using pulse trigger** - Check to enable triggering used to gate the ADC for wideband receiver measurements. The Width can NOT be configured.

**ADC trigger delay** - Set the amount of time to wait before triggering the ADC to begin acquisition.

**Pulse Trigger Block Diagram**

See complete description at IF Path Configuration

See also Using External Pulse Generators

**Using External Pulse Generators**

Setup the External Pulse Generator as an External Device.

**Calibration in Pulse Mode**
To perform a calibration in pulse mode, first configure and apply the pulse parameters (PRF, Pulse Width, Delays, IF gating, and so forth) **before** calibrating the system. This will ensure the VNA is configured properly during the calibration and measurement.
Global Source

Global Source allows any internal or external source to be set globally and retain its settings even after an instrument preset.

How to access the Global Source dialog

Using Hardkey/SoftTab/Softkey

1. Press **Sweep > Source Control > Global Source...**
   
   or

2. Press **System > System Setup > Preferences...**

3. Press **Global Sources...** button.

Using a mouse

1. Click **Stimulus**.

2. Select **Sweep**.

3. Select **Source Control**.

4. Select **Global Source**.

Global Source dialog box help

**Power On (All Channels)** - Check to enable source power for all channels.

**Global Sources Ignore "Power ON" Setting** - This setting is a preference. Sources set to Global
are not turned off when **Power On (All Channels)** is checked. Users of high power systems often turn off the RF Power to avoid damaging the system or the DUT. However, they may not want to turn off Global Sources being used as the clock signal for synthesizers.

**Name** - Lists the names of the sources.

**Global** - Set the state of the source.

- **OFF**  Do not set as a global source.

- **Global**  Set as a global source and set to a fixed frequency and power level defined in this dialog. Once a source is set to Global, it becomes "locked" and its settings cannot be controlled by any channel settings (frequency, power, RF on/off): It will be unaffected even if sweeps are disabled and if there are no channels defined for the VNA. The Recall feature will not change the Global Source setting. If Global Source is ON and a state saved with Global Source OFF is recalled, then Global Source will remain ON and the state will not be completely recalled and the VNA displays the warning message: **The Recall is incomplete because Global Source is ON.** When the GUI is exited, the Global Sources will be turned OFF.

The following shows Source3 has been set to Global.
**State** - Turns source power on and off.

**Frequency** - Sets source frequency.

**Power** - Sets the power level at the output of the source.
Amplifiers, Mixers, and Band pass filters are common DUTs that are characterized with VNAs. In some cases the setup of these devices can be time consuming due to the measurements needed and, in the case of mixers and band pass filters, multiple frequencies that need to be entered to characterize the DUT. Device Expert allows you to quickly set up common measurements for these DUTs and save the DUT configuration.

In this topic:

- Features
- Select Device page help
- Select DUT Ports page help
- Define DUT Frequencies page help
- Define DUT Power page help
- Select and Apply Measurements page help
- Edit Channel Parameters and Properties dialog help
- External Source Error Messages

### Features

- Intuitive user interface for the setup of measurements common for Amplifiers, Mixers, and Band Pass Filter devices.
- Apply selected measurements to VNA.
- Append DE Wizard configured measurements to existing VNA setup, without disturbing any of your already configured window arrangements.
- Save and recall DUT configurations and mixer frequency ranges.

### Accessing DMX Settings

**Using Hardkey/SoftTab/Softkey**

1. Press **Setup** > **Main** > **Device Expert**....

**Using a mouse**

1. Click **Instrument**.
2. Select **Setup**.
3. Select the **Device Expert**....
Choose from the following selections, the wizard will automatically advance.

**Amplifier** – Set up a selection of common measurements for 2-port amplifier devices with gain.

**Converter** – Set up a selection of common measurements for frequency translating devices such as mixers, including up- or down-converters.

**Band Pass Filter** – Set up a selection of common measurements for a 2-port band pass filter device. This will setup measurements in 3 channels, corresponding to the upper and lower rejection band and the pass band, to optimize measurement speed.

**Save...** – Saves a DUT configuration.

**Recall...** - Recalls a DUT configuration. Clicking on the down arrow in the **Recall...** button accesses the following:

- **Recall auto-saved configuration** - Recalls the previous session whether it was manually saved or not.
- **Delete auto-saved configuration** - Deleted the auto-saved configuration.
Select DUT Ports page help

Amplifier

Converter
Band Pass Filter
Using the pull-down selection menus on the side of the DUT graphic the ports that the DUT will be connected to on the VNA can be selected. If the VNA supports IMD style measurements a check box will be visible to enable or disable the setup of IMD measurements by the Wizard. If IMD measurements are enabled the port selection for the input port will be disabled and automatically set to port 1 to take advantage of the internal combiner and second source on the VNA, if available.

For mixer/converter devices the LO port source can be selected from the pull-down menu of available sources. If external sources are configured those will also be present in the list as possible external sources. If the mixer/converter device has an embedded LO or is a 2-stage converter device those DUT properties can be selected on this page.

**Save...** – Saves a DUT configuration.

**Recall...** - Recalls a DUT configuration. Clicking on the down arrow in the **Recall...** button accesses the following:

- **Recall auto-saved configuration** - Recalls the previous session whether it was manually saved or not.
- **Delete auto-saved configuration** - Deleted the auto-saved configuration.
Define DUT Frequencies page help

The following images show the define DUT frequency pages available in the wizard. The start/stop or center/span frequencies for the measurements can be specified. In addition, the number of points for the measurement can be set on this page.

Amplifier

Converter
Band Pass Filter
For the converter device, two additional options are available to save and load defined mixer setups. **Save Mixer**… and **Load Mixer**… are different from the Wizard **Save**…/**Recall**… function as the mixer file only contains information regarding the mixer frequencies and high-/low-side selection without the additional information of the measurement configuration and VNA setup.

- **Save Mixer**… saves the mixer definition file including frequencies, the high-/low-side setting, and fixed/swept input settings.

- **Load Mixer**… loads the mixer frequency range definition file.

For BPF DUTs, there are three sets of frequency range inputs that you can use to specify the frequency ranges of the lower and upper rejection band as well as the pass band. Also available is a **Band Name** field that allows you to name the frequency range for easier recognition when the channels are set up on the VNA using the Wizard.

**Save**… – Saves a DUT configuration.

**Recall**… - Recalls a DUT configuration. Clicking on the down arrow in the **Recall**… button accesses the following:

- **Recall auto-saved configuration** - Recalls the previous session whether it was manually saved or not.
Delete auto-saved configuration - Deleted the auto-saved configuration.

**Define DUT Power page help**  
The DUT power stimulus conditions can specified in the pages shown below.

**Amplifier**

![Amplifier diagram]

**Converter**
Band Pass Filter
**Amplifier Gain** and **Conversion Gain** – Provide an estimate of the gain of the device. This is used by the Wizard to compute optimal input power levels to the DUT to prevent compression of the VNA receivers.

**Linear Input Power** – Nominal input power level to achieve linear DUT operation.

**Max Input Power** – The maximum input power level to apply to the DUT. For measurements requiring the power to be swept, for example gain compression measurement class, this sets the maximum power level supplied to the DUT during power sweeps.

**Caution!** Setting this to a high value on devices with gain could result in output powers that could damage the VNS receivers.

**LO1 Power** and **LO2 Power** – The power level to use for LO1 and, if applicable, LO2 for converter devices.

**Nominal Insertion Loss** – An estimate of the nominal insertion loss in the pass band region of a band pass filter. This value is used by the Wizard to estimate an optimal IFBW to achieve the target trace noise (described below).
Rejection Target – An estimate of the rejection level desired in the rejection regions of the band pass filter DUT.

Trace Noise Target – A value of the target trace noise, in dB(rms), the Wizard should attempt to optimize the VNA to achieve for pass band measurements on a band pass filter DUT.

Save... – Saves a DUT configuration.

Recall... - Recalls a DUT configuration. Clicking on the down arrow in the Recall... button accesses the following:

Recall auto-saved configuration - Recalls the previous session whether it was manually saved or not.

Delete auto-saved configuration - Deleted the auto-saved configuration.

Select and Apply Measurements page help

Measurements and parameters to be created using the wizard can be enabled or disabled in pages shown below.

Amplifier
Converter
Band Pass Filter
The check box next to each item can be used to enable or disable that item’s creation by the wizard.

Clicking the pencil icon next to each measurement will open the Edit Channel Parameters and Properties dialog where additional measurement parameters can be added and the settings for that measurement can be manually adjusted, if desired.

The parameters that will be measured for each of the measurement types can be displayed by clicking on the Parameters expander. Individual parameters can be enabled or disabled from this page.

Clicking the Finish button will apply the selected measurements and parameters to the VNA. Once completed the wizard will automatically exit.

**Delete All Currently Existing Measurements** – Enabling this option will delete all measurements that are currently configured on the VNA when the Finish button is clicked in the wizard.

**Calibrate All Measurements**… - Enabling this option will launch the VNA’s Calibrate All Channels wizard after the Device Expert wizard completes the setup of the selected measurements.

**Sweep after Applying Setup** - Enabling this option (default) applies the setup to the VNA and
performs a single sweep afterwards. Disabling this option applies the setup to the VNA but does not perform a single sweep afterwards.

**Save...** – Saves a DUT configuration.

**Recall...** - Recalls a DUT configuration. Clicking on the down arrow in the **Recall...** button accesses the following:

- **Recall auto-saved configuration** - Recalls the previous session whether it was manually saved or not.
- **Delete auto-saved configuration** - Deleted the auto-saved configuration.

---

**Edit Channel Parameters and Properties**

An example of the edit channel parameters and properties dialog is shown in the image below.

---

**Add** – Add a new parameter to the measurement.
**Change Format** – Change format of the selected parameter in the list of parameters

**Remove** – Delete parameter from the measurement

**Ok** – Save and apply changes to the measurement

**Cancel** – Abandons changes made in this dialog to the measurement.

---

**External Source Error Messages**

If an external source has been set up as an external device but is turned off, then Device Expert may cause errors to appear in the PNA firmware status bar when it loads. Device Expert attempts to activate these external sources. These errors can be safely ignored:

- **VI_ERROR_CONN_LOST**: The IO connection for the session has been lost
- **[SCPI: 1073]**: Unexpected error
Undo/Redo Settings

If you make an incorrect setting, you can quickly recover by selecting Undo. If you then incorrectly Undo a setting, you can Redo the undone setting.

- Undo and Redo applies ONLY to selected settings.
- The Undo stack remembers 16 levels of Undo-able settings.

How to Undo or Redo a setting

Tips:

- Click or touch the Undo and Redo Icons:

  ![Undo Redo Icons]

- With a mouse, right-click on the Softkeys or on the Entry toolbar.
- With a keyboard:
  - Undo....Ctrl+Z
  - Redo....Ctrl+Y

Using **Hardkey/SoftTab/Softkey**

1. Press **Undo > Main**.
2. Click **Undo** or **Redo**.

SCPI and COM programming and Undo/Redo:

- There are NO Programming commands to invoke Undo/Redo
- Programing commands are NOT Undo-able.
- The Undo stack is cleared when programming commands are sent to the VNA.

Using **a mouse**

1. Click **Undo** and **Redo** Icons on Active Entry or Softkey Toolbar.

Return To Task
To return to the previous task, press **Undo > Main > Return To Task**.

**Clear Undo History**

To clear the Undo stack, press **Undo > Main > Clear Undo History**.

**Undo and Security**

- Undo/Redo is disabled with **High** and **Extra** security levels. [Learn more.](#)
- State files that are saved for Undo/Redo purposes (for example: Preset) are deleted when any of the following occur:
  - The Security level is changed
  - The Network Analyzer App is started or closed.

**Selected Undo-able settings**

You can Undo or Redo the following **settings**:

**Note:** There are several settings that are NOT Undo-able. Because of this, when you attempt to Undo a long sequence of operations, it is unlikely that the original state can be recreated exactly.

- **Preset**
- **File Recall**
- **Frequency Settings**
  - For Standard Class, Gain Compression, and NF: Start, Stop, Center, Span, CW
  - For SMC, VMC: Mixer Setup dialog Apply
  - For Swept IMD: Freq softkeys, setup dialog Apply
  - IM Spectrum: Freq softkeys, setup dialog Apply
  - Noise Figure: Freq softkeys, setup dialog Apply
  - GCX, IMDX, NFX: setup dialog Apply
- **Turn off Marker** and **Marker All OFF**
- **Number of Points**
- **Power Level** - most applications and S-parameters
• Add or Change Measurement Class
• Turn OFF Channel
• Close Window
• New Channel, new Window, and new Trace.
• Delete Trace
• Window Tile
• Change Layout (1x, 2x, 3x, 4x)
• Move Trace, Drag Trace
• Zoom XY, Zoom Out Full
• Autoscale All, Autoscale
• Scale, Reference Level, Reference Position
• Scale Coupling dialog
• Electrical Delay
• Phase Offset
• Measurement Setups dialog
• Format
• Sweep Type
• Data->Memory
• Single Marker Searches (Max, Min, Target, Peak…)
• Multi-marker Searches (Bandwidth, Power Saturation, Normal Operating Pt)
• Change a Marker’s stimulus value: softkeys, dialog or drag
• Change cell in Segment Table
• Mechanical Settings dialog
Optimize Measurements

A measurement has many interdependent settings. You can modify the settings to achieve the goals of your measurement application: faster throughput or greater measurement accuracy.

**Increase Measurement Throughput**

- Achieve Fastest Sweep
- Switch Between Multiple Measurements
- Data Transfer Speed
- Using Macros

**Improve Measurement Accuracy**

- Increase Dynamic Range
- Reduce Noise
  - Averaging
  - IFBW
  - Smoothing
- Group Delay (Separate Topic)
- Improve Phase Measurement Accuracy
- Improve Accuracy for Electrically Long Devices
- Improve Reflection Accuracy on Low-Loss 2-Port Devices
- Increase Measurement Stability
- Decrease Receiver Crosstalk
- Reduce the Effects of Test Accessories

**Caution:** Avoid expensive repairs to your analyzer. Read Electrostatic Discharge Protection.
Dynamic Range

Dynamic range is the difference between the analyzer receiver's maximum input power and the minimum measurable power (noise floor). For a measurement to be valid, input signals must be within these boundaries.

Increasing dynamic range is important if you need to measure very large variations in signal amplitude, such as filter bandpass and rejection. The dynamic range is shown below for an example measurement.

To help reduce measurement uncertainty, the analyzer dynamic range should be greater than the response that the DUT exhibits. For example, measurement accuracy is increased when the DUT response is at least 10 dB above the noise floor. The following methods can help you increase the dynamic range.

- Increase the Device Input Power
- Reduce the Receiver Noise Floor
- Improving Dynamic Range using the Front-Panel Jumpers

Other topics about Optimizing Measurements

Increase Device Input Power

Increase the DUT input power so that the analyzer can more accurately detect and measure the DUT output power. However, use caution - too much power can damage the analyzer receiver or cause compression distortion.

**Caution! Receiver input damage level: +15 dBm.**

See how to increase input power to the device

**Tip:** You can further increase dynamic range by using an external booster amplifier to increase the input
power to the DUT. See High Power Amplifier Measurements.

Reduce the Receiver Noise Floor

You can use the following techniques to lower the noise floor and increase the analyzer's dynamic range.

- Reduce crosstalk between the VNA receivers when measuring signals close to the noise floor. See Receiver Crosstalk.)

- Use **Sweep Averaging** - learn more about Sweep Average

- Reduce the **IF Bandwidth** - learn more about IF Bandwidth.

- In **Segment sweep** mode each segment can have its own IF bandwidth. For example, when measuring a filter:
  
  - In the passband, the IF bandwidth can be set wider for a fast sweep rate, as long as high-level trace noise is kept sufficiently small.

  - In the reject band, where noise floor contributes significantly to measurement error, the IF bandwidth can be set low enough to achieve the desired reduction in average noise level.

Improving Dynamic Range using the Front-Panel Jumpers

**Direct Access**

The simplest method to improve dynamic range is to remove a RCVR 'n' IN front-panel jumper and route the DUT output directly into that VNA receiver. This bypasses the directional coupler and limits the ability to provide Full Error Correction because the signal can not be applied in the reverse direction.

Refer to the VNA specifications to learn the dynamic range that is available with direct receiver access.

**Reverse the directional coupler**

Another method to improve dynamic range is to reverse the signal path in the test-port coupler and bypass the loss typically associated with the coupled arm.

As shown in the following graphic, the signal is applied to Port 2. The signal bypasses the coupled arm via the jumper cable connected to the Coupler Thru (or Coupler In) and the Receiver B In (or B In) ports.

When making measurements in reverse direction, the system dynamic range is degraded by 15 dB.

**Note:** Your analyzer's block diagram may contain different components than shown below. To see the block diagram for your VNA model, see the bottom of the specs/data sheet.
See Also

Front-panel Jumpers (image)

Using the Front Panel Jumpers

Specifications
Number of Points

A data point is a sample of data representing a measurement at a single stimulus value. You can specify the number of data points that the analyzer measures across a sweep. (A "sweep" is a series of consecutive data point measurements, taken over a sequence of stimulus values.)

The analyzer sweep time changes proportionally with the number of points. However, the overall measurement cycle time does not. See Technical Specifications for more information on how the number of points, and other settings, affect the sweep time.

How to change the number of data points

Select a number or click Custom to invoke a dialog box

Using Hardkey/SoftTab/Softkey

1. Press Sweep > Main > Number of Points.

Number of Points dialog box help

Specifies the number of data points that the analyzer gathers during a measurement sweep. You can specify any number from 1 to 100,003. The default value is 201.

Note: Some measurement classes (such as GCA and Swept IMD) may have different maximum points limitations.

Two data points are required for Time Domain.

Tips:

- To achieve the greatest trace resolution, use the maximum number of data points.
- For faster throughput use the smallest number of data points that will give you acceptable resolution.
- To find an optimized number of points, look for a value where there is not a significant difference in the measurement when you increase the number of points.
- To ensure an accurate measurement calibration, perform the calibration with the same number of points that will be used for the measurement.

The number of points is the number of data items collected in one sweep. It can be set for each...
channel independently.

- To obtain a higher trace resolution against the stimulus value, choose a larger value for number of points.
- To obtain higher throughput, keep the number of points to a smaller value within an allowable trace resolution.
- To obtain higher measurement accuracy after calibration, perform calibration using the same number of points as in actual measurements.
Phase Measurement Accuracy

You can increase the accuracy of phase measurements by using the following features:

- **Electrical Delay**
- **Phase Offset**
- **Spacing Between Frequency Points (Aliasing)**

**See Also**

- Port Extensions
- Comparing the Delay Functions
- Phase Control
- Phase Coherent Measurements

**Learn more about Phase measurements**

**Electrical Delay**

Electrical delay is a mathematical function that simulates a variable length of lossless transmission line.

Use the electrical delay feature to compensate for the linear phase shift through a device. This feature allows you to look at only the deviation from linear phase of the device.

You can set the electrical delay independently for each measurement trace.

**How to set Electrical Delay**

**Using** Hardkey/SoftTab/Softkey

1. Press Scale > Electrical Delay.
### Electrical Delay dialog box help

**Electrical Delay** Specifies the value of delay added or removed, in Time or Distance. This compensates for the linear phase shift through a device. You can set the electrical delay independently for each measurement trace.

**Velocity Factor** Specifies the velocity factor that applies to the medium of the device that was inserted after the measurement calibration. The value for a polyethylene dielectric cable is 0.66 and 0.7 for PTFE dielectric. 1.0 corresponds to the speed of light in a vacuum.

Velocity factor can also be set from the Port Extensions dialog and Time Domain Distance Marker Settings.

**Softkey Display** Allows you to enter delay in either Time or Distance using the softkeys and Active Entry toolbar.

**Delay Distance** Changes the value when the Delay Time or Delay Distance values are changed.

**Distance Units** Select from Meters, Inches, or Feet. The step size will not change automatically when this value is changed.

**Media**

- **Coax** Select if the added length is coax. Also specify the velocity factor of the coax.

- **Waveguide** Select if the added length is waveguide. Also specify the low frequency cutoff of the waveguide.

- **Cutoff Freq** Low frequency cutoff of the waveguide.

Learn about Electrical Delay (scroll up)

### Phase Offset

Phase offset mathematically adjusts the phase measurement by a specified amount, up to 360°. Use this feature in the following ways:

- **Improve the display of a phase measurement.** This is similar to the way you would change the reference level in an amplitude measurement. Change the phase response to center or align the response on the screen.

- **Emulate a projected phase shift in your measurement.** For example, if you know that you need to add a cable and that the length of that cable will add a certain phase shift to your measurement, you can use phase offset to add that amount and simulate the complete device measurement.
How to set Phase Offset

Using Hardkey/SoftTab/Softkey

1. Press **Scale > Constants > Phase Offset.**

---

**Phase Offset** dialog box help

**Phase Offset** Type a value or use the up and down arrows to select any value up to 360 degrees.

Learn about **Phase Offset** (scroll up)

---

**Spacing Between Frequency Points (Aliasing)**

The analyzer samples data at discrete frequency points, then connects the points, creating a trace on the screen.

If the phase shift through a device is >180° between adjacent frequency points, the display can look like the phase slope is reversed. This is because the data is undersampled and aliasing is occurring.

If you are measuring group delay and the slope of the phase is reversed, then the group delay will change sign. For example, the following graphic shows a measurement of a SAW bandpass filter.

- The left measurement has 51 points and indicates the group delay is negative, which is a physical impossibility. That is, the response is below 0 seconds reference line.

- The right measurement shows an increase to 201 points which indicates the group delay is positive. That is, the response is above the 0 seconds reference line.
**Tip:** To check if aliasing might be occurring in a measurement, either increase the number of points or reduce the frequency span.
Electrically-Long Device Measurements

A signal coming out of a device under test may not be exactly the same frequency as the signal going in to a device at a given instant in time. This can sometimes lead to inaccurate measurement results. You can choose between two techniques to eliminate this situation and increase measurement accuracy.

- Why Device Delay May Create Inaccurate Results
- Solutions to Increase Measurement Accuracy
  - Slow the Sweep Speed
  - Add Electrical Length to the R Channel

Other topics about Optimizing Measurements

Why Device Delay May Create Inaccurate Results

The following graphic shows an example of this situation:

- In the network analyzer, the source and receiver are phase locked together and sweep simultaneously through a span of frequencies.
- The signal flow through the Device Under Test (DUT) is shown as different colors for different frequencies.
- You can see as a stimulus frequency travels through the DUT, the analyzer tunes to a new frequency just before the signal arrives at the receiver. This causes inaccurate measurement results.

If the analyzer is measuring a long cable, the signal frequency at the end of the cable will lag behind the network analyzer source frequency. If the frequency shift is appreciable compared to the network analyzer's IF detection bandwidth (typically a few kHz), then the measured result will be in error by the rolloff of the IF filter.

Note: There is no fixed electrical length of a device where this becomes an issue. This is because there
are many variables that lead to measurement speed. When high measurement accuracy is critical, lower the sweep speed until measurement results no longer change.

### Solutions to Increase Measurement Accuracy

Choose from the following methods to compensate for the time delay of an electrically long device.

#### Slow the Sweep Speed

The following methods will slow the sweep speed.

- Increase the Sweep Time
- Increase the Number of Points
- Use Stepped Sweep
- Set Dwell Time

#### Add Electrical Length to the R Channel

**Note**: This method applies to VNA models with front panel loops.

Instead of slowing the sweep, you can compensate for the electrical length of a cable or fixture.

a. Remove the R-channel jumper on the front panel of the analyzer.

b. Replace the jumper with a cable of about the same length as the device under test.

1. Add the cable on the R1 channel for $S_{11}$ and $S_{21}$ measurements.
2. Add the cable on the R2 channels for $S_{22}$ and $S_{12}$ measurements.

c. Set the analyzer for a fast sweep.

#### Configuration for $S_{22}$ and $S_{12}$ Measurements

This method balances the delays in the reference and test paths, so that the network analyzer's ratioed transmission measurement does not have a frequency-shift error.
Note: This method works well if the delay is in a cable or fixture. For devices with long delays, this method is only suitable for uncalibrated measurements.
Reflection Accuracy on Low-Loss 2-Port Devices

To make accurate reflection measurements that have a 1-port calibration, you should terminate the unmeasured port.

- Why Terminate the Unmeasured Port
- How to Terminate the Unmeasured Port
- Resulting Measurement Uncertainty

Other topics about Optimizing Measurements

Why Terminate the Unmeasured Port

A 2-port calibration corrects for all 12 twelve error terms. A 1-port calibration corrects for directivity, source match and frequency response, but not load match. Therefore, for highest accuracy, you must make the load match error as small as possible. This especially applies for low-loss, bi-directional devices such as filter passbands and cables. You do not need to be concerned with load match when you are measuring a device with high reverse isolation, such as an amplifier.

How to Terminate the Unmeasured Port

Use one of the following methods:

- Connect a high-quality termination load (from a calibration kit, for example) to the unmeasured port of your device. This technique yields measurement accuracy close to that of a Full SOLT 2-port calibration.

- Connect the unmeasured port of your device directly to the analyzer, inserting a 10 dB precision attenuator between the device output and the analyzer. This improves the effective load match of the analyzer by approximately twice the value of the attenuator, or 20 dB.
Resulting Measurement Uncertainty

The following graph illustrates the measurement uncertainty that results from terminating with and without a precision 10 dB attenuator on the output of the test device.

Legend

- Filter Reflection
- Uncertainty with attenuator
- Uncertainty without attenuator

The calculations below show how adding a high-quality 10 dB attenuator improves the load match of the analyzer.

Note: The corresponding linear value is shown in parentheses.
**Network Analyzer:**

| Load match (NALM) | 18 dB (.126) |
| Directivity (NAD) | 40 db (.010) |

**Filter:**

| Insertion loss (FIL) | 1dB (.891) |
| Return loss (FRL) | 16 dB (.158) |

**Attenuator:**

| Insertion loss (AIL) | 10 dB (.316) |
| SWR (ASWR) | 1.05 (.024) |

32.26 dB Return Loss

**Calculations:**

<table>
<thead>
<tr>
<th>Without Attenuator</th>
<th>With Attenuator</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho_{NA}$</td>
<td></td>
</tr>
<tr>
<td>$= (FIL)<em>(NALM)</em>(FIL)$</td>
<td>$= (FIL)<em>(AIL)</em>(NALM)<em>(AIL)</em>(FIL)$</td>
</tr>
<tr>
<td>$= (.891)<em>(.126)</em>(.891)$</td>
<td>$= (.891)<em>(.316)</em>(.126)<em>(.316)</em>(.891)$</td>
</tr>
<tr>
<td>$.100$</td>
<td>$.010$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\rho_{Attenuator}$</th>
<th>NA</th>
<th>NA</th>
</tr>
</thead>
<tbody>
<tr>
<td>$= (FIL)<em>(ASWR)</em>(FIL)$</td>
<td>$= (FIL)<em>(ASWR)</em>(FIL)$</td>
<td></td>
</tr>
<tr>
<td>$= (.891)<em>(.024)</em>(.891)$</td>
<td>$= (.891)<em>(.024)</em>(.891)$</td>
<td></td>
</tr>
<tr>
<td>$.019$</td>
<td>$.019$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Worst Case Error</th>
<th>$\rho_{NA}$</th>
<th>$\rho_{NA} + \rho_{Attn.}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$= .1$</td>
<td>$=.100$</td>
<td>$.01+.019$</td>
</tr>
<tr>
<td>$(EWC)$</td>
<td>$(EWC)$</td>
<td>$(EWC)$</td>
</tr>
<tr>
<td>$=.029$</td>
<td>$=.029$</td>
<td>$=.029$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Uncertainty Adds</th>
<th>$-20\log(FRL)+(EWC)+(NAD)$</th>
<th>$-20\log(FRL)+(EWC)+(NAD)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$= 11.4$ dB</td>
<td>$= 11.4$ dB</td>
<td>$= 14.1$ dB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Uncertainty Subtracts</th>
<th>$-20\log(FRL)-(EWC)-(NAD)$</th>
<th>$-20\log(FRL)-(EWC)-(NAD)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$= 26.4$ dB</td>
<td>$= 26.4$ dB</td>
<td>$= 18.5$ dB</td>
</tr>
</tbody>
</table>
Measurement Stability

There are several situations that can cause unstable measurements. To ensure that you are making repeatable measurements, you can use various methods to create a stable measurement environment.

- Frequency Drift
- Temperature Drift
- Inaccurate Measurement Calibrations
- Device Connections

Other topics about Optimizing Measurements

Frequency Drift

The analyzer frequency accuracy is based on an internal 10 MHz frequency oscillator. See Technical Specifications for stability and aging specifications.

If your measurement application requires better frequency accuracy and stability, you can override the internal frequency standard and provide your own high-stability external frequency source through the 10 MHz Reference Input connector on the rear panel.

Temperature Drift

Thermal expansion and contraction changes the electrical characteristics of the following components:

- Devices within the analyzer
- Calibration kit standards
- Test devices
- Cables
- Adapters

To reduce the effects of temperature drift on your measurements, do the following.

- Switch on the analyzer 1/2 hour before performing a measurement calibration or making a device measurement.
• One hour before you perform a measurement calibration, open the case of the calibration kit and take the standards out of the protective foam.

• Use a temperature-controlled environment. All specifications and characteristics apply over a 25 °C ±5 °C range (unless otherwise stated).

• Ensure the temperature stability of the calibration kit devices.

• Avoid handling the calibration kit devices unnecessarily during the calibration procedure.

• Ensure the ambient temperature is ±1°C of the measurement calibration temperature.

**Inaccurate Measurement Calibrations**

If a measurement calibration is inaccurate, you will not measure the true response of a device under test. To ensure that your calibration is accurate, you should consider the following practices:

• Perform a measurement calibration at the points where you connect the device under test, that is, the reference plane.

• If you insert any additional accessory (cable, adapter, attenuator) to the test setup after you have performed a measurement calibration, use the port extensions function to compensate for the added electrical length and delay.

• Use calibration standards that match the definitions used in the calibration process.

• Inspect, clean, and gage connectors. See Connector Care.

See Accurate Measurement Calibrations for more detailed information.

**Device Connections**

Good connections are necessary for repeatable measurements. To help make good connections, do the following:

• Inspect and clean the connectors for all of the components in the measurement setup.

• Use proper connection techniques.

• Avoid moving the cables during a measurement.
Noise Reduction Techniques

Random electrical noise which shows up in the analyzer receiver chain can reduce measurement accuracy. The following features help reduce trace noise and the noise floor which can lead to better dynamic range and more accurate measurements.

**Note:** The trace noise in microwave VNAs becomes worse below 748 MHz and is especially obvious between 10 MHz and 45 MHz. See [Reduce IFBW](#).

- Averaging
- IF Bandwidth
- Modulation Distortion Bandwidth Settings *(Option S93070xB,A9x070A/B Modulation Distortion only)*
- LF Auto BW
- Trace Smoothing

**See Also**

- Group Delay
- Increase Dynamic Range
- VNA data processing map.

### Other topics about Optimizing Measurements

**Averaging**

Averaging is a feature that reduces the effects of random noise on a measurement. There are two types of averaging: Point or Sweep.

The Point averaging type computes averaging on each data point before stepping to the next data point. You determine the number of measurements by setting the averaging factor (enabled by clicking the Averaging button). The higher the averaging factor, the greater the amount of noise reduction.

The Sweep averaging type computes averaging on subsequent sweeps until the required number of averaging sweeps are performed.

**Effects of Sweep Average**
Both Averaging and IF Bandwidth can be used for the same benefit of general noise reduction. For minimizing very low noise, Averaging is more effective than reducing IF bandwidth. Generally, Averaging takes slightly longer than IF bandwidth reduction to lower noise, especially if many averages are required. Also, changing the IF bandwidth after calibration results in uncertain accuracy.

**How to Set Averaging**

**Using Hardkey/SoftTab/Softkey**

1. Press **Avg BW > Main > Averaging**.
2. Enter the Averaging number.

**Using a mouse**

1. Click **Response**.
2. Select **Avg BW**.
3. Select **Averaging...**

**Average dialog box help**

**Average ON** Check to enable Averaging.

**Average Factor** Specifies the number of measurements that are averaged. Range of 1 to 65536 ($2^{16}$).

**Average Type**

- **Sweep** Each data point is based on the average of the same data point measured over consecutive sweeps. When the number of sweeps = Average Factor, the averaging continues following the Sweep Averaging formula.

- **(Sweep) Restart** Begins a new set of measurements that are used for the average. Applies only to Sweep averaging - NOT Point.
**Point** Each data point is measured the number of times specified by the Average Factor, and then averaged, before going to the next data point.

- On subsequent sweeps, averaging is automatically restarted by measuring each data point again the number of times specified by the Average Factor.
- Because measurements occur quickly in the background, the Average Counter is NOT updated.
- Point averaging is NOT available in **Gain Compression**, or **Noise Figure** Apps.

**Notes**

- An **Average Counter** appears on the screen when Sweep averaging is selected, displaying the number of sweeps that has been averaged. The effect on the signal trace can be viewed as the Average Factor increases. This can assist in the selection of the optimum number of sweep averages. The Average Counter is NOT updated for **Point** averaging.

- **Channel-wide scope** - Averaging is enabled and the factor is set for all measurements in a channel. The Average counter is displayed for each channel.

- **Calibration** - Because averaging is a mathematical process that occurs after the raw measurement is made, averaging can be turned ON before or after calibration without invalidating the error correction terms. If averaging is ON before calibration, the measurement of calibration standards are averaged measurements. More time is needed to perform the calibration, but there will be less noise in the resulting error correction terms. Subsequent corrected measurements will also have less noise error. In addition, noise is further reduced by turning Averaging ON after calibration.

- **Triggering** is implemented separately from Averaging. For example, setting averaging factor to 4 has NO effect on the number of triggers that are required to achieve 4 sweeps or 4 data points.

- **Unratioed** measurements - Although averaging unratioed (single receiver) measurements is allowed, you may see unexpected results.

  - The noise floor does not drop when averaging unratioed measurements as on ratioed measurements.
  - Phase results may tend toward 0. This is because phase measurements are relative by nature. Measuring absolute phase with a single receiver appears random. Averaging random positive and negative numbers will tend toward 0.

**Sweep Averaging Formula**

\[
\text{NewAvg} = \left( \frac{\text{NewData}}{n} \right) + \left( \frac{\text{OldAvg} \times (n-1/n)}{n} \right) \quad \text{where } n = \text{average factor}
\]

From the formula, you can see that data from the first \(n\) sweeps continues to be included in the results of subsequent sweeps. Its effect is increasingly smaller but never diminishes to zero. For example, with \(n = 5\), the average of the 5 sweeps is displayed. On the 6th sweep, you see 4/5 the
average of the first 5 sweeps plus 1/5 the new sweep.

The effects of older data can be eliminated by clicking **Restart**.

Learn more about Averaging (scroll up)

**IF Bandwidth**

The received signal is converted from its source frequency to a lower intermediate frequency (IF). The bandwidth of the IF bandpass filter is adjustable down to a minimum of 1 Hz. The maximum IF varies depending on the VNA model.

Reducing the IF receiver bandwidth reduces the effect of random noise on a measurement. Each tenfold reduction in IF bandwidth lowers the noise floor by 10 dB. However, narrower IF bandwidths cause longer sweep times.

- **Channel** - IF bandwidth can be set independently for each channel
- **Segment sweep** - IF bandwidth can be set independently for each segment of segment sweep.
- **Calibration** - Changing the IF bandwidth after calibration will cause a 'C-delta' correction level, which means that calibration accuracy is uncertain.

**Effect of Reducing IF Bandwidth**

![IF Bandwidth comparison](image)

**How to set IF Bandwidth**

**Using Hardkey/SoftTab/Softkey**

1. Press **Avg BW > Main > IF Bandwidth**.
2. Enter the IF Bandwidth value.

**Using a mouse**

1. Right click on the BW icons on the status bar.
2. Select an **IF Bandwidth**...

**IF Bandwidth** dialog box help

Right click on the BW icons on the status bar then select **IF Bandwidth**... to display the IF Bandwidth dialog:
IF Bandwidth  Specifies the IF (receiver) bandwidth. The value of IF bandwidth is selected by scrolling through the values available in the IF bandwidth text box. The IF BW is set independently for each channel.

The following IFBW values are common to all models:

1 | 2 | 3 | 5 | 7 | 10 | 15 | 20 | 30 | 50 | 70 | 100 | 150 | 200 | 300 | 500 | 700 | 1k | 1.5k | 2k | 3k | 5k | 7k | 10k | 15k | 20k | 30k | 50k | 70k | 100k | 150k | 200k | 280k | 360k | 600k | 1M | 1.5M | 2M | 3M | 5M | 7M | 10M | 15M

For the highlighted IFBW settings (1 MHz and above):

- 7 MHz to 15 MHz settings are available ONLY with DSP version 5.0 and above.
- The primary use for IFBW > 600 kHz is for wideband pulsed and pulse profile measurements. They do NOT provide faster sweep speeds for non-pulsed measurements.
- A slight shift (1dB or more) in Log Mag traces may be seen when switching in and out of these bandwidths.
- Available in Step sweep mode only - NOT available in Analog sweep.

Note: The IFBW is limited to 600 kHz when performing Swept IMD measurements even if the Wide IF path is selected.

IFBW Shape  Selects the digital filter (window) to apply to the time domain IF signal. The filter effectively "shapes" the signal before application of the DFT to help avoid discontinuities which add unwanted frequency content to the spectrum. Each filter has its own advantages and disadvantages.

Gaussian filter shapes have much lower side lobe levels than the standard VNA filters, and so measurements with them will be less sensitive to closely spaced signals. Normally, in standard S-parameters this is not important and these filters show no benefit. But in cases such as mixer measurements, they may reduce spurious responses when spurious signals are close to the main
signal to be measured. They have already been implemented in the spectrum analyzer mode.

**Standard** - Legacy filter. This filter is the fastest, but has side lobes that rise to ~-32 dBC.

**Gaussian** - This filter takes longer to acquire a point but has no side lobes.

The following shows the Standard and Gaussian 100 kHz filters:

![Filter Comparison](image)

---

**Reduce IF BW at Low Frequencies**

On VNA models with a maximum frequency of 20 GHz and higher, the trace noise becomes worse below about 400 MHz. This is especially obvious between 10 MHz and 45 MHz and also when Time Domain is ON. [See VNA models / maximum frequencies.](#)

When this box is checked, the VNA uses a smaller IF Bandwidth than the selected value at frequencies indicated below.

This setting:

- can be made for each channel.
- is ON (checked) by default.
Use the following calculations to determine the actual IF Bandwidth that is used. If the result is NOT a selectable IF BW, the next higher selectable value is used.

In the following table and example, the next band starts at .01 Hz above the Stop Frequency. This is indicated with '+'.

<table>
<thead>
<tr>
<th>Stop Frequency</th>
<th>VNA Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>14 MHz</td>
<td>0.05</td>
</tr>
<tr>
<td>19 MHz</td>
<td>0.05</td>
</tr>
<tr>
<td>27 MHz</td>
<td>0.05</td>
</tr>
<tr>
<td>38 MHz</td>
<td>0.05</td>
</tr>
<tr>
<td>53 MHz</td>
<td>0.05</td>
</tr>
<tr>
<td>75 MHz</td>
<td>0.1</td>
</tr>
<tr>
<td>105 MHz</td>
<td>0.1</td>
</tr>
<tr>
<td>146 MHz</td>
<td>0.14</td>
</tr>
<tr>
<td>175 MHz</td>
<td>0.14</td>
</tr>
<tr>
<td>205 MHz</td>
<td>0.29</td>
</tr>
<tr>
<td>250 MHz</td>
<td>0.5</td>
</tr>
<tr>
<td>396 MHz</td>
<td>1</td>
</tr>
<tr>
<td>396+ MHz and above</td>
<td>1</td>
</tr>
</tbody>
</table>

Example:

On a N5224B, the selected IF BW is 30 KHz.

With Reduce IF BW at Low Frequencies checked, the actual IF Bandwidths used are:

- From 53+ MHz to 175 MHz: 30,000 Hz * .025 = 750 Hz (next higher selectable value: 1 kHz)
- From 175+ MHz to 250 MHz: 30,000 Hz * .15 = 4.5 kHz (next higher selectable value: 5 kHz)
- From 250+ MHz to 396 MHz: 30,000 Hz * .5 = 15 kHz
- From 396+ MHz to stop sweep = 30 kHz
**OK**  Selects the IF BW value shown in the text box.

Learn about IF Bandwidth (scroll up)

Modulation Distortion Bandwidth Settings (Option S9x070xB, S9x070A/B Modulation Distortion only)

**How to access Bandwidth settings**

**Using Hardkey/SoftTab/Softkey**

1. Press **Avg BW > Main**

**Using a mouse**

1. Click **Response**.
2. Select **Avg BW**.
3. Select **Bandwidth**.

**Bandwidth dialog box help**

**Noise BW** is equal to the Resolution BW divided by the Vector Average factor. You cannot directly set the Resolution BW or Vector Average in a Modulation Distortion channel. Resolution BW is always set to its maximum value for the given signal chosen. Vector Average is automatically set depending on the Noise BW setting.

**Auto** - Check to set the Noise BW to its maximum possible value. This results in the fastest possible sweep but increases the noise floor. If Auto is checked, then the Vector Average is set to 1, resulting in the widest Noise BW.

The Noise BW is displayed in the Softkey and in the middle of the display x-axis annotation.
**LF Auto BW**

When **LF Auto BW** is ON (default), the VNA uses a smaller IF Bandwidth than the selected value at low frequencies. This is the same as the **Reduce IF BW at Low Frequencies** selection in the IF Bandwidth dialog. Learn about IF Bandwidth (scroll up).

LFE (**Low Frequency Extension**) also uses a smaller IF Bandwidth than the selected value at low frequencies but does not use the same tables. Instead, LFE selects the IF Bandwidth based on the following:

1. User set IF Bandwidth.
2. Closest IF Bandwidth below Receiver Frequency / 4.

For example:

User set IF Bandwidth - 100 kHz

Receiver Frequency = 180 kHz

\[ \frac{180 \text{ kHz}}{4} = 45 \text{ kHz} \]

Next lower IF Bandwidth is 30 kHz

**How to enable/disable LF Auto BW**

**Using Hardkey/SoftTab/Softkey**

1. Press **Avg BW > Main > LF Auto BW**.
2. ON enables and OFF disables **LF Auto BW**.

**Trace Smoothing**

Trace smoothing averages a number of adjacent data points to smooth the displayed trace. The number of adjacent data points that get averaged together is also known as the smoothing aperture. You can specify aperture as either the number of data points or the percentage of the x-axis span.

Trace Smoothing reduces the peak-to-peak noise values on broadband measured data. It smooths trace noise and does not increase measurement time significantly.

Because Trace Smoothing follows Format in the data processing map, the formatted data is smoothed. Smoothing is automatically turned off if the format is Polar or Smith Chart.

Learn more about Data Format Types.
See the data processing map.

Tips:

- Start with a high number of display points and reduce until you are confident that the trace is not giving misleading results.

- Do not use smoothing for high-resonance devices, or devices with wide trace variations. It may introduce misleading information.

- Smoothing is set independently for each trace.

Effects of Smoothing on a Trace

<table>
<thead>
<tr>
<th>Without Smoothing</th>
<th>With Smoothing</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Graph]</td>
<td>[Graph]</td>
</tr>
</tbody>
</table>

How to set Trace Smoothing

Using Hardkey/SoftTab/Softkey

1. Press Avg BW > **Smoothing** > Smoothing ON|OFF.

Smoothing ON  When checked, applies smoothing to the displayed trace.

Percent of Span  Specify percent of the swept stimulus span to smooth. For example, for a trace that contains 100 data points, and specify a percent of span = 11%, then the number of data points that are averaged is 11.

Points  Specify the number of adjacent data points to average.

Learn about Trace Smoothing (scroll up)
Crosstalk

Crosstalk is energy leakage between analyzer signal paths. This can be a problem with high-loss transmission measurements. Although the crosstalk specification of the analyzer is exceptional, you can reduce the effects of crosstalk by doing the following:

- Set the Sweep to Alternate
- Perform an Isolation Calibration

Other topics about Optimizing Measurements

Set the Sweep to Alternate

This selection is no longer available from the user interface. Learn more.

Perform an Isolation Calibration

For transmission measurements, a response and isolation measurement calibration helps reduce crosstalk because the analyzer measures and then subtracts the leakage signal during the measurement calibration. The calibration improves isolation so that it is limited only by the noise floor.

Note: Isolation is never performed on a Smart (Guided) Calibration. Learn more.

Generally, the isolation error falls below the noise floor. So when you are performing an isolation calibration you should use a noise reduction technique such as sweep averages or reducing the IF bandwidth.
Effects of Accessories

Accessories in a configuration may affect the results of a device measurement. You can choose between these analyzer features that reduce or remove the effects of accessories:

- **Power Slope to Compensate for Cable Loss**
- **Gating to Selectively Remove Responses**
- **De-embedding a 2-port device** (separate topic)

**Power Slope to Compensate for Cable Loss**

If you have a long cable or other accessory in a measurement configuration where a power loss occurs over frequency, apply the power slope function. This function increases the analyzer source power by a rate that you define (dB/GHz).

1. Press **Power > Leveling & Offsets**.
2. If the slope function is not already switched on, click the button beside **Slope**.
3. In the **Slope** box, enter the rate that you want the source power to increase over the frequency sweep.

**Gating to Selectively Remove Responses**

Gating is a feature in the time domain (option S93010A/B) that allows the analyzer to mathematically remove responses. You can set the gate for either a reflection or transmission response, but you will see different results.

- **Gating a reflection response** isolates a desired response (such as a filter’s return loss), from unwanted responses (such as adapter reflections or connector mismatches).
- **Gating a transmission response** isolates a specific path in a multipath device that has long electrical lengths.

See **Time Domain Gating** for more information.
Achieve Fastest Sweep

You can achieve the fastest measurement sweep by adjusting the following:

- **Sweep Settings**
- **Noise Reduction Settings**
- **Measurement Calibration Choice**
- **Unnecessary Functions**

### Other topics about Optimizing Measurements

### Sweep Settings

Consider changing each of the following settings as suggested.

- **Frequency Span** - Measure only the frequencies that are necessary for your device.
- **Segment Sweep** - Use segments to focus test data only where you need it.
- **Switch Off Stepped Sweep** - Use linear swept mode to minimize sweep time when possible.
- **Auto Sweep Time** - Use this default to sweep as quickly as possible for the current settings.
- **Number of Points** - Use the minimum number of points required for the measurement.

For more information on how number of points and other settings affect sweep cycle time, see Technical Specifications.

### Noise Reduction Settings

Using a combination of these settings, you can decrease the sweep time while still achieving an acceptable measurement.

- **IF Bandwidth.** Use the widest IF bandwidth that will produce acceptable trace noise and dynamic range.
- **Average.** Reduce the average factor, or switch Average off.
Measurement Calibration Choice

Choose the appropriate type of calibration for the required level of accuracy.

When full 2-port error correction is applied, the analyzer takes both forward and reverse sweeps to gather all 12 error correction terms. This occurs even with a single S11 measurement displayed. All displayed measurements are updated as the second sweep is performed. Both sweeps are performed using the specified sweep time.

When calibrating greater than 2 ports, the following formula is used to determine the number of sweeps required:

- \( N \times (N-1) \) where \( N \) = the number of ports.

When full 3-port calibration is applied, 6 sweeps are required; forward and reverse for each port pair. With full 4-port correction, 12 sweeps are required, and so forth.

To limit the measurement time, perform ONLY the level of calibration that your measurements require. For example, if making only an S11 measurement, perform a 1-port calibration on that port.

Sweep speed is about the same for uncorrected measurements and measurements done using a response calibration, or one-port calibration. For more information see Select a Calibration.

Unnecessary Functions

The analyzer must update information for all active functions. To achieve an additional increase in sweep speed, switch off all of the analyzer functions that are not necessary for your measurement application.

- **Delete Unwanted Traces**
- **Switch Off Unwanted Markers**
- **Switch Off Smoothing**
- **Switch Off Limit Testing**
- **Switch Off Math Functions**

Analyzer sweep speed is dependent on various measurement settings. Experiment with the settings to get the fastest sweep and the measurement results that you need.
If you need to make multiple measurements to characterize a device, you can use various methods to increase throughput. Experiment with these methods to find what is best for your measurement application needs.

- Set Up Measurements for Increased Throughput
  - Arrange Measurements in Sets
  - Use Segment Sweep
  - Trigger Measurements Selectively
- Automate Changes Between Measurements
- Recall Measurements Quickly

**Other topics about Optimizing Measurements**

### Set Up Measurements for Increased Throughput

To achieve optimum throughput of devices that require multiple measurements, it is helpful to know the operation of the analyzer. This knowledge allows you to set up the measurement scenarios that are best for your applications.

Learn more about Traces, Channels, and Windows

### Arrange Measurements in Sets

If you arrange measurements to keep the complete set of device measurements in one instrument state, you can save them so that you can later recall a number of measurements with one recall function.

See Pre-configured Measurement Setups for more information.

### Use Segment Sweep

Segment sweep is helpful if you need to change the following settings to characterize a device under test.
The segment sweep allows you to define a set of frequency ranges that have independent attributes. This allows you to use one measurement sweep to measure a device that has varying characteristics.

See Segment Sweep for more information.

Trigger Measurements Selectively

You can use the measurement trigger to make measurements as follows:

- Continuously update only the measurements that have rapidly changing data.
- Occasionally update measurements that have infrequently changing data.

For example, if you had four channels set up as follows:

- Two channels measuring the data that is used to tune a filter
- Two channels measuring the data for the out-of-band responses of the filter

You would want to constantly monitor only the measurement data that you use for tuning the filter. If you continuously update all of the channels, this could slow the response of the analyzer so that you would not be able to tune the filter as effectively.

Note: You must either trigger the infrequent measurement manually or with remote interface commands.

To trigger measurements selectively:
This procedure shows you how to set up two different measurements with the following behavior:

- Channel 1 measurement will continuously update the data.
- Channel 2 measurement will occasionally update the data.

1. Press **Setup > Quick Start**.
2. At the **Quick Start** dialog box, click **Create in new channel**.
3. **Frequency Sweep** dialog box shows. Enter the preferred sweep setting.

**Set Up a Measurement Trigger for Continuous Updates**

1. Press **Trigger > Trigger Source** and select **Internal**.
2. Press **Trigger > Trigger...**.
3. At the **Trigger** dialog box under **Channel Trigger State**, select **Channel 1**, and click **Continuous**.

**Set Up a Measurement Trigger for Occasional Updates**

1. At the **Trigger** dialog box under **Channel Trigger State**, select **Channel 2**, and click **Single, OK**.
2. Press **Trigger > Restart**.

**Update the Measurement**

1. Click on the lower window to make Channel 2 the .
2. On the active entry toolbar, click the type of trigger you set up.
   - Click **Single** if you set up the analyzer for a single sweep per trigger.
   - Click **Groups** if you set up the multiple sweeps per trigger.

**Note:** A trace must be active for you to initiate a trigger for that measurement.

**Automate Changes Between Measurements**

If there are slight differences between the various measurements that you need to characterize a device, you may find that it is faster to change the measurement settings using programming.
Recall Measurements Quickly

The most efficient way to recall measurements is to recall them as a set of measurements (instrument state).

- It only takes a short time longer to recall an instrument state that includes multiple measurements, than it does to recall an instrument state with only one measurement.

- Each recall function has time associated with it. You can eliminate that time by setting up the measurements as a set so you can recall them as a set.

See Save and Recall Files for more information.
Data Transfer Speed

When testing devices remotely using COM or SCPI, the following techniques can be used to transfer data quickly between the analyzer and remote computer, helping you achieve the best measurement throughput.

- **Use single sweep (trigger) mode** to ensure that a measurement is complete before starting a data transfer.
- **Transfer the minimum amount of data** needed. For example, a trace with a few points, using segment sweep rather than a full trace with many linearly spaced points. Also, use markers instead of trace transfers.
- **Choose the REAL data format** to provide the fastest transfer speed when using SCPI programs for automated applications.
- **Use SCPI over LAN** for applications that are automated with SCPI programs.
- **Use COM programs** to provide the fastest transfer speed when using an automated application. See Data Transfer Time specifications.

**Note:** The following data is obsolete, but still serves to illustrate the relative speed between COM and SCPI.

![Data Transfer Speed Comparison](chart.png)
Using Macros

Macros are executable programs that you write, load into the analyzer, and then run from the analyzer. You can have up to 25 macros set up to run on the analyzer.

- How to Setup Macros
- How to Run Macros
- Macro Example

How to Setup Macros

Using Hardkey/SoftTab/Softkey

1. Press Macro > Key Setup > Macro Setup....

In the Macro Setup dialog box:

1. Create an executable program and save it on the VNA hard drive. See SCPI or COM example programs in VBscript.

2. Use a mouse or the front-panel 'down-arrow' to select a blank line below the last entry. (There may be NO entry.)

3. Click Edit to start the Edit Macro Setup dialog.

4. In the Macro Title box, type a descriptive title for your macro.

5. Click Browse.

6. Change Files of Type.

7. Find and select your executable file. Change Files of Type if necessary.

8. Click OK on the Edit Macro Setup dialog.

9. Click OK on the Macro Setup dialog.

10. Press MACRO to run. It may be necessary to first Preset the VNA to see your macro in the menu.
### Macro Setup dialog box help

Macro setup allows you to create up to 25 macros that can be launched from the VNA application.

An external keyboard is required to enter the Macro Title and the Run string parameters.

**To add a Macro**, use a mouse or the front-panel 'down arrow' (NOT the 'Down' key) to select a blank line. Then click **Edit**.

**Macro Title**  Shows the titles that appear in the softkeys and menu when you press the Macro key. These titles are associated with the executable files and should be descriptive so you can easily identify them.

**Macro Executable**  Lists the complete path to the executable file. To follow the example of launching the Keysight VNA Series Home Page, the path to the executable could be "C:\Program Files(x86)\Internet Explorer\iexplore.exe.

**Macro Runstring Parameters**  Lists the parameters that get passed to the program that is referenced in the executable file. Again following the example of launching the VNA Series Home Page, you could assign the runstring parameters "http://www.Keysight.com/find/pna".

**Edit**  Invokes the Macro Edit dialog box.

**Delete**  Deletes the selected macro.

**Up**  Allows you to reorder the macros, moving the selected macro up one line. This order determines how they appear in the VNA Menu and in the softkeys and when you press the Macro front-panel key.

**Down**  Moves the selection down one line in the list of macros.
Macro Edit dialog box help

Macro Title  Add a title that appears in the softkeys and menu.

Macro Executable  Set the complete path to the macro executable file. Click Browse to navigate to the macro executable file and establish the complete path to the file.

Macro run string parameters  Optionally add parameters that are passed to the program referenced in the executable file.

See Macro Setup dialog box

How to Run Macros

Using Hardkey/SoftTab/Softkey

1. Press Macro > Macro<#>.

Macro Example

The following is an example Visual Basic Scripting (vbs) program that you can copy, install, and run on your VNA.

Note: Print these instructions if viewing in the analyzer. This topic will be covered by the Macro Setup dialog box.

1. Copy the following code into a Notepad file.
2. Save the file on the analyzer hard drive in the C:/Documents folder. Name the file FilterTest.vbs.
4. Setup the macro in the VNA
5. Run the macro

' Start copying here
'This program creates a S21 measurement
'It is written in VBscript using SCPI commands

Dim app
Dim scpi
'Create / Get the VNA application
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.SCPIStringParser

' Preset the Analyzer.FPreset presets the setting and deletes all traces and windows
scpi.Execute("SYST:FPReset")
'Create and turn on window 1
scpi.Execute("DISPlay:WINDow1:STATE ON")
'Define a measurement name, parameter
scpi.Execute("CALCulate:PARameter:DEFine:EXT 'MyMeas', 's21'")
'Associate ("FEED") the measurement name ('MyMeas') to WINDow (1), and give the new TRACE a number (1).
scri.Execute("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")

'End copying here
The following calibration types are available in the VNA.

<table>
<thead>
<tr>
<th>Cal Type</th>
<th>Interface</th>
<th>Accuracy</th>
<th>Thru Methods allowed</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRL Family</td>
<td>Basic Cal</td>
<td>Very High</td>
<td>All except Unknown Thru</td>
</tr>
<tr>
<td></td>
<td>Cal All</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SmartCal</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ECal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOLT</td>
<td>Basic Cal</td>
<td>High</td>
<td>All</td>
</tr>
<tr>
<td></td>
<td>Cal All</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SmartCal</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ECal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOLR</td>
<td>Basic Cal</td>
<td>High</td>
<td>Unknown Thru</td>
</tr>
<tr>
<td></td>
<td>Cal All</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SmartCal</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ECal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enhanced Response</td>
<td>Basic Cal</td>
<td>High</td>
<td>Defined Thru or Flush Thru</td>
</tr>
<tr>
<td></td>
<td>SmartCal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>QSOLT (Quick SOLT)</td>
<td>Basic Cal</td>
<td>Medium</td>
<td>Defined Thru or Flush Thru</td>
</tr>
<tr>
<td></td>
<td>SmartCal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-Port Reflection</td>
<td>Basic Cal</td>
<td>High</td>
<td>Not Applicable</td>
</tr>
<tr>
<td></td>
<td>Cal All</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SmartCal</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ECal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open/Short Response</td>
<td>Basic Cal</td>
<td>Low</td>
<td>Not Applicable</td>
</tr>
<tr>
<td></td>
<td>Response Cal</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Learn how to select a default Cal Type.

Other Cal Types (Separate Topic)

- Source and Receiver Power Cals

See other Calibration Topics

TRL Family

Application: Used to accurately calibrate any pair of ports when calibration standards are not readily available.

Note: A Delta Match Cal may be required.

- Learn more about TRL family cal
- For more information on modifying standards, see Calibration Standards.

Calibration Method: Basic Cal, SmartCal, Cal All

General Accuracy: Very High

Standards Required: THRU, REFLECT, LINE or similar combination

Systematic Errors Corrected:
- Directivity
- Source match
- Isolation (see exceptions)
- Load match
- Frequency response transmission tracking
- Frequency response reflection tracking
SOLT

Application: Used to accurately calibrate any number of ports.

General Accuracy: High

Calibration Method: Basic Cal, SmartCal, Cal All, ECal

Standards Required: (SHORT, OPEN, LOAD, THRU) or ECal module

Systematic Errors Corrected (on all ports):
- Directivity
- Source match
- Isolation (see exceptions)
- Load match
- Frequency response transmission tracking
- Frequency response reflection tracking

SOLR

Application: Used to accurately calibrate any two ports.

General Accuracy: High

Calibration Method: Basic Cal, SmartCal, Cal All, ECal

Standards Required: (SHORT, OPEN, LOAD, Reciprocal THRU) or ECal module

Systematic Errors Corrected (on all ports):
- Directivity
- Source match
- Isolation (see exceptions)
- Load match
- Frequency response transmission tracking
- Frequency response reflection tracking
Enhanced Response

Application: Used to calibrate two ports when only measurements in one direction (forward OR reverse) are required. Measurements are faster because a second sweep is NOT required.

- Reflection Standards (OPEN, SHORT, LOAD) are connected to the source port to be calibrated.
- Defined THRU or Flush THRU standard is connected between port pairs.
- Much quicker than SOLT when using a mechanical cal kit. ECal can also be used.

To select Enhanced Response:

For a standard S-parameter Cal, select Cal > Main > Basic Cal...

Then, In the Basic Cal dialog box:

1. Under 'Cal Type', select Enh Response 1-> 2 Enh or Response 2-> 1.

General Accuracy: High
Calibration Method: Basic Cal, SmartCal, ECal
Standards Required: (SHORT, OPEN, LOAD, Defined THRU or Flush THRU)
Systematic Errors Corrected:
- Directivity (source port)
- Source match (source port)
- Isolation (see exceptions)
- Load match (receiver port) - used only to produce transmission tracking term.
- Frequency response transmission tracking (receiver port).
- Frequency response reflection tracking (source port).

QSOLT (Quick SOLT)

Application: Used to quickly calibrate any number of ports. Developed specifically for use with external multiport test sets.

Note: A Delta Match Cal is required to cal test ports that do not have a dedicated reference receiver.

- Reflection Standards (OPEN, SHORT, LOAD) are connected to only ONE of the ports to be calibrated. The lower port number of the ports to be calibrated is selected by default. This can be changed through the Modify Cal / Cal Type setting.
- Defined THRU or Flush THRU standards are connected from the reflection standard port to the remaining ports to be calibrated.
- Much quicker than SOLT when using a mechanical cal kit.
- Based on TRL math.

General Accuracy: Not as high as SOLT
Calibration Method: Basic Cal, SmartCal, ECal
Standards Required: (SHORT, OPEN, LOAD, Defined THRU or Flush THRU)

Systematic Errors Corrected:
- Directivity
- Source match
- Isolation (see exceptions)
- Load match
- Frequency response transmission tracking
- Frequency response reflection tracking

1-Port (Reflection)

Application: Used to accurately calibrate any single test port for reflection measurements only.

Calibration Method: Basic Cal, SmartCal, Cal All, ECal

General Accuracy: High

Standards Required: (SHORT, OPEN, LOAD) or ECal module

Systematic Errors Corrected:
- Directivity
- Source match
- Frequency response reflection tracking

Open / Short Response

Application: Used to quickly calibrate any single test port for reflection measurements only.

Calibration Method: Basic Cal, Response Cal

General Accuracy: Low

Standards Required: OPEN or SHORT

Systematic Errors Corrected:
Frequency response reflection tracking
Thru / Transmission Response (Isolation Optional)

Application: Used to quickly calibrate any pair of test ports for transmission measurements only. Isolation is not usually recommended. Learn more about Isolation

Calibration Method: Basic Cal, Response Cal, and Guided Cal from the ‘Select DUT Connectors page’, check Modify Cal, then click Next.

General Accuracy: Low

Standards Required: THRU

Isolation: One LOAD for each VNA test port.

Systematic Errors Corrected:
- Frequency response transmission tracking
- Isolation
Calibration Thru Methods

- What is a Non-Insertable Device
- Choosing a Thru Method
- Flush Thru
- Adapter Removal
- Swap Adapters and Offset Delay (separate topic)
- Defined Thru
- Unknown Thru
- ECal Thru Method Choices

Other Cal Topics

What is a Non-Insertable Device

To understand the Thru method choices, you must first understand what is meant by "Non-Insertable device". These definitions also apply to ECal modules. Substitute "ECal module" for "device". Then see ECal Thru Method Choices.

A non-insertable device is one whose connectors could NOT mate together. They either do not have the same type of connector or they have the same gender. This also means that the test port cables would not mate together, as in the following diagram.
An **insertable** device is one whose connectors could mate together. They have the same type of connector and opposite, or no, gender. This also means that the test port cables would mate together, as in the following diagram.

Choosing a Thru Method of Calibration

The Thru method is selected from the Cal Wizard. Select the **Modify** checkbox in the **Select DUT Connectors and Cal Kits** dialog box.

**Notes:**

For ECal, the following choices have different meanings. See [THRU methods for ECal](#).

For 4-port calibration, see [How can we measure only 3 THRU connections?](#)

**Choice for Insertable Devices: FLUSH Thru** *(also known as Zero-length Thru)*

When calibrating for an insertable device, the test ports at your measurement reference plane connect directly together. This is called a zero-length THRU, or Flush THRU meaning that the THRU standard has zero-length: no delay, no loss, no capacitance, and no inductance. Your calibration kit may not have a physical THRU standard because it is assumed you have an insertable device and will be using a zero-length THRU.

**Choices for Non-Insertable Devices**

The following methods calibrate for a non-insertable device:

- **Adapter Removal** Accurate, but least convenient.
- **Defined Thru**
- **Unknown Thru Cal** Preferred method.
### Adapter Removal Calibration

The accuracy of the Adapter Removal calibration is very similar to the accuracy of the Unknown Thru calibration. However, the Unknown Thru calibration has fewer connections and therefore has the potential of being more accurate than the Adapter Removal calibration.

Two full 2-port calibrations are performed: one with the adapter connected at port 1, and the other with the adapter connected to port 2. The result of the two calibrations is a single full 2-port calibration that includes accurate characterization and removal of the mismatch caused by the adapter.

Performing an Adapter Removal Cal requires:

- a THRU adapter with connectors that match those on the DUT.
- calibration standards for both DUT connectors.

To select Adapter Removal during a SmartCal, select the **Modify** checkbox in the **Select DUT Connectors and Cal Kits** dialog box. The Cal Wizard will guide you through the steps.

Learn how to perform an **Adapter Removal Cal using ECal**.

### Defined Thru (also known as Known Thru, Cal Kit Thru, ECal Thru, Characterized Thru)

Defined Thru uses the THRU definition that is stored in the Cal Kit file or ECal module. The THRU standard may have worn over time, making it not as accurate as when it was new. Defined Thru is usually more accurate than Adapter Removal, but not as accurate as **Unknown Thru** method.

---

**Notes**

- If performing an ECal, this is the THRU standard in the ECal Module.
- If Defined Thru appears as a potential THRU method in the **SmartCal Wizard**, this means that there is a defined THRU standard in the selected Cal Kit. This could be a **Zero-length Thru**. The SmartCal Wizard will prompt you to connect the required standard when appropriate.

To define a THRU standard in a Cal Kit (not ECal module):

1. Click **Cal > Cal Sets & Cal Kits**.
2. Click **Cal Kit**....
3. Select the Cal Kit from the list.
4. Click **Edit**....
5. Select the Standards tab.

6. Click Add....

7. Select THRU.

8. Complete the dialog box.

The next time you perform a Guided Cal, this Defined THRU standard will be available if the DUT connector types match the THRU standard.

**Known Thru Cal**

Unknown Thru Cal is the preferred THRU method of calibrating the analyzer to measure a non-insertable device.

The Unknown Thru calibration is also known as Short-Open-Load-Reciprocal Thru (SOLR) calibration.

- Very easy to perform.
- Better accuracy than Defined Thru and usually better than Adapter Removal.
- Does not rely on existing standard definitions that may no longer be accurate.
- Causes minimal cable movement if the THRU standard has the same footprint as the DUT. In fact, the DUT can often BE the THRU standard.
- NOT recommended when there is 40 dB or more of combined loss in the Unknown Thru and calibration path. This would NOT allow enough signal to accurately measure at the receiver.

**About the Unknown Thru Process**

SmartCal guides you through the process. Although the following process describes ports 1 and 2, Unknown Thru can be performed on any two ports when using a multiport analyzer.

1. Perform 1-port cal on port 1.

2. Perform 1-port cal on port 2.

3. Connect Unknown Thru between ports 1 and 2.

4. Measure Unknown Thru.

5. Confirm Estimated Delay. This estimate may be wrong if there are too few frequency points over the given frequency span. You can measure the delay value independently and enter that value in the dialog box.
The Unknown Thru Standard

- Can have up to 40 dB of combined loss in the Unknown Thru and calibration path.
- Must be reciprocal: $S_{21}=S_{12}$.
- Must know the phase response to within 1/4 wavelength (see step 5 above).
- Can be the DUT if it meets these conditions.

Unknown Thru Limitations

- Unknown Thru is NOT supported during a TRL calibration from the GUI.
- Unknown Thru CAN be performed using a 4-port PNA-L that does NOT have a reference receiver for each test port. However, a Delta Match Calibration is usually required before the Unknown Thru is measured.
- Cable movement introduces measurement errors.

ECal Thru Method Choices

When the ECal module connectors exactly match the DUT connectors, choose from the following THRU methods:

**ECal Thru as Unknown Thru**  Learn more about Unknown Thru.

- Measures the THRU state of the ECal module as an Unknown Thru.
- The default method when the ECal module connectors match the DUT.
- Very accurate and easy.
- May require a Delta Match Cal.

**Flush Thru (zero-length Thru)**  Learn more about Flush Thru

- Requires an insertable ECal module / DUT.
- Remove the ECal module and connect the two reference planes directly together for a zero-length thru.
- Accurate, but not as easy as ‘ECal Thru as Unknown Thru’.

**ECal (Defined Thru)**
- Measures the THRU state of the ECal module.
- Very easy, but not as accurate as ‘ECal Thru as Unknown Thru’

### Unknown Thru

- Remove the ECal module.
- Then connect a Thru adapter to be measured as Unknown Thru.
- May require a Delta Match Cal.

When the ECal module connectors do NOT exactly match the DUT connectors, choose from the following two methods:

### Adapter Removal

- Can be used with ECal when your DUT is NON-insertable. However, the ECal module MUST be insertable, and the adapter connectors must exactly match the connectors of the DUT as in the following diagram.
- Adapter removal performs 2-port measurements on both sides of the adapter.

### ECal User Characterization

In cases when adapter removal cannot be performed, ECal User Characterization is ALWAYS possible if you have the right adapters. A User Characterization is performed once and stored in the ECal module. However, accuracy is compromised every time you remove, then reconnect, the adapter with the ECal module.
Calibration Wizard

The Calibration Wizard allows you to choose a Calibration method and then perform the calibration.

- How to Start Calibration Wizard
- SmartCal (Guided Calibration)
- Basic Calibration
- Unguided Calibration
- Saving a Calibration

Other Cal Topics

How to start Calibration Wizard

Using Hardkey/SoftTab/Softkey

1. Press Cal > Other Cals > Smart Cal....

The Calibration Window / Channel

During a Guided Calibration, a 'Cal Window' is created for you to view the connection of calibration standards before standards are measured. This Cal Window uses a new Cal channel that is created and duplicates the settings in the channel being calibrated. Correction is ALWAYS OFF for the displayed calibration channel. At the completion of the calibration, the calibration channel and window are deleted.

The measurement of calibration standards can be performed while viewing any VNA window configuration you choose. The Cal Window is appended to your Custom Cal Window setting, and all windows are visible and sweeping below the Cal Wizard before the Measure (cal standard) button is pressed. The windows to be viewed and channels to be swept during the cal process are specified using Remote commands. See an example.

SmartCal (Guided Calibration)
A Guided Calibration automatically determines the calibration type and suggests a calibration kit that matches your DUT connectors.

Guided Calibration can perform the following Cal Types:

- **ALL Cals EXCEPT Open, Short, and Thru Response Cals.**
- **ECal on one or more ports.**
- **TRL - Learn how to do TRL cals**

**Note:** SmartCal DOES allow you to measure calibration standards in any order. However, you must click **Next** and **Back** without measuring standards until you get to the standard you want to measure.

The following dialog boxes appear when performing a Guided calibration on standard channels.

To learn about Calibrations for Application channels, refer to the help topic for the **Application**.

**Select Ports for Guided Calibration** dialog box help

Allows you to select ports to calibrate.

**Select Ports**  All ports are selected by default. User can select which port to calibrate.

**Calibrate Power**  Disabled by default. Learn more.

**Calibrate Uncertainty**  Include measurement uncertainties from calibration.

**Perform Guided Cal**  Check to perform a Guided Power Calibration. Learn more.

**Display Smith Chart During Cal**  When enabled, reflection measurements will be displayed on Smith Chart during calibration.
**Use existing SrcCal when measuring cal standards**  When enabled, and the following conditions are met, the user's existing source power calibration array will be used when acquiring calibration standard data:

- The user is performing a vector calibration only (no power).
- A valid source cal is present in the channel being calibrated.
- The user has elected to enable this feature (via GUI or SCPI).

For greater than 4-port cals, see [External Test Set calibration - Select Cal Type](#).

---

**Select DUT Connectors and Cal Kits** dialog box help

![Select DUT Connectors and Cal Kits](image)

Allows you to select the connector type and Cal Kit for each DUT port to be calibrated.

**Connectors**  To change selection, click the connector field for each DUT port.

If your DUT connectors are **not listed**, you can create your own connector type and calibration kit file. The VNA includes the following example cal kits that can be used as a template. See [Calibration kits](#) for more information.

- If using a gendered (male and female) connector type, select **Type A** as the connector type.
- If using a connectorless device such as on-wafer probes, select **Type B** as the connector type.

**One Connector**  If enabled, then all connectors will be set the same and changing any one of them will change the others.

**DUT Gender**  To select gender of the connector. If the connector has no gender, the pulldown
displays "None".

One Gender  If enabled, then all genders will be set the same, and changing any one of them will change the others.

Cal Kit  Select the Cal Kit to be used to calibrate each test port. The list for each DUT Port displays kits having the same connector type as the DUT.

One Cal Kit  If enabled, then all Cal Kits will be set the same, and changing any one of them will change the others.

Identical ECal models connected?  ECal modules can be distinguished by serial number. This can have implications on your remote SCPI programs.

<table>
<thead>
<tr>
<th>Cal Kit Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>85056K</strong></td>
</tr>
</tbody>
</table>

The 85056K definitions in the analyzer are for 2.92mm standards (2.4mm plus 2.92 adapters). To calibrate 2.4 mm connectors using the 85056K cal kit, select 85056A as the cal kit when you need the sliding load. Otherwise, select 85056D as the cal kit. Both the 85056A and the 85056D kits contain exactly the same standards as the 85056K cal kit WITHOUT the adapters.

TRL

- To perform a TRL Cal, assign a TRL Cal Kit to the lowest port number of each port pair.

- When selecting a TRL Cal Kit on a PNA-L model that requires a Delta Match Cal, and a Global Delta Match Cal is not available, the Cal type will be set to SOLT and a "Could not find a Global Delta Match Cal." message is displayed on the dialog box. If the selected Cal Kit will not support SOLT, the Next button will not be available. Then you must select a different Cal Kit to proceed or Cancel and perform a Global Delta Match Cal.

Modify Cal Type  Check, then click Next, to Modify Cal (Standards AND Thru Method).

For greater than 4-port cals, see External Test Set calibration - Select DUT Connectors.
The current cal kit does not cover the current frequency range of the measurement. Do one of the following to correct the problem:

**Cal Kit Class Category**  Choose from SOLT and TRL. Not available with ECal modules. Click **Edit** to modify the appropriate class assignments.

**Frequency**  Change the frequency range of the active channel.

**Edit**  Modify the class assignments so that a different standard is selected.

**Back**  Select a different Cal Kit that covers the required frequency range.

**Cancel**  Exit the Cal Wizard

**Modify Cal** dialog box help

**Thru #n**  Lists the proposed Thru connections to be made during the calibration process. You can change these Thru connections to better suit your test setup.
- The proposed Thru connections are listed automatically.

- Additional Thru connections can be selected for higher accuracy. Learn more.

### Add Thru

Click to add a Thru connection. Learn more

### Remove Thru

Select a Thru by clicking the "Thru #N" field or the "1st Port / 2nd Port" field. Then click "Remove Thru". This selection is NOT available if the selected Thru is required for the calibration.

### 1st Port / 2nd Port

Click to select the two ports to be included in the Thru connection. The order of the port numbers is not critical.

### Thru Cal Method

Lists the available Thru Cal methods for the specified port pairs.

Learn about the Thru Cal Method choices.

### Cal Type/ Stds

Click to invoke the View / Modify Properties of Cal dialog box

### Do orientation - Appears ONLY if an ECal module is selected for use.

When this box is checked (default) the analyzer automatically senses the model and direction in which an ECal module port is connected to the test ports. If power to the ECal module is too low, it will appear as if there is no ECal module connected. If you use low power and are having this problem, clear this check box to provide the orientation manually.

Orientation occurs first at the middle of the frequency range that you are calibrating. If a signal is not detected, it tries again at the lowest frequency in the range.

### Choose delta match - Available when a Delta Match Cal is required.

- Check, then click Next to invoke the Select Cal Set for Delta Match dialog box.
- Clear - The Cal Wizard uses the Global Delta Match Cal if available.

### View/Detect ECal Characterizations - Appears ONLY if an ECal module is selected for use.

Click to invoke the View ECal Modules and Characterizations dialog box. Displays a list of connected
ECal modules.

**View/Modify Properties of Cal for Ports... dialog box help**

Select calibration type

Another chance to change the Thru method.

Learn about the Thru Cal Method choices.

Advanced

Select the cal method for each connector of the Thru pair.

- **TRL** - Available ONLY when a TRL cal kit was selected for the lowest port number of the port pair.

- **QSOLT**  Available ONLY when "Defined Thru" or "Flush Thru" is selected. "QSOLT 2 <= 1" refers to the receive port 2 and source port 1 (where reflection standards are connected).

- **Enhanced Response** Available ONLY when "Defined Thru" or "Flush Thru" is selected. "EnhResp 2 <= 1" refers to the receive port 2 and source port 1.

- **Transmission Response**  Available ONLY when "Defined Thru" or "Flush Thru" is selected, when Mechanical Cal is selected, and when 2 ports are being calibrated. "TransResp 2 <= 1" refers to the receive port 2 and source port 1.

**View Modify** Click to invoke the Preview and Modify Calibration Selections dialog box.

**Note:** Changes made to the Cal Kit through this dialog are **temporary** that last only for this calibration. To make permanent changes to the Cal Kit, perform Advanced Modify Cal Kits.
Select Cal Set for Delta Match dialog box help

This dialog box appears when a Delta Match Cal is required and Choose delta match was selected.
Learn more.

Displays the Cal Sets that meet the requirements of the Delta Match Cal.

Select either a User Cal Set or Global Delta Match Cal.

If there is no suitable choice for a Delta Match Cal:

1. Click Cancel, then Cancel again to quit the Cal Wizard.
2. Perform either a Global Delta Match Cal or a SOLT cal and save the result in a User Cal Set.
3. Start the Cal Wizard to re-initiate this calibration.
4. Select the Global Delta Match Cal or User Cal Set.

Calibration Steps dialog box help

Note: Calibration can be performed with External triggers. Learn more.

As each new cal step prompt appears, the traces are setup for the next standard measurement. Also, sweeps are triggered continuously until the Measure button is pressed. This way you can view the
integrity of the standard connection.

Prompts for standards to be measured.

**Measure**  Click to measure the standard.

**Done**  Click *after* a standard is re-measured and all measurements for the calibration are complete.

**Next**  Click to continue to the next calibration step. Does **NOT** measure the standard.

If a standard is **NOT** measured, a warning appears and **Done** will not be available after the last Cal step.

**Note:** Smart (Guided) Cal allows you to measure calibration standards in any order. However, you must click **Next** and **Back** without measuring standards until you get to the standard you want to measure.

---

**Sliding Load Measurement** dialog box help

Allows you to measure the sliding load standard.

**To Measure a Sliding Load:**

1. Connect the sliding load to the measurement port following the procedure described in the Calibration Kit User’s and Service Guide.

   **Note:** Do **NOT** set the center conductor to be an interference fit with the center conductor of the testport.

2. Position the sliding element, then click **Measure**. Do not move the sliding element until
measurement is complete.

**Note:** The direction in which the slide moves is NOT important. You can start with the slide at the front and move it backward or start at the back and move it forward. To minimize stability errors it is important to start at one end and move it in the same direction for each of the measurement steps.

3. Measure the sliding load for at least **five** positions for best accuracy.

**Note:** The positions of the sliding element should cover the full length of the slide, but be unequally spaced to reduce the possibility of overlapping data points. Most sliding loads have marks for each slide position.

4. Click **Done** after the final measurement.

5. Disengage sliding load lock (if available), and remove sliding load from the measurement port.

6. Measure the remaining standards.

**How to Verify Sliding Load Calibration Measurements**

Once the calibration is completed, the sliding load can be measured again. The magnitude of the return loss should remain nearly constant as the slide is moved. If the slide spacing was not adequate due to slide position selections, there will be frequency ranges where the magnitude will not remain nearly constant.
Specify delay dialog box help

This dialog appears ONLY when Adapter Removal or Unknown Thru calibrations are performed.

The following values were estimated from the measurement. Most of the time, they are adequate. However, for CW sweep or frequency sweep with large step sizes, the accuracy of the values may be improved.

**Adapter delay** To improve this value, measure and record the delay of the adapter with a dense step size. Enter that value here. The required precision value is the accuracy that is required to characterize the delay value.

**Nominal phase offset** (Waveguide ONLY). To improve this value, measure and record the phase offset of the Waveguide adapter with dense step size. Enter that value here.

When one connector is coax and the other connector is waveguide, the phase offset has an ambiguity of 180 degrees. For consistency, the estimate provided here is always between 0 and 180 degrees. You can change this estimate to any value between -180 degrees and +180 degrees.

**For FCA/Mixer calibrations**, this dialog box appears twice: once for the input frequencies and once for the output frequencies. The values can be slightly different.

Unguided Cal dialog box help

When the **Perform Guided Cal** is unchecked, this dialog box is displayed.

Connect the each standard one by one and click the corresponding standard button.
Save As Calset... Its grayed out when calibration is incomplete. Once the calibration is completed, press this button to save the calibration.

The Calibration Complete dialog box appears after all standards are measured.

**Basic Calibration**

**Using Hardkey/SoftTab/Softkey**

1. Press **Cal > Main > Basic Cal...**

It provides basic calibration. The limitations of basic calibration are:

- one connector type
- one cal kit
- one cal type
- no isolation cal
- no power calibration
**Basic Cal dialog box help**

**Connector Type** Allows user to select single connector type.

**Cal Kit** Allows users to select single Cal Kit. The displayed selection options are according to the selected connector type.

**Port** Allows users to select the port gender. *No Connect* indicates that no port is connected to the VNA.

**Cal Type** Allows users to select calibration type. The displayed selection options are according to the selected cal kit, connector type and gender.

**Save As Calset...** Its grayed out when calibration is incomplete. Once the calibration is completed, press this button to save the calibration.

**Unguided Calibration (Response Cal)**
Unguided Calibration

Using Hardkey/SoftTab/Softkey

1. Press Cal > Main > Other Cals > Response Cal....

The following dialog boxes appear when performing an Unguided calibration:

**Select Calibration Type for Mechanical Standards** dialog box help

Unguided calibration does NOT support cals greater than 2 ports or ECal calibrations.

**Calibration Type Selection**

- 2-Port SOLT
- 1-Port SOL
- TRL - Learn more about TRL
- **Response** - Reflection and Thru (if the active measurement is transmission)

**Cal Configuration**  If not calibrating all test ports, specify which ports to calibrate.

**Back**  Return to Cal Wizard Begin dialog.

**View/Select Cal Kit**  Click to invoke the Select Cal Kit dialog box.

**Note:** When selecting a Cal Kit with an impedance other than 50 ohms (Waveguide = 1 ohm), it is NO LONGER NECESSARY to change the System Impedance setting before performing a calibration. The impedance for the calibration is now derived from the Cal Kit impedance.
Next  Click to continue to Measure Mechanical Standards dialog box.

**Select Cal Type** dialog box help

This dialog box appears ONLY when the selected Cal Type is TRL in the previous dialog box.

**TRL Reference Plane**  Select which standard to use to establish the position of the measurement reference plane.

- **THRU Standard**  Select if the THRU standard is zero-length or very short.

- **REFLECT Standard**  Select if the THRU standard is not appropriate AND the delay of the REFLECT standard is well defined.

**TRL Impedance**

- **LINE Standard**  Specifies that the characteristic impedance of the LINE standard should be used as the system impedance. This ignores any difference between Offset Z0, Offset Loss, and System Z0.

- **SYSTEM Impedance**  Transforms the LINE standard impedance and loss to that of the system impedance for use with the calibration error terms. The TRL calibration will first compute the error terms assuming the LINE standard impedance is the system's characteristic impedance (same as previous LINE selection), then modify the error terms to include the impedance transformation. This should only be used with coax since the skin effect model used is a coaxial model.

Learn how to change System Z0.

To learn to substitute other calibration kits, see Advanced Modify Cal Kits
Select Cal Kit dialog box help

Displays the calibration kit files available for Unguided calibration. Select the desired calibration kit file and click OK.

Choose class type

Edit Class Assignments Allows modification of the selected Cal Kit class assignments.

- To learn to substitute other calibration kits, see Advanced Modify Cal Kits
- Unguided Cal can access only mechanical cal kits #1 through #95, although more cal kits can imported. Learn how.

Note: When selecting a Cal Kit with an impedance other than 50 ohms (Waveguide = 1 ohm), it is NO LONGER NECESSARY to change the System Impedance setting before performing a calibration. The impedance for the calibration is now derived from the Cal Kit impedance.

Measure Mechanical Standards dialog box help

Note: Calibration can be performed with External triggers. Learn more.

Displays the calibration kit file and standards required for the calibration.
Standards may be connected and measured in any order.

Connect the standard to the measurement port and click its associated green button. A check mark indicates the standard has been measured.

If a standard type contains multiple standards, the Multiple Standards dialog box opens to display the multiple standards included in the calibration kit file.

If a sliding load is included in the calibration kit file, the Sliding Load dialog box opens to perform the measurement with the standard.

**Reflection Response** Select EITHER Open or Short standard, then click **Next**.

**Isolation** Requires one load for each test port. Learn more about Isolation. Use when your measurement requires maximum dynamic range (> 90 dB). See also Isolation Portion of 2-Port Calibration.

**Normalize** Available when performing a response cal for any measurement. After Normalize is pressed and the Cal is complete, the data trace is flat when the same physical connections are present on the port. This is similar to Data/Memory, except that the response cal is saved with Cal data and can be applied to other like measurements. Data/Memory is still available after using Normalize. You would usually connect a THRU standard when calibrating a transmission measurement, and a SHORT standard when calibrating a reflection measurement.

**Show Prompts** Check to provide a reminder for the required connection when you click on the standard.

---

**Multiple Standards** dialog box help

Select the standards to be measured.

**Note:** You may see both male and female standards. The Unguided cal has no knowledge of the gender of your connector types. **Choose the gender of your DUT connector**; NOT the test port. Then click OK.

To modify this calibration class to show only one standard, on the Calibration menu, click **Advanced Modify Cal Kits**. Select the Cal kit and click **Edit Kit**. In **Class Assignment**, click **Edit**. Learn more about **Modify Calibration Class Assignments**.
Connect the standard to the measurement port and click its associated button. A check mark in the Acquired box indicates the standard has been measured.

To cover the entire frequency range, you may need to measure more than one standard. The order in which the standards are measured is important. The last standard that is measured will override the others in respect to the frequency range of the standard definition. **Example:** In the case of measuring both a broadband load and a sliding load, you would measure the sliding load last. This is because the frequency range of the sliding load is a subset of the broadband load.

**Saving a Calibration**

SmartCal, ECal, and Unguided Calibrations end with the following dialog box:

**Calibration Completed**

Finish  Save to the channel's calibration register.

Save As User Cal Set  Invokes the Save as User Cal Set dialog box AND save to the channel's calibration register.

Cancel  Calibration is NOT applied or saved.

Learn about Calibration Registers.

Learn about User Cal Sets
Copy Cal Set dialog box help

**Existing Cal Sets** - Lists the previously-saved Cal Set names.

**to** Specify a name for the new Cal Set. Either accept the suggested new name, type a new name, or select a name from the list to overwrite an existing name.

**OK** Saves the Cal Set to the new Cal Set name and exit the dialog message.

Learn about User Cal Sets
Source and Receiver Power Calibration can be performed during a standard S-parameter Guided Calibration. This power cal provides the following enhancements over the standard source and receiver power calibration:

- A source and receiver power cal can be performed for all PNA ports with a single power sensor connection.
- Multiple power sensors can be used to cover wide frequency ranges.
- The receivers are corrected automatically.
- Optionally compensates for an adapter that may be used to connect the power sensor.
- Provides optional match-corrected power measurements.
- Source and Receiver power correction is stored to the Cal Set along with S-parameter correction.

**Note:** A Guided Power Calibration is not accurate when Frequency Offset Mode is enabled.

Learn more about the standard Source and Receiver Power Cals.

In this topic:

- How to perform a Guided Power Cal
- Power Cal Settings dialog box
- Power Cal Settings - Use Multiple Sensors dialog box
- Power Sensor Connection step dialog box
- Correction Methods dialog box
  - Port Subset Correction (Devolve Calibration) - (separate topic)
How to perform a Guided Power Cal

1. In a Standard (S-parameter) channel, setup your measurements (sweep type, frequency range, IFBW, and so forth). A special version of this feature is available on mmWave SMC measurements. Learn more.

2. Connect the Power Meter / Sensor the same as a standard Source Power Cal. Learn more.
   - See Supported Power Meters
   - See Important first-time USB connection note.

3. Start the Cal Wizard, then select Guided (Smart) Cal. Learn how.

On the following Select Ports dialog, check Calibrate source and receiver power, then click Next.

Two Cal Wizard pages later, complete the following dialog.

Power Cal Settings dialog box help
Note: A Use Power Table checkbox (not shown) is available when a mmWave SMC measurement is active. Learn more.

**Power Cal at:** Select the source port for which a Power Calibration will be performed. The source and receiver correction will be transferred to all other sources and receivers involved in the S-parameter measurements.

**Use Multiple Sensors** NOT available with SMC measurements.

Check this box to use one or more power sensors that are configured as PMAR devices. This dialog is replaced with the Multiple Sensors dialog. See following image.

When "Use Multiple Sensors" is cleared (default setting), click Power Meter Settings to configure the power meter.

**De-embed (power sensor) adapter** When the power sensor connector is NOT the same type and gender as the DUT connector for the specified port, then for optimum accuracy, extra cal steps are required to measure and correct for the adapter that is used to connect the power sensor to the reference plane.

Clear this box to NOT compensate for the added adapter.

Check this box to perform extra calibration steps to measure and correct for the adapter.

Then select the **Power Sensor Connector** type and gender of the power sensor. "Ignored" does NOT compensate for the added adapter, just as if the checkbox were cleared.

When this connector matches the DUT connector for the same port, then the PNA assumes that there is no adapter. Extra cal steps are NOT required and the Cal Kit selection is not available.

Otherwise, select the **Cal Kit** to be used to calibrate at the adapter.

See Accuracy Settings below.
"Multiple sensors" are allowed ONLY on standard channels and during a Cal All calibration.

The power sensors that are used as "multiple sensors" MUST be configured PMAR devices.

**Power Cal at:** Select the source port for which a Power Calibration will be performed. The source and receiver correction will be transferred to all other sources and receivers involved in the S-parameter measurements.

**Sensor Settings** Click to start the Sensor Settings dialog, used to **ADD** / Configure an External Device.

**Sensor Grid**

- **Sensor** Select the power sensor and the associated **Start** and **Stop** frequency range.

- **Adapter** When the power sensor connector is NOT the same type and gender as the DUT connector for the specified port, then for optimum accuracy, extra cal steps are required to measure and correct for the adapter that is used to connect the power sensor to the reference plane.
  
  **Clear** this box to NOT compensate for the added adapter.
  
  **Check** this box to perform extra calibration steps to measure and correct for the adapter. Then specify the **Power Sensor Connector** type and gender of the power sensor. When this connector matches the DUT connector for the same port, then extra cal steps are NOT required, and the Cal Kit selection is not available. Otherwise, select the **Cal Kit** to be used to calibrate at the adapter.

- **Remove** Click to remove the power sensor from the list.

- **Add Sensor** Click to add a new line, then click the down-arrow to select a sensor. If a power
sensor does NOT appear in the list, click the **Sensor Settings** button to configure a power sensor.

### Accuracy

**Tolerance** When consecutive power sensor readings are within this value of each other, then the reading is considered settled.

**Max Readings** Sets the maximum number of readings the power sensor will take to achieve settling. Each power reading is "settled" when either:

- Two consecutive readings are within this **Tolerance** value or
- When the **Max Number of Readings** has been met.

The readings that were taken are averaged together to become the "settled" reading.

**Set Power For Best Accuracy** Select to use the power level associated with the best uncertainty for a specific power meter.

### Power Sensor Connection step dialog box help

**Power Level** Set the power level at which the Source Power Cal is to be performed.

It is usually best to perform the Source Power Cal at 0 dBm because the power sensor is calibrated at that level. If 0 dBm is not achievable for your measurement, then set to the power level with the lowest level of measurement noise.

### Correction Methods
Accessing Correction Methods

Using **Hardkey /SoftTab /Softkey**

1. Press **Cal > Main > Correction Methods...**

Using a mouse

1. Click **Response**.
2. Select **Cal**.
3. Select **Correction Methods...**.

Turn OFF Match Correction

During a Guided Power Cal, the match between the power sensor and the VNA source port is measured. The source power correction array is compensated to account for the measured mismatch. In addition, the reference receiver measurement is also compensated to account for the mismatch of the DUT.

**How to turn OFF match correction:**

Click **Cal > Main > Correction Methods...** then click on the **Exclude all ports** button.
The following dialog applies to the Spectrum Analyzer, Modulation Distortion, and Modulation Distortion Converters applications only.

**Power Wave Correction Type**

**Match Corrected** - These waves are calculated from the actual waves and the S-parameters of the DUT to determine the waves if the VNA test ports were perfectly matched. However, you may not want match correction in the following cases:

- When making non-traditional measurements, such as high-power or multiport configurations. Because of added components or reconfigurations, the mismatch measurement may not be valid.
- When you have a remote program that already accounts for the match effects of the sensor.

**Actual Waves** - These are the full error corrected actual waves at device reference planes.

**Response Corrected** - These are raw measurements scaled with the response terms and do not include any match correction.

**Selected Ports Will Correct Power Using Actual Waves** (Spectrum Analyzer, Modulation Distortion, and Modulation Distortion Converters applications only) - This is the same as **Match Corrected**. See description above.

### 8-term Error Correction Model

**Enable 8-term Error Correction Model** - Check to enable the 8-term error correction model for measurement calibration.

The VNA may correct S-Parameters by using either the 12-term or the 8-term error correction model. (The 12-term model is also known as the 10-Term model if the two crosstalk terms are ignored.) Changing the model will affect the S-Parameter and wave measurements.

When measuring a DUT, the VNA will stimulate the DUT in the forward and reverse direction, measure the waves, then calculate the S-Parameters of the DUT. A 2-port VNA forward flow diagram is shown below and will be used to compare the two correction models.

**Forward Direction Block Diagram**

For an 8-term model, the forward direction state will measure the waves \((a_0, b_0, a_3, b_3)\) and will calculate the corrected waves \((a_1, b_1, a_2, b_2)\) at the DUT test ports. For a 12-term model, the VNA will measure the waves \((a_0, b_0, b_3)\) and will calculate the corrected waves \((a_1, b_1, a_2, b_2)\) at the DUT test ports. The 12-term model does not need to measure the \(a_3\) wave because its value is calculated from the \(b_3\) wave and the Port 2 load match characterized during calibration.

The following table describes the tradeoffs between the 8-term and 12-term models:
<table>
<thead>
<tr>
<th>Model</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>12-Term</td>
<td>Measures fewer waves so it may be faster and have lower noise.</td>
<td>VNA load match changes after calibration will result in errors.</td>
</tr>
<tr>
<td>8-Term</td>
<td>VNA load match changes after calibration will not result in errors.</td>
<td>Measures more waves so it may be slower and have higher noise.</td>
</tr>
</tbody>
</table>

The VNA uses the 12-term model by default because it is fast, low noise, and typically the VNA load match is very stable. However, there are several cases where the 8-term model is a good choice:

- Load Pull measurements vary the load match to determine its effect on the DUT's S-parameters. The 8-term error model is immune to these intentional changes in VNA load match.

- High power test setups often include external couplers, preamps, and attenuators. During calibration the preamps and attenuators may be removed to avoid damaging the cal standards and to optimize signal levels. These components are reattached following calibration. If the components are removed and added on the VNA-side of the couplers, the 8-term model will correct for any changes.

- A preamp may be added behind the couplers of one port to increase the output power. When that port is used as the load-side of the measurement, the load match presented by the output of the preamp may not be very stable over time. The 8-term error model will correct for this load match instability.

Port subset correction

**Enable Port Subset Correction** - Enabling port subset correction to reduce the number of corrected ports.

**Select Ports in Subset** - Selects which ports should be included in a full N-port correction. Unselected ports will be corrected on a “best effort” basis: In other words, these ports will be corrected with an enhanced response calibration if the error terms are available in the calset.

**Clear All** button - De-selects all ports from correction. The button will change to **Select All** to include all ports for correction.

To learn more about Port subset correction, go to Port Subset Correction (Devolve Calibration).

**Correction Results**

Correction results are displayed in this table. This is the same table displayed when the user right-clicks on the Cal pane of the status bar at the bottom of the screen. The following indicate the correction applied:

- **F1** - 1-port calibration.
**eR** - Enhanced response.

**Blank** - No correction applied.

**F** - Multiport calibration.
"Cal All" allows you to calibrate multiple channels in a single calibration session. This not only reduces the number of connections that need to be made, but also the number of cal standard measurements that must be performed.

**Note:** Beginning with the A.12.80 release, Cal All has been extended in order to deal with the new Low Frequency Extension option. If a user has a mixture of LFE and non-LFE channels and they would like to use Calibrate All to calibrate them at the same time, two calibration channels are created to account for the hardware differences between the two situations. When using the GUI or COM to set calibration and stimulus conditions, the settings are applied to both calibration channels. With SCPI, the user can query the primary guided calibration channel using SYST:CAL:ALL:GUID:CHAN:VAL?. This will return the primary calibration channel. When subsequent Guided Cal commands are used, settings will be transferred to the second calibration channel. If there is a desire to set these settings separately, the user should query for all Cal All Calibration channels with SYST:CAL:ALL:GUID:CHAN:LIST?. The user should set values for the primary calibration first, and then secondary calibrations. When initializing the calibration and acquiring steps, use the primary cal all channel number.

In this topic:

- Features
- Limitations
- How to perform a Cal All Channels Calibration
  - Select Channels dialog
  - Measurement Class Cal Properties dialog
  - Setting Up an Independent Power Calibration
  - Split Cal and Independent Calibration Channels
  - Calibration Attenuator Settings dialog
  - Select DUT Connectors and Cal Kits dialog
  - Power Cal Settings dialog
  - Cal Steps dialog
  - Finish
Other Cal Topics

See Also

- Cal All Channels Calibration (links to SCPI Cal All Channels programming examples)
- COM Examples (links to COM programming examples)

Features

Cal All offers a single, optimized calibration procedure for all channels (with some limitations, see below). The optimizations include:

- Minimizing the number of physical connection of standards.
- Minimizing the number of power meter calibration sweeps.
- User-settable power levels for S-Parameter as well as power calibration steps.
- Accounting for different switch and attenuator settings among different channels. This reduces the number of measurements required to characterize different switch/attenuator settings (channel setup differences).
- Cal All will produce the same number and format of Cal Sets (error terms) that would be realized had the calibrations been performed one at a time.
- Calibrate External Sources that are connected to the analyzer using Configure an External Source.
- ECal Extrapolation is supported on Swept IMD and IMDx channels.
- mm Wave measurements are supported.

Limitations

- VMC channels are NOT supported.
- SMC+ Phase with phase enabled is supported using a known delay mixer or a phase reference cal set. S2P file characterized mixers are NOT supported.
- For non-IMD channels, the wideband IF path is used during Cal All. Therefore, non-IMD channels that use the narrowband IF path (for Narrow band pulse measurements for example) or have manually-selected IF frequencies will not be properly calibrated.
- Starting the VNA in Multiport mode is NOT supported.
- Cal All is performed at one IFBW.
All channels that are calibrated are forced into stepped sweep mode.

All channels to be calibrated MUST have the same cal reference plane. In other words, Cal All cannot compensate for any path changes that occur external to the analyzer.

Cal All cannot be used for the following configurations with Swept IMD and IM Spectrum (for these configurations use the Cal Wizard instead):

- Port 3 as input and port 4 as output using an external combiner.

- Using an N5242B with external combiner that has a large path loss difference from the internal combiner.

Cal All nominally uses preset conditions for path configuration switches and attenuators during the calibration (unless you change them via the Cal All Wizard pages). The benefit of the Cal All approach is that it measures all the switch path differences between channels and compensates the error terms appropriately for each channel. However, the “Port 2 Source” configuration setting is not something that Cal All can automatically remove. Therefore, if you are attempting to calibrate a channel with an alternate Port 2 Source configuration, you will need to modify the Cal All settings as follows:

If the Configuration is set to **Src2 out Port 2**, the Cal All settings must be changed to match the RF Path configuration settings for the channel. Use the Mechanical Devices dialog in the Cal All Wizard for this purpose. The dialog on the left shows the default Cal All settings at Preset and the channel settings when Configuration is set to **Src2 out Port 2**. The dialog on the right shows the edited (highlighted) Cal All settings to match the channel settings.
Cal All power offsets must match the power offsets of the client channels. If not, the Source Power Correction may not be correct.

How to perform a Cal All Channels Calibration

Using Hardkey / SoftTab / Softkey

1. Press **Cal > Other Cals > Cal All...**

Selected Channels dialog box help

1. Check the channels to be calibrated.

2. Check the ports to be calibrated. Click on the **Apply to all channels** button to apply the port selections to all channels.
3. Click **Next>**

**Note:** To perform an LO power cal for a mixer channel, set the LO port to a VNA or external source in the Mixer Setup dialog. Then select that port in this dialog.

### Measurement Class Cal Properties dialog box help

Confirm or change the following unique cal properties for each channel to be calibrated. Click a link to learn about these properties.

The properties with *(NOT available in Cal All)* are NOT available in a Cal All calibration as they are in a stand-alone calibration.

#### Differential IQ

<table>
<thead>
<tr>
<th><strong>UI Setting</strong></th>
<th><strong>Property</strong></th>
<th><strong>Value</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Extra Power Cals</td>
<td>&quot;Enable Extra Power Cals&quot;</td>
<td>&quot;Port 1&quot;, &quot;Port 2&quot;, &quot;Port 3&quot;, &quot;Port 4&quot;, &quot;Port 1 Src2&quot;, and/or Source3</td>
</tr>
<tr>
<td>Independent Calibration Channels</td>
<td>&quot;Independent Calibration Channels&quot;</td>
<td>Any selected channels that support SmartCal.</td>
</tr>
</tbody>
</table>

### Spectrum Analyzer
## Programming

<table>
<thead>
<tr>
<th>UI Setting</th>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calibration Span</td>
<td>Calibration Span</td>
<td>&quot;User Span&quot; or &quot;Instrument Span&quot;</td>
</tr>
<tr>
<td>Calibration Points</td>
<td>Calibration Points</td>
<td>Integer indicating number of calibration points.</td>
</tr>
<tr>
<td>Enable Extra Power Cals</td>
<td>&quot;Enable Extra Power Cals&quot;</td>
<td>&quot;Port 1&quot;, &quot;Port 2&quot;, &quot;Port 3&quot;, &quot;Port 4&quot;, &quot;Port 1 Src2&quot;, and/or Source3</td>
</tr>
<tr>
<td>Independent Calibration Channels</td>
<td>&quot;Independent Calibration Channels&quot;</td>
<td>Any selected channels that support SmartCal.</td>
</tr>
</tbody>
</table>

## Gain Compression

<table>
<thead>
<tr>
<th>UI Setting</th>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Extra Power Cals</td>
<td>&quot;Enable Extra Power Cals&quot;</td>
<td>&quot;Port 1&quot;, &quot;Port 2&quot;, &quot;Port 3&quot;, &quot;Port 4&quot;, &quot;Port 1 Src2&quot;, and/or Source3</td>
</tr>
<tr>
<td>Independent Calibration Channels</td>
<td>&quot;Independent Calibration Channels&quot;</td>
<td>Any selected channels that support SmartCal.</td>
</tr>
</tbody>
</table>

## GCX (Gain Compression Converters)

<table>
<thead>
<tr>
<th>UI Setting</th>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Extra Power Cals</td>
<td>&quot;Enable Extra Power Cals&quot;</td>
<td>&quot;Port 1&quot;, &quot;Port 2&quot;, &quot;Port 3&quot;, &quot;Port 4&quot;, &quot;Port 1 Src2&quot;, and/or Source3</td>
</tr>
<tr>
<td>Independent Calibration Channels</td>
<td>&quot;Independent Calibration Channels&quot;</td>
<td>Any selected channels that support SmartCal.</td>
</tr>
<tr>
<td>Split Cal</td>
<td>&quot;Split Cal&quot;</td>
<td>&quot;true&quot; or &quot;false&quot;</td>
</tr>
</tbody>
</table>

## Noise Figure and NFX

<table>
<thead>
<tr>
<th>UI Setting</th>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal Method</td>
<td>&quot;Noise Cal Method&quot;</td>
<td>&quot;Scalar&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Vector&quot;</td>
</tr>
</tbody>
</table>
| Noise Tuner (used with Vector Cal Method) | "AutoOrient Tuner" | "true" - AutoOrient Tuner  
"false" - Do NOT autoOrient |
|-----------------------------------------|-------------------|--------------------------------------------------|
| When "AutoOrient Tuner" = false (Manually Orient) | "Tuner In"  
"Tuner Out" | Tuner port connected to DUT input  
Tuner port connected to DUT output  
For example: "A" or "B" |
| Receiver Characterization Method | "Receiver Characterization Method" | "Use Power Meter"  
"Use Noise Source" |
| Note: When selecting "Use Power Meter", the power cal step must be completed first when acquiring the calibration. Furthermore, the power cal port must be the same as the input port on the NF channel. |
| Specify the Noise Source connector - used for the 1 port cal if needed. | "Noise Source Connector" | String name of valid connector type.  
To query valid connector types:  
COM: ValidConnectorTypes  
SCPI:  
SENS:CORR:COLL:GUID:CONN:CAT? |
| Specify the Noise Source cal kit - used for the 1 port cal if needed. | "Noise Source CalKit" | String name of valid cal kit.  
To query valid cal kits names:  
COM:GetCompatibleCalKits  
SCPI:  
SENS:CORR:COLL:GUID:CKIT:CAT? |
| Specify ENR file | "ENR File" | String path and filename of the ENR file. |
| Include Power Calibration | "Include Power Calibration"  
"Enable Extra Power Cals"  
"Port 1 Src2 Cal Power" | "true" or "false"  
"Port 1", "Port 2", "Port 3", "Port 4", and/or "Port 1 Src2"  
Integer indicating valid calibration power. For example, "-20" indicates a power level of -20 dBm. Only valid if an independent port calibration on Port 1
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Noise Tuner&quot;</td>
<td>Tuner description in the form of <code>&lt;model&gt; ECal &lt;serialNumber&gt;</code>, as in “N4691-60004 ECal 10775”.</td>
<td></td>
</tr>
<tr>
<td>&quot;Force Thru Adapter De-embed&quot;</td>
<td>“true” – Always measure and de-embed the thru adapter. “false” – Only measure and de-embed the thru adapter if no flush thru is possible.</td>
<td></td>
</tr>
<tr>
<td>&quot;Force Noise Source Adapter De-embed&quot;</td>
<td>“true” – Always measure and de-embed the noise source adapter. “false” – Only measure and de-embed the adapter if no direct connection is possible.</td>
<td></td>
</tr>
<tr>
<td>&quot;Cal Receiver Port Only at Low Frequencies&quot;</td>
<td><em>Only available when doing scalar downconverter calibrations on PNAs with a max frequency of 40 GHz or higher.</em> “true” – only make receiver port measurements during cal at downconverted frequencies. “false” – make receiver port measurements during cal at both input and output frequencies.</td>
<td></td>
</tr>
<tr>
<td>&quot;Noise Source Temperature&quot;</td>
<td>The cold temperature of the noise source, in Kelvin.</td>
<td></td>
</tr>
<tr>
<td>USB Noise Source</td>
<td>&quot;USB Noise Source&quot;</td>
<td>“usbNsrcId” (example: “U1831C MY12345678”)</td>
</tr>
<tr>
<td>UI Setting</td>
<td>Property</td>
<td>Value</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Max Product Order</td>
<td>&quot;Max Product Order&quot;</td>
<td>Integer indicating desired product order.</td>
</tr>
<tr>
<td>Include 2nd Order</td>
<td>&quot;Include 2nd Order&quot;</td>
<td>&quot;true&quot; or &quot;false&quot;</td>
</tr>
<tr>
<td>Exclude 2nd Order</td>
<td>&quot;Exclude Channels From 2nd Order&quot;</td>
<td></td>
</tr>
<tr>
<td>Enable Extra Power Cals</td>
<td>&quot;Enable Extra Power Cals&quot;</td>
<td>&quot;Port 1&quot;, &quot;Port 2&quot;, &quot;Port 3&quot;, &quot;Port 4&quot;, &quot;Port 1 Src2&quot;, and/or Source3</td>
</tr>
<tr>
<td>Independent Calibration Channels</td>
<td>&quot;Independent Calibration Channels&quot;</td>
<td>Any selected channels that support SmartCal.</td>
</tr>
<tr>
<td>Response only</td>
<td>(NOT available in Cal All)</td>
<td></td>
</tr>
<tr>
<td>Center Frequencies only</td>
<td>(NOT available in Cal All)</td>
<td></td>
</tr>
</tbody>
</table>

**MOD (Modulation Distortion) and MODX (Modulation Distortion Converters)**

<table>
<thead>
<tr>
<th>UI Setting</th>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Extra Power Cals</td>
<td>&quot;Enable Extra Power Cals&quot;</td>
<td>&quot;Port 1&quot;, &quot;Port 2&quot;, &quot;Port 3&quot;, &quot;Port 4&quot;, &quot;Port 1 Src2&quot;, and/or Source3</td>
</tr>
<tr>
<td>Independent Calibration Channels</td>
<td>&quot;Independent Calibration Channels&quot;</td>
<td>Any selected channels that support SmartCal</td>
</tr>
<tr>
<td>Calibration Span (MOD only)</td>
<td>&quot;Calibration Span&quot;</td>
<td>&quot;User Span&quot; or &quot;Instrument Span&quot;</td>
</tr>
<tr>
<td>Calibration Points (MOD only)</td>
<td>&quot;Calibration Points&quot;</td>
<td>Integer indicating number of calibration points</td>
</tr>
<tr>
<td>Enable Phase Correction (MODX only)</td>
<td>&quot;Enable Phase Correction&quot;</td>
<td>&quot;true&quot; or &quot;false&quot;</td>
</tr>
<tr>
<td>Phase Correction Method (MODX only)</td>
<td>&quot;Phase Correction Method&quot;</td>
<td>&quot;Use Mixer Delay&quot;</td>
</tr>
<tr>
<td>Calset (MODX only)</td>
<td>&quot;Calset&quot;</td>
<td>String of Cal Set Name</td>
</tr>
</tbody>
</table>

**PN (Phase Noise)**
## Programming

<table>
<thead>
<tr>
<th>UI Setting</th>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Smart Cal Order</td>
<td>&quot;Use Smart Cal Order&quot;</td>
<td>true or false</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cal All optimizes the order of the calibration standards for acquisition to improve calibration results. This may result in a different order than a single channel Smart Calibration. Use “True” to have Cal All attempt to preserve the order of Smart Cal.</td>
</tr>
<tr>
<td>Enable Extra Power Cals</td>
<td>&quot;Enable Extra Power Cals&quot;</td>
<td>&quot;Port 1&quot;, &quot;Port 2&quot;, &quot;Port 3&quot;, &quot;Port 4&quot;, &quot;Port 1 Src2&quot;, and/or Source3</td>
</tr>
<tr>
<td>Independent Calibration</td>
<td>&quot;Independent Calibration Channels&quot;</td>
<td>Any selected channels that support SmartCal</td>
</tr>
<tr>
<td>Channels</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Calibration Span</td>
<td>&quot;Calibration Span&quot;</td>
<td>&quot;User Span&quot; or &quot;Instrument Span&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If “User Span” is selected, Cal All will calibrate +/- 30 MHz around the carrier frequency (calibrated at a 10 MHz spacing).</td>
</tr>
<tr>
<td>Calibration Points</td>
<td>&quot;Calibration Points&quot;</td>
<td>Integer indicating number of calibration points</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note that this option is only available if “Instrument Span is selected. If “User Span” is selected, 7 points will be used (every 10 MHz).</td>
</tr>
<tr>
<td>Programming</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>UI Setting</td>
<td>Property</td>
<td>Value</td>
</tr>
<tr>
<td>Enable Extra Power Cals</td>
<td>&quot;Enable Extra Power Cals&quot;</td>
<td>&quot;Port 1&quot;, &quot;Port 2&quot;, &quot;Port 3&quot;, &quot;Port 4&quot;, &quot;Port 1 Src2&quot;, and/or Source3</td>
</tr>
<tr>
<td>Independent Calibration Channels</td>
<td>&quot;Independent Calibration Channels&quot;</td>
<td>Any selected channels that support SmartCal.</td>
</tr>
</tbody>
</table>

SMC (VMC is NOT offered)

| Programming |
|-----------------|-----------------|-----------------|
| UI Setting      | Property        | Value           |
| Enable Phase Correction | "Enable Phase Correction" | "true" or "false" |
| Phase Correction Method | "Phase Correction Method" | "Use Mixer Delay" |
| | | "Use Characterized Mixer" |
| | | "Use Receiver Characterization Calset" |
| Mixer Delay | "Mixer Delay" | Real number indicating delay value. |
| Receiver Characterization Cal Set | "Calset" | String of Cal Set Name |
| Enable Extra Power Cals | "Enable Extra Power Cals" | "Port 1", "Port 2", "Port 3", "Port 4", "Port 1 Src2", and/or Source3 |
| Independent Calibration Channels | "Independent Calibration Channels" | Any selected channels that support SmartCal. |
| Split Cal | "Split Cal" | "true" or "false" |
| Characterized mixer (s2p file) (NOT available in Cal All) | | |

Standard Channel
<table>
<thead>
<tr>
<th>UI Setting</th>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include Power Calibration</td>
<td>&quot;Include Power Calibration&quot;</td>
<td>&quot;true&quot; or &quot;false&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;Enable Extra Power Cals&quot;</td>
<td>&quot;Port 1&quot;, &quot;Port 2&quot;, &quot;Port 3&quot;, &quot;Port 4&quot;, &quot;Port 1 Src2&quot;, and/or Source3</td>
</tr>
</tbody>
</table>
|                          | "Port 1 Src2 Cal Power"                  | Integer indicating valid calibration power. For example, 
|                          |                                          | 
|                          |                                          | 
|                          |                                          | 
|                          |                                          | Only valid if an independent port calibration on Port 1 Src2 is selected. |
| Independent Calibration  | "Independent Calibration Channels"       | Any selected channels that support SmartCal.                        |
| Channels                 |                                          |                                                                      |

The power cal is optional only if none of the selected channels require a power cal.

### Setting Up an Independent Power Calibration

#### Independent Power Calibration

Several applications control internal and external sources in a mode that is often decoupled from the span over which the receivers are swept. This includes, for instance, Differential IQ, and Spectrum Analyzer. Cal All can add a power calibration for any port (including external sources) over an arbitrary frequency span defined by the user.

For all ports selected for an independent power calibration (except for Port 1 Src 2, see below), a power sensor calibration measurement is performed. The resulting source match correction terms are added to the calsets for ALL channels selected for the Cal All calibration. The power calibrations used in Cal All have all the same features as typical power calibrations. These include the ability to specify power offsets, the power at which the calibration is completed, and the ability to use multiple power sensors (note that using multiple power sensors is a feature only available on regular PNA ports – that is, not external sources or auxiliary ports).

#### Port 1 Src2 Calibration

The Port 1 Src2 calibration is a special case: In this case, a calibration is requested for situations where the Port 1 Bypass Switch is in “Combiner Path” mode and either the “Port 3 Bypass Switch” (4-port PNA-X) or the “Source 2 out 1 Bypass Switch” (2-port 2-source PNA-X) is also in “Combiner Path”. Therefore, if a user requests a calibration of this port:
1. Port 1 will also be calibrated at the same frequencies in the combiner path (*note: The user can still elect to add more Port 1 independent cal ranges, but this will be done in “Bypass” mode, not “Combiner mode.”

2. The calibrations for both Port 1 and Port 1 Src2 will be done with both sources on, but offset by 10MHz (to more accurately characterize the behavior during use).

3. These calibrations will ONLY be applied to user channels in which both the Port 1 Bypass Switch and either the Port 3 or Source 2 Out 1 Bypass Switches are set to “Combiner Path.”

Important Notes:

- If a user selects an independent power calibration on a port also used for the typical S-parameter calibration, then the frequencies calibrated will be in addition to those frequencies used via the typical calibration (a separate measurement will be completed at the time).

- Previously a user could select a similar calibration of “LO” ports for application channels (for instance, in a GCX measurement with Port 3 used for LO1, a user can select (via GUI or remote) to do an “LO1” calibration. This will do a power sensor calibration of LO1 (port 3) at the frequencies specified by the channel. This is still available, but if the user selects to do an Independent Power Calibration instead, this will override the selection by the channel.

- If the user has a mixture of LFE (Low Frequency Extension ) and non-LFE channels the calibration will be done by default in the NON-LFE configuration (this is because most applications do not support LFE and external sources and sources like Src2 Out 1 do not have LFE paths). Instead, if it’s desired to do these calibrations in LFE mode, the user must select only LFE channels over which to do the cal all calibration.

- If setting up the calibration remotely, it is not necessary to set the power cal state for the extra power calibrations (for example, “SENS200:CORR:COLL:GUID:PSEN2 ON”). These state commands should only be used to specify the power cal port for the main calibration.

Independent Power Calibration for Cal All Setup

1. Start the **Cal All Wizard**.
2. In the **Measurement Class Cal Properties** dialog, enable Independent Power Calibration by selecting the port(s) to cal then clicking **OK**.
3. If Port 1 Scr2 is a selected port, the Port 1 Src2 Cal Power level is displayed allowing a calibration power level to be entered.

4. To enter the Port 1 Src2 Cal Power level, click in the field then enter a valid calibration power level.

5. In the Power Cal Settings for Port<n> dialog, select the port from the Power cal at dropdown.
6. To add a new Cal range, click the New button.
7. Specify a frequency range for each Range Name by clicking in the corresponding Settings field then clicking on the Edit button that appears.
8. Enter the **Start** and **Stop** frequency and the number of **Points** then click **OK**.
9. Continue the Cal All process by clicking on the **Next** button.

---

### Split Cal and Independent Calibration Channels

Beginning with firmware revision A.13.67.xx, there are two new attributes called **Split Cal** and **Independent Calibration Channels** that allow channels to perform their own calibrations.

#### Split Cal

The Split Cal applies to SMC and GCX channels. If selected, these channels will perform their own calibrations, performing two 1-port calibrations (no thru).

#### Independent Calibration Channels

The Independent Calibration Channels is a comma-separated channel list. Any selected channels will perform their own SmartCal instead of importing the calibration from the SmartCal performed on the special Cal All channel (typically Channel 200).

**Note:** Channels that do not support their own SmartCal cannot be Independent Calibration Channels. They will be filtered out if selected as Independent Calibration Channels.

### Example of Cal All Process

The following is an example showing the typical Cal All process using three standard channels:

- **Ch 1:** 1-2 GHz, 101 points
- **Ch 2:** 2-3 GHz, 101 points
- **Ch 3:** 3-4 GHz, 101 points

Cal All process without Independent Calibration Channel selections:

1. Creates a Cal All channel (typically Channel 200), which goes from 1-4 GHz, 303 points.
2. Cal All tells the Cal All channel to perform a SmartCal.

3. When the calibration is done, the SmartCal on the Cal All channel computes the error terms, and passes them to Ch1, Ch2, and Ch3.

Cal All process with channel 3 selected as an Independent Calibration Channel:

1. Creates a Cal All channel (typically Channel 200), which goes from 1-3 GHz, 202 points.

2. Cal All tells the Cal All channel to perform a SmartCal.

3. Cal All tells Ch 3 to perform its own SmartCal.

4. When the calibration is done:
   a. Channel 200 SmartCal computes error terms, and passes them to Ch1 and Ch2.
   b. Ch 3 computes its own error terms (since it performed its own SmartCal).

Split Cal and Independent Calibration Channels Setup

1. Start the Cal All Wizard.

2. In the Measurement Class Cal Properties dialog, enable Split Cal by selecting True from the drop-down menu:

   ![Measurement Class Cal Properties](image)

3. Select the channels for Independent Calibration Channels as shown below:
3. For any channels that are 1) Split Cal (SMC/GCX), 2) Cal Receiver Port Only at Low Frequency (banded noise only), or 3) Independent Calibration Channels, note the following:

   a. These channels will be performing their own SmartCal calibrations and will NOT inherit a calest from the Cal All calibration.

   b. Connection lists will be organized such that even if several channels are performing the same connection as part of their calibration the user will be prompted only once. For example, if channels 2, 3, and 200 (CalAll channel) will be needing a power sensor on port one, then the user will be prompted once for this connection, and channels 2, 3, and 200 will take power sweeps consecutively.

   c. Settings made from the GUI apply to all active calibrations. If a user wants to set different calibration settings, use the remote UI (with the appropriate channel number). The only exception is that the user may choose specific calibration powers (if allowed) for each channel performing its own Calibration. This is accomplished using the SENSE:CORR:COLL:GUID:PSEN:POW:LEVEL SCPI command and the PowerCalibration:PowerLevel COM command. From the GUI, this is accomplished in the following Power Cal Page:

4. For any channels that are 1) Split Cal (SMC/GCX), 2) Cal Receiver Port Only at Low Frequency (banded noise only), or 3) Independent Calibration Channels, note the following:

   a. These channels will be performing their own SmartCal calibrations and will NOT inherit a calest from the Cal All calibration.

   b. Connection lists will be organized such that even if several channels are performing the same connection as part of their calibration the user will be prompted only once. For example, if channels 2, 3, and 200 (CalAll channel) will be needing a power sensor on port one, then the user will be prompted once for this connection, and channels 2, 3, and 200 will take power sweeps consecutively.

   c. Settings made from the GUI apply to all active calibrations. If a user wants to set different calibration settings, use the remote UI (with the appropriate channel number). The only exception is that the user may choose specific calibration powers (if allowed) for each channel performing its own Calibration. This is accomplished using the SENSE:CORR:COLL:GUID:PSEN:POW:LEVEL SCPI command and the PowerCalibration:PowerLevel COM command. From the GUI, this is accomplished in the following Power Cal Page:

**Note:** Some channels do not support setting the cal power level. If this is the case, then
the field will be disabled.

5. On the last **Cal All Wizard** page, the **Description** column specifies whether the channel is being calibrated by Cal All or performing its own calibration: If it is being calibrated by Cal All, it displays which Cal All channel is performing the calibration.

**Note:** If a user chooses to **Enable Extra Power Cals**, then the only client channels that are participating in Cal All as normal will inherit the corresponding power calibration terms (this is most often used with SA, which cannot be an Independent Calibration Channel).
This dialog shows the Power, Attenuator, and IFBW settings for the Cal All calibration. The default values for the Cal All session are the preset values of a standard S-parameter channel. These values are not necessarily the same as those of the channels that are selected for calibration. When there are differences in measurement path (switch) settings between the Cal All channel and the selected channels, these differences are detected by Cal All and additional measurements are made for each path condition. These additional measurements allow Cal All to produce error terms appropriate for each of the selected channels. In general, the Cal All session should be performed at a power level that is high enough to prevent noise in the error terms. However, an increase in power could cause compression or damage to the analyzer receivers. The following settings allow you to increase the power level ONLY during the Cal All session.

**Power Limit (Disable)**

Cal All shows you when power limits are enabled. This setting provides you a convenient way to TEMPORARILY disable these limits in order to take advantage of the power settings available in Cal All. If power limits are on, your DUT is probably a high-gain device and the attenuator settings in your channels are high resulting in lower power at the cal reference plane. This lower signal can result in noisier measurements during the acquisition of cal. This situation is precisely what Cal All is intended to improve. Cal All allows you to configure the calibration conditions for better signal-to-noise performance during the cal while leaving your DUT conditions alone. You can elect to clear the “Disable Power Limits during cal” checkbox when you prefer to calibrate at a higher power level than is allowed by your limit. The limit is restored after the Cal All session.

**Source / Receiver Attenuator**

By default, the Cal All calibration is performed with Source and Receiver attenuators set to 0. Change the Source or Receiver attenuator settings when external hardware (such as a booster amplifier) would cause the analyzer receivers to be compressed or damaged.

You may also want to change the attenuator or path configuration settings to force the cal channel to match settings of the selected channels. If all of the selected channels are set to identical hardware settings, it may be better to apply these settings to the cal channel. For example, if your channels all use a 5 or 10 dB attenuator step at port 1, you might elect to change the Cal All
channels to use the same low attenuator settings. This will result in the cal measurements being made under the same path conditions as the channel and it will eliminate the need to mathematically compensate for the difference. However, if large attenuator values are used, the default Cal All settings will likely improve your results.

**S-Parameter Cal Port Power**

Set the power level at which the S-Parameter cal is performed.

**Power Offsets**

Power Offsets are channel-scoped. Consequently, offsets that you already set are NOT automatically copied to the Cal All session. This setting allows you to also apply a Power Offset during the Cal All session. Learn about Power Offsets.

**Noise Reduction**

This button accesses the following dialog for settings that help reduce trace noise and the noise floor which can lead to better dynamic range and more accurate measurements. Learn more.

![Noise Reduction](image)

**IF Bandwidth**

Set the IFBW used to perform the Cal All calibration. The default IFBW setting of 1 kHz is a good nominal setting for most measurements. Lowering the IFBW removes noise from the calibration measurement, but also causes slower sweeps.

**Always ON**

Check to enable averaging.

**Average Factor**

Specifies the number of measurements that are averaged. Range of 1 to 65536 (2^16).
Average Type

Sweep  Each data point is based on the average of the same data point measured over consecutive sweeps.

(Sweep) Restart  Begins a new set of measurements that are used for the average. Applies only to Sweep averaging - NOT Point.

Point  Each data point is measured the number of times specified by the Average Factor, and then averaged, before going to the next data point.

Reduce IF BW at Low Frequencies

When this box is checked, the VNA uses a smaller IF Bandwidth than the selected value. Learn more .

Mechanical Devices

This button accesses a dialog that shows the settings for all active channels. These settings are shown side-by-side for easy comparison. Learn more .

Select DUT Connectors and Cal Kits dialog box help

For each DUT port:

- Select the connector at the calibration reference plane (where the cal standards will be connected).
- Select the cal kit to be used.

Check Modify Cal to change the Thru method. An Unknown Thru cal is performed by default. Learn about THRU methods .

Learn more about this dialog .
**Power Cal Settings** dialog box help

A guided power cal is performed on the source ports for the Cal All calibration.

This dialog is displayed for each source port to receive a power cal.

To perform an LO power cal for a mixer channel, set the LO port to a VNA or external source in the Mixer Setup dialog. Then select that port in the Selected Channels dialog.

- To use the **same** power sensor for all power cals, do **NOT** check Use Multiple Sensors.
- To use **different** power sensors, check **Use Multiple Sensors**. The sensor must be configured as a PMAR device. Learn how.

Learn about this dialog box.

---

**Cal All Summary** dialog box help

This page is a summary of the Cal All settings. Confirm the settings, then click **Next >** or **< Back** to change settings.
**Cal Steps**  dialog box help

Follow the prompts to connect each standard. Then click **Measure**.

Click **Re-measure** if necessary.

Then click **Next >**

**Finish Cal**  dialog box help

Click **Finish** to save the Cal All session results to Cal Registers.

Or click **Save As User CalSet**, then enter a prefix title. The Meas Class and channel number are appended to this prefix to save to a User Cal Set for each calibrated channel.

Learn more about this dialog.
Using Calibration Sets

- What are Cal Sets
- Cal Registers and User Cal Sets
- How to Manage and Apply Cal Sets
  - Cal Set Selection dialog box help
  - Cal Set Properties dialog box help
  - Select Cal Set -- Choose Stimulus Settings dialog box help
- Examples of Cal Set Usage
- Archiving Cal Sets using .cal files

See Also

Save and Recall: Instrument States and Cal Set Data

See other Calibration Topics

What are Cal Sets

At the completion of a calibration, all calibration data is stored to a Cal Set. The Cal Set can be applied later to any channel that has the same stimulus settings as the Cal Set, thereby saving the time it takes to perform another calibration. The following data is saved to a Cal Set:

- Name
- Cal Set Description
- Cal Set Attributes - stimulus settings, cal type, port association
- Standards data - The “Standards data” container in the Cal Set is intended for internal use only. External access is provided for use in diagnosing calibration problems. Users should not form any expectations as to the presence of the data or the naming conventions used.
- Error term data
- GUID (Globally Unique IDentifier)
Cal Registers and User Cal Sets

There are two types of Cal Sets:

- **Cal Registers** (channel specific)
- **User Cal Sets**

Calibration data is automatically saved to a Cal Register at the end of every calibration. You can also choose to save the cal data to a User Cal Set.

Preset clears all PNA settings but the data inside a Cal Set is not volatile until, for example, another calibration is performed. Also, preset will disassociate all Cal Sets from any channel. But the process of calibration will reestablish that association.

**Calibration Registers**

Calibration Registers are designed to simplify calibrations for most users. When a calibration is complete, the data is automatically saved to the channel's Cal Register, overwriting (or appended to) the previous cal data stored in that register. This concept is similar to 'legacy' Vector Network Analyzers.

- Every channel has ONE dedicated Cal Register. They are named CHn_CALREG, where n is the channel number. The name cannot be changed.
- Data in Cal Registers are less permanent than User Cal Sets because they are the default container for a new calibration. However, preset never clears the contents of any Cal Set.
- Cal Registers can be applied to other measurements, but ONLY on the same channel as the Cal Register.

**User Cal Sets**

At the end of a calibration, you can choose to also save cal data to an existing or new User Cal Set.

- User Cal Sets can be applied to any number of channels simultaneously.
- User Cal Sets are named by you for easy identification.
- You can have an unlimited number of User Cal Sets.
- At any time, you can copy Cal Register data to create a User Cal Set. See Cal Set Properties.
- User Cal Sets are never cleared unless the user deletes them or uses them in a new calibration.

**Appending Data in a Cal Set**

---

530
At the end of a calibration, data is saved to the channel's Cal Register and, if you choose, to a User Cal Set. When you choose to save to an existing User Cal Set, the analyzer attempts to append the new error terms to the existing User Cal Set. The existing Cal Set data is completely overwritten UNLESS the new data can coexist with the existing data according to the following two rules:

- The stimulus settings between the two Cal Sets must be identical. Otherwise, the new cal will overwrite the old cal and one will not be able to append to the existing calibration. The comparison is strict, and includes the leveling mode and the source power state (including AUTO vs ON). In addition, frequencies, IFBW, source and receiver attenuators, source power value, power slope, IF Path, Shift LO mode, and any path settings are also compared. Therefore, it is highly recommended that immediately before starting the second calibration, recall the first calibration and choose to change the stimulus to match the Cal Set. Once you have a complete Cal Set, you can perform normal operations such as turning on receiver leveling, etc.

- The new cal must involve different ports from the existing cal.

**Note:** The reason for this behavior is that the S-parameter channel, unlike most of the applications, supports multiport. Therefore, it is possible that a user may need to accumulate separate calibrations for different subsets of ports

For example:

**Case 1** - An existing Cal Set contains a full 2-port cal between ports 1 and 2. Using the same stimulus settings, you perform a 1-port cal on port 3. At the end of the cal, you click *Save As User Cal Set* and select the existing full 2-port User Cal Set.

**Result:** The 1-port cal is appended to the 2-port User Cal Set. There is NO overlap between them.

**Case 2** - Same situation as Case 1, except the 1-port cal is performed on port 1.

**Result:** The Cal Set will contain a 1 port cal on port 1 and a 1 port cal on port 2. The overlapping tracking terms are removed rendering the original full 2 port cal invalid.

### How to Manage and Apply Cal Sets and Cal Types

The analyzer attempts to apply a Cal Set and turn error correction ON for ALL of the measurements on the active channel. This may not always be possible. For example, suppose a channel contains both S11 (reflection) and S21 (transmission) measurements. If a Cal Set that contains only an S11 Cal Type is applied to that channel, the Cal Set does not contain the error terms to correct the S21 measurement. Error correction is turned ON for the S11 measurement and NOT turned on for the S21 measurement.

There are two ways to apply an existing Cal Set (Cal Register or User Cal Set) to a measurement:

1. Recalling an Instrument State with Cal data (.cst file) - A .cst file contains an Instrument State with all measurement attributes AND a ‘pointer’ to the Cal Set that was used to calibrate the measurement. Before
saving a .cst file, be sure that a User Cal Set (NOT a Cal Register) is being used for the measurement. Because Cal Registers are automatically overwritten when a new calibration is performed, it is likely that the Cal Register data will change before the .cst file is recalled.

2. Create a new measurement and select a Cal Set to apply to the active channel.

**Note:** NEVER copy or modify Cal Sets from Windows Explorer or other applications. Cal Sets should only be accessed through the VNA Application.

**How to select and apply a Cal Set to the active channel**

**Using Hardkey/SoftTab/Softkey**

1. Press **Cal > Cal Sets & Cal Kits > Cal Set...**

**Cal Set Selection** dialog box help

This dialog allows you to manage and apply Cal Sets

Since the number of Cal Sets you can have is limited by the amount of analyzer memory, old Cal Sets (with 'stale' data) should be deleted or overwritten.

- Learn about Cal Registers.
- Learn how to View the Error Terms of a Cal Set.
To apply a Cal Set to the active channel, click a row to select that Cal Set, then click OK. The Cal Set used by the currently active channel is indicated with dotted outline. Currently selected Cal Set is indicated with blue highlight.

Note: A Cal Set must have been generated from the same measurement class as the active channel in order for it to be applied.

Columns: click a heading to sort by that column

**Cal Set Name** Name to identify the Cal Set.

**Class** Indicates the type of channel that created the calset.

**Active Channels** Channel numbers that are currently using this Cal Set. A blank entry means it is not currently in use.

**CalType / Ports** Type of Cal contained in the Cal Set. Learn about applying appropriate Cal Types.

**Cal Type Abbreviations:**

1P, 2P, 3P, 4P... Port list indicating which ports were calibrated.

+ - Indicates source and receiver Power Correction is included for the ports that are listed. So 2P+(2,3) means that the test and reference receivers on ports 2 and 3 are power calibrated.

R - Response (parameter).

ER/x-y Enhanced Response, where x is the receive port; y is the source port. ER/2-3, therefore, corrects S23.

ER+/x-y Enhanced response plus power. Also contains tracking terms for bx and ay.

**VMC** Vector Mixer Cal

**SMC** Scalar Mixer Cal

**Date** Date and time the Cal Set was last modified.

Buttons

**Select** Applies the selected Cal Set to the active channel. If the stimulus settings of the Cal Set and channel are different, a choice must be made. This button is greyed-out if the highlighted Cal Set is not compatible with the active channel.

**Unselect** Available ONLY if the selected Cal Set is being used by the active channel. Click 'Unselect', then click OK to exit with the Cal Set un-applied.
**Close**  Exit the dialog box. Performs no further action.

**Drop-Down Selector**

**Channel**  To choose the active channel.

**Edit**  This includes Properties, Copy, Delete and Delete All.

**Properties**  Starts the Cal Set Properties dialog box. This allows you to view all of the Cal Set properties.

**Copy**  Invokes the Copy Cal Set dialog box. Type a name for the copy of the selected Cal Set data.

**Delete**  Permanently deletes the Cal Set after you choose OK to a warning prompt.

**Delete All**  Permanently deletes ALL listed Cal Sets and Cal Registers after you choose OK to a warning prompt.

---

**Cal Set Properties** dialog box help

![Cal Set Properties dialog box](image)

Allows you to view all of the Cal Set properties.
**Drop-Down Selector**

**Edit**  This includes Cal Set Name and Description.

- **Cal Set Name**  Edit name of the User Cal Set. You CANNOT change the name of a Cal Register.

- **Description**  Descriptive text to further identify the Cal Set.

**Buttons**

- **OK**  Applies the changes on Cal Set name or description if any and then closes the dialog box.

- **Cancel**  Discard changes made to the Cal Set name or description.

Learn how to [View the Error Terms of a Cal Set](#).

**Note:** Only temperature compensated calibrations show the temperature in the Calset Properties dialog.

---

**Correction Properties** dialog box help

**Tabs**

**Summary**
Allow you to view summary of correction properties including Correction, Receiver Power, Source Power and Correction Level.

**Channel Selector** Select the channel that you want to view the summary of correction properties.

**Cal Set**
Allow you to view the Cal Set properties. Shows the same data as shown in the Cal Set Properties dialog accessed from the Cal Set Selection dialog.

**Channel Selector** Select the channel that you want to view the Cal Set properties.

**Stimulus Setting Different between Cal Set and Measurement**

**Select Cal Set -- Choose Stimulus Settings** dialog box help
The Cal Set contains the channel stimulus settings that were in place when the Cal Set was saved. This dialog appears when the Cal Set channel settings are different than those of the channel to which the Cal Set is being applied. Choose between the following options. (See above image).

- A. Keep the Active Channel Stimulus settings. Interpolate if possible.
  - If the Cal Set frequency range is greater the active channel, then Interpolation will be turned ON. Learn more about Interpolation Accuracy
  - If the Cal Set frequency range is less than the active channel, then this option is not available.

- B. Keep the Cal Set Stimulus settings. The Active Channel stimulus setting are changed.

OK  Make the change.

Cancel  Cal Set will NOT be applied.

Examples of Cal Set Usage

The following examples show how Cal Sets increase flexibility and speed in making analyzer measurements.

- Using one User Cal Set with many Channels
  - Using one Measurement with many Cal Sets

Using one User Cal Set with many Channels

It is possible to do one calibration, then apply it to several channels.

An example:

During a manufacturing process, you may have many calibrated channels. You may wish to continuously cycle through the measurements and examine them individually. Occasionally, you may wish to refresh the calibration without having to recreate all the measurement state files.

Here is how: Examine the stimulus settings for each channel. Then make the User Cal Set stimulus range a super-set of the whole group. Each channel can then use the same User Cal Set. Some calibrations will be interpolated. Note: Make sure that interpolation is turned on.
Notice in the following image, Cal Set 78 is used on more than one channel, in this case Channel 5 and 16.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Cal Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>78</td>
</tr>
<tr>
<td>15</td>
<td>98</td>
</tr>
<tr>
<td>16</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>100</td>
</tr>
</tbody>
</table>

Using one Measurement with many Cal Sets

The drawback with having one very large User Cal Set associated with many instrument states could be a loss of accuracy due to interpolation. In such cases, consider using one User Cal Set for each stimulus setting. The stimulus conditions can then be changed for a channel by applying different User Cal Sets. Other settings (window setups, measurement definitions, scaling, limits, markers) will not change. This may result in faster state changes than if you saved and recalled *.cst files for each set of stimulus conditions.

**Example #1:** An amplifier needs to be measured at several input power levels. Calibrate at several power levels and save each calibration in a separate User Cal Set. Then, apply the User Cal Sets to the single measurement consecutively.

**Example #2:** Making an S21 Measurement, you need to measure both wide span and narrow span characteristics of the device. One Cal Set covers the wide span setup; another the narrow span setup.

Archiving Cal Sets using .cal or .csa files

Because User Cal Sets can easily be deleted, provide extra backup by also saving your calibration as a .cal or .csa file (see saving a .cal file).
Example:

One person performs a calibration, names and saves it as a User Cal Set. This Cal Set is available for any other person to use. A second user could accidentally delete or modify the User Cal Set requiring the originator to repeat the calibration.

Security can be provided for calibration data by saving the Cal Set to a .cal file or .csa file. At a later time, the file could be recalled and the original calibration restored.
Error Correction and Interpolation

Error Correction and Interpolation settings work together to provide you with the highest level of calibration accuracy possible.

- How to set Error Correction
- Error Correction
- Viewing Correction Levels
- How to set Interpolation
- Interpolation Accuracy

See other Calibration Topics

How to set Error Correction

Using Hardkey/SoftTab/Softkey

1. Press Cal > Main > Correction > Channel Correction On|Channel Correction Off.

Error Correction

The Error Correction ON setting means that the calibration error terms are applied to the measurement. Error Correction is automatically turned ON when a calibration is performed or if a Cal Set is applied to a measurement. The VNA attempts to turn error correction ON for ALL of the measurements on the active channel. This may not always be possible when applying Cal Sets. For more information, see Applying Cal Sets.

When full 2-port error correction is ON, both forward and reverse sweeps are required to gather all 12 error terms, even if only one reflection measurement is displayed. This may result in a higher measurement speed than expected. Learn more.

You can always turn Error Correction OFF for the active measurement by clicking Correction OFF. The VNA will turn Error Correction OFF automatically when making stimulus changes under some conditions. To turn correction back ON, click Correction ON. Then:
If Interpolation can NOT be performed, a dialog box will ask if you would like to change the stimulus settings to those of the applied calibration. Click OK or Cancel.

If Interpolation can be performed, the stimulus setting will change and correction turned ON.

### How to set Factory Error Correction

**Using Hardkey/SoftTab/Softkey**

1. Press Cal > Main > Factory Cal On | Off

---

**Factory Calibration**

The factory calibration is a calibration which is done at the factory shipment or service centers for periodical calibration.

**Viewing Correction Level**

The correction level provides information about the accuracy of the active measurement. Correction level notation is displayed on the status bar for different calibration types like response, full 2-port, TRL, or power calibration.

#### To View Correction Levels:

Right-click in the display, select Customize Display, select Toolbars tab, then select Status Bar. The status bar appears and displays the following items:

<table>
<thead>
<tr>
<th>Correction Level</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>C N-Port</td>
<td>Highest</td>
</tr>
<tr>
<td>C Enh Resp</td>
<td></td>
</tr>
<tr>
<td>Enhanced Response</td>
<td></td>
</tr>
<tr>
<td>Response</td>
<td></td>
</tr>
<tr>
<td>Factory</td>
<td>Lowest</td>
</tr>
<tr>
<td>Factory calibration at test port</td>
<td></td>
</tr>
<tr>
<td>No Cor</td>
<td></td>
</tr>
<tr>
<td>No Correction</td>
<td></td>
</tr>
<tr>
<td>C*</td>
<td>Uncertain</td>
</tr>
<tr>
<td>Interpolated</td>
<td></td>
</tr>
<tr>
<td>CΔ</td>
<td>Uncertain</td>
</tr>
<tr>
<td>Changed</td>
<td></td>
</tr>
</tbody>
</table>
**C N-Port**

Full N Port correction, where N is the number of fully calibrated ports.

This correction is applied to SParameters.

If the calibration was performed with a receiver power cal, this correction can be applied to receiver measurements. (eg: a1, b1, A, R1, b1/a1). (See Correction Methods for the ability to control the level of calibration applied to receiver measurements).

**C Enh Resp**

Enhanced response cal is an aggregate of a 1 Port calibration and a transmission response cal.

This correction is applied to reflection and transmission parameters in either the forward direction (S11, S21) or the reverse direction (S12, S22) depending on how the calibration was performed.

For reflection measurements, Enhanced Response correction is equivalent to C 1-Port correction. For transmission measurements, the correction is equivalent to a match-corrected transmission response cal.

**C Resp**

Response calibrations are one term calibrations that correct for the frequency response of the measurement. It does not correct for impedance mismatches.

**No Corr  No Correction**

The following will cause the VNA to turn Error Correction OFF for the channel:

- Decrease the start frequency
- Increase the stop frequency
- Change start frequency, stop frequency, or number of points with Interpolation OFF.
- Change sweep type

The correction pop up pane, accessed by right-clicking on the Correction item in the status bar, indicates port by port correction methods for a VNA with 12 or less test ports. This table is updated when the port subset correction is turned on to reflect the correction methods being applied. In the image below, the pane indicates a full 4-port calibration. On the right, the table indicates the methods after the correction was devolved to ports 1, 2, and 3.
The $F^+$ indicates that the port had the full error correction applied. The $e^+$ indicates that the enhanced response correction method was applied to the port.

**Note:** For more information about port sub-setting, refer to Port Sub-Setting Examples.

**C* Interpolated Correction**

"C star" appears in the status bar when a measurement is being interpolated. See Interpolation (above) and Interpolation Accuracy.

**CΔ Changed Settings**

"C-delta" appears in the status bar when one or more of the following stimulus settings change. The resulting measurement accuracy depends on which parameter has changed and how much it has changed. For optimum accuracy, recalibrate using the new settings.

- Sweep time
- IF Bandwidth
- Port power
- Stepped sweep enabled/disabled
How to set Interpolation

Using Hardkey/SoftTab/Softkey

1. Press Cal > Main > Interpolation ON|OFF.

Interpolation

Calibration interpolation adjusts calibration error terms to match changes to the following settings that you make AFTER a calibration is performed or a Cal Set applied.

The Interpolation ON setting means that interpolation is **enabled** for the active measurement. This does not necessarily mean that the measurement is interpolated. When enabled (ON), if interpolation becomes necessary because you change any of the following stimulus settings, **then** interpolation will be applied. When stimulus settings change while interpolation is OFF, interpolation is NOT applied but instead, error correction is turned OFF.

Interpolation occurs (if enabled) when you change any of the following settings:

- Start frequency increased
- Stop frequency decreased
- Number of points

**Note:** Decreasing the start frequency, or increasing the stop frequency will always turn correction OFF. (Exception: Power Calibration DOES extrapolate to the start and stop frequencies.)

Interpolation Accuracy

When a measurement is interpolated, the accuracy of the measurements cannot be predicted. It may be affected significantly or not at all. Identifying measurement errors in these cases must be determined on a case-by-case basis. In general, the magnitude and phase stimulus from the VNA and the response from the DUT need to be smooth and continuous for measurement interpolation to give accurate results.

Significant measurement inaccuracy WILL occur when the phase shift response between measurement points increases changes more than 180 degrees. The VNA will incorrectly interpolate the new phase data. For more information, see phase accuracy.

In general, the chances of significant inaccuracy increases when interpolating measurements under the
following conditions:

- when frequency span between measurement points becomes much greater.
- when measurement frequencies are above 10 GHz where phase changes happen more rapidly.
- when interpolating across frequency band crossings. Learn more about band crossings.

**Note:** When the interpolation algorithm encounters an abrupt or large change in the response magnitude or phase, such as can occur at band crossings, large interpolation errors can be included in the displayed data. These errors can be seen as steps or spikes. If this occurs, consider turning off interpolation, changing the measurement parameters, or creating sweep segments that skip over the band crossings.
Using ECal

This topic discusses all aspects of performing an ECal:

- ECal Overview
- Connect ECal Module to the Analyzer
- How to Perform a Calibration Using ECal

See Also:

ECal User-Characterization
Perform a 4-Port Cal with ONE 2-Port ECal Module
Restore ECal Module Memory

See other Calibration Topics

ECal Overview

ECal is a complete solid-state calibration solution. Every ECal module contains electronic standards that are automatically switched into position during a measurement calibration. These electronic standards have been measured at the factory and the data stored within the memory of the ECal module. The analyzer uses this stored data, along with the measured data, to calculate the error terms for a measurement calibration.

ECal modules are available in 2-port and 4-port models and a variety of connector types, covering many frequency ranges. See Analyzer Accessories for more about available ECal modules and ordering information.

You can perform the following calibrations with ECal:

- 1-Port Reflection calibration
- Full 2-Port calibration
- Full 3-Port calibration
- And so forth...

Verify the validity of a mechanical or ECal calibration with ECal confidence check.
Care and Handling of ECal Modules

You can improve accuracy, repeatability, and avoid costly repair of equipment in the following ways.

- Practice proper connector care. See Connector Care.
- Protect equipment against ESD damage. Read Electrostatic Discharge Protection.

Power Level into an ECal module

- NEVER exceed the following Damage levels to the ECal module.
- For highest accuracy, do not exceed the following ECal Compression levels when calibrating:

<table>
<thead>
<tr>
<th>Model</th>
<th>Compression level</th>
<th>Damage level</th>
</tr>
</thead>
<tbody>
<tr>
<td>N469x series</td>
<td>-5 dBm</td>
<td>+10 dBm</td>
</tr>
<tr>
<td>N4432x series</td>
<td>-7 dBm</td>
<td>+20 dBm</td>
</tr>
<tr>
<td>N4433x series</td>
<td>+7 dBm</td>
<td>+20 dBm</td>
</tr>
<tr>
<td>N4431x series</td>
<td>+7 dBm</td>
<td>+20 dBm</td>
</tr>
<tr>
<td>N755xA series</td>
<td>-15 dBm</td>
<td>+10 dBm</td>
</tr>
<tr>
<td>8509x series</td>
<td>+9 dBm</td>
<td>+20 dBm</td>
</tr>
</tbody>
</table>

The power level can be increased after calibration with minimal impact on measurement accuracy.

Connect ECal Module to the Analyzer

ECal modules are controlled and powered through a USB connection. When you connect the module, the type of module, frequency range, and connector type are automatically recognized.

Important Note: DO NOT connect/disconnect USB devices during ECal calibrations. Doing so may cause problems with the calibration.

See Important First-time USB connection note.

ECal modules connect to the USB port on the front or rear panel of the VNA.
1. Wear a grounded wrist strap when making connections.

2. Connect the USB cable **Type B** connector to the ECal module and the USB cable **Type A** connector to the USB connector of the analyzer, as shown in the following graphics.

![ECal Module USB Port](image1) ![Analyzer Front Panel USB Port](image2)

**Notes:**

- Unused ECal modules that have completed a calibration may remain connected to the USB port.
- You can connect and disconnect the ECal module while the analyzer is operating. However, DO NOT connect or disconnect the module while data transfer is in progress. This can result in damage or at least corrupted data.

---

**How to Perform a Calibration Using ECal**

Select an ECal module that has connectors of the same type and gender as the DUT. If such an ECal module is not available, a module with connectors different from the DUT can be used by using Advanced Settings or User Characterization. See Also: Perform a 4-Port Cal with ONE 2-Port ECal Module

Connect the ECal module ports to the analyzer ports. During the calibration process the analyzer can either automatically detect how the ECal module is connected, or the orientation can be performed manually.

1. Connect the ECal module USB cable to the analyzer USB. See Connect ECal Module to USB.

2. Allow the module to warm up until it indicates READY.
3. Enter the analyzer settings. See Set Up Measurements.

4. Do one of the following to start the Calibration Wizard

Using **Hardkey/SoftTab/Softkey**

1. Press **Cal > Main > Other Cals > Ecal...**

2. In the Guided Calibration Wizard dialog box (step 2), select ECal option from the Cal Kits combo box.

---

**Select Calibration Ports and ECal Module** dialog box help

Allows you to select calibration type and settings.

**Cal Type Selection / Configuration** Select the number of ports to calibrate. Then select the port number configuration.

- **4 Port ECal**
- **3 Port ECal**
- **2 Port ECal**
- **1 Port ECal- (Reflection)** Advanced Settings are not available.

**View/Select ECal Module** Click to Select the ECal module if more than one ECal module is connected to the USB. Also, Select the User Characterization within the module. Learn more about User Characterization.

**Show Advanced Settings** Check to display the Advanced Settings when Next is clicked.

**Calibrate source and receiver power** Check to perform a Guided Power Calibration. Learn more.
**Note:** ECal isolation is not performed. The inherent isolation of the analyzer is better than that attained with correction using an ECal module.

**Note:** Terminate any unused ECal ports with a 50 ohm load. Refer to [Determining Effects of Not Terminating Unused ECal Ports](#).

**Note:** Do not connect any USB memory during ECal calibration.

---

**ECal module not found** dialog box help

When this dialog appears, the ECal module is not connected or has not been recognized by the network analyzer.

**Retry**  Check the USB connections and click to continue.

**Notes:**

- If your ECal module is not detected, try to unplug, then reconnect to the USB.

- When the ECal module is connected to the network analyzer for the first time, it may take approximately 30 seconds for the analyzer to recognize the module and make it available for calibration.

- For best accuracy, allow the ECal module to warm-up until it indicates READY.

- Keysight 8509x and N4431 ECal modules, when first connected, draw significantly more current than other modules. This could cause the USB to stop working in certain situations. [See USB limitations](#).

- See [Connect ECal Module to USB](#).

---

**Select Module and Characterization** dialog box help
Note: User Characterizations listed in the dialog below that have no temperature shown cannot be temperature-compensated during calibrations. Also, this is true of CalPod as ECal characterizations that were performed prior to this temperature capability in the VNA firmware, because temperature was not measured-and-recorded. However, the firmware will still recognize those and allow them to be used for cals.

**ECal Module** Select one of the ECal modules that are connected to the analyzer.
Detect Connected ECals  Click to rescan the USB for ECal modules.

Available Characterizations

**ECal Module Memory** - Displays the factory and [user characterizations](#) that are stored in the ECal module.

**Instrument Disk Memory** - Displays the user characterizations that are stored in Disk Memory. [Learn more User Characterizations in Disk Memory](#).

**Temperature** - Displays the temperature reading at the time a characterization was performed.

The information in the Calset Properties dialog confirms which of those characterizations were temperature-compensated during calibration. The *(compensated)* notation for a CalPod indicates that the CalPod’s characterization was temperature-compensated when that cal’s error terms were computed (it also implies the factory thermal data for that CalPod must be present on the VNA). Whereas the *(uncompensated)* notation indicates that temperature-compensation could not be done for that characterization during the cal, which could mean either that one is an older characterization that didn’t record its temperature, or else factory thermal data for that CalPod serial # was not installed on the VNA.

Select the characterization data to use for the calibration. Once selected, that characterization becomes the default selection until the analyzer is turned OFF and restarted. When restarted, **Factory** again becomes the default selection.

---

### Error: Frequency Range dialog box help

When this dialog appears, the current cal standards (or ECAL module) does not cover the current frequency range of the measurement. Do one of the following to correct the problem:

**Cal Kit Class Category**  Not available with ECal modules.

**Frequency**  Change the frequency range of the active channel.

**Edit**  Not available with ECal modules.
Back  Select a different characterization that covers the required frequency range.

Cancel  Re-characterize the module with an increased frequency range.

**Select DUT Connectors and Cal Kits** dialog box help

If the ECal module or selected User Characterization has more than one connector type, then the following dialog box is presented which allows you to describe the DUT connector type. Otherwise, click next to proceed to **Advanced Settings** (if checked) or **ECal Steps**.

**Connectors**

The available connectors are listed for each DUT port.

**Advanced Settings** dialog box help

**Thru #n**

Lists the proposed Thru connections to be made during the calibration process. You can change these Thru connections to better suit your test setup.

- The proposed Thru connections are listed automatically.
- Additional Thru connections can be selected for higher accuracy. Learn more.
For Balanced measurements, learn which Thru paths to select.

Add Thru

Click to add a Thru connection. Learn more

Remove Thru

Select a Thru by clicking the "Thru #N" field or the "1st Port / 2nd Port" field. Then click "Remove Thru". This selection is NOT available if the selected Thru is required for the calibration.

1st Port / 2nd Port

Click to change the two ports to be included in the Thru connection. The order of the port numbers (1st or 2nd) is not critical.

Thru Cal Method

Lists the available Thru Cal methods for the specified port pairs.

Learn about ECal Thru Methods

Cal Type/ Stds

Click to invoke the View / Modify Properties of Cal dialog box

Do orientation

When this box is checked (the default setting) the VNA automatically senses the model and direction in which an ECal module port is connected to the VNA ports. If power to the ECal module is too low, it will appear as if there is no ECal module connected. If you use low power and are having this problem, clear this check box to provide the orientation manually.

Orientation occurs first at the middle of the frequency range that you are calibrating. If a signal is not detected, it tries again at the lowest frequency in the range.

Choose delta match

Available only when a Delta Match Cal is required.

- Check, then click Next to invoke the Select Cal Set for Delta Match dialog box.
- Clear - The Cal Wizard uses the Global Delta Match Cal if available.
Specify how the ECal module is connected dialog box help

This dialog box appears when the Do orientation checkbox in the previous dialog box is cleared. Click the ECal Port that is connected to each VNA port.

Electronic Calibration Steps dialog box help

Note: Beginning in VNA Rev. 6.0, ECal can be performed with External triggers. Learn more.

Displays the instructions for each measurement required for calibration.

Measure  Measures the ECal standards.

Done   Click when last standard has been measured.

Saving an ECal Calibration

When complete, you can save the new calibration. Learn how.
## ECal User Characterization

- **Overview**
- **How to Perform a User Characterization**
- **Manage Disk Memory**
- **Restore ECal Module Memory**

### See Also

Using ECal
- Perform a 4-Port Cal with a 2-Port ECal Module

### Other Calibration Topics

#### Overview

A user-characterized ECal module allows you to add adapters to the ECal module, re-measure the standards in the ECal module, INCLUDING the adapters, then add that data to ECal memory or save it to disk memory. This extends the reference plane from the module test ports to the adapters.

Compared to legacy ECal modules, the new N755xA ECal modules have greater flash memory.

**Important Note:** DO NOT connect/disconnect USB devices during ECal calibrations. Doing so may cause problems with the calibration.

#### Why perform a User Characterization?

- If you need to use adapters with your ECal module, you could characterize your ECal module with the adapters attached and perform subsequent ECals in a single step.

- If you have a 4-port ECal module, you could configure the module with adapters of different connector types, then perform a User Characterization of the module. When you need to test a DUT with a pair of the connector types on your module, calibrate the analyzer with a 1-step ECal using the same two connectors on the User-characterized module.
If you test devices in a fixture, you could embed the characterization of the fixture in the characterization of the module. To do this, during the mechanical calibration portion of the User Characterization, calibrate at the reference plane of the device as you would normally calibrate. Then remove the fixturing to be embedded and insert the ECal module to be characterized. When measuring the ECal module, the analyzer removes the effects of the fixturing and stores the measurement results in the user characterized ECal module. Subsequent calibrations with that user-characterized module will also remove the fixture effects.

Notes:

- Both 2-port and 4-port ECal modules support User Characterization.

- User Characterization does not delete the factory characterization data. The factory data is saved in the ECal module in addition to the User Characterization data.

- The ECal Data Wipe Utility is the only way that data can be deleted from the module. Learn more at http://na.support.keysight.com/pna/apps/applications.htm.

- A User Characterization can be performed beyond the frequency range of the ECal module. Although this practice is allowed, calibration accuracy with the extended User Characterization is likely to be degraded. To determine the level of degradation, compare measurements of a variety of devices with a mechanical cal kit calibration versus an ECal extended User Characterization calibration.

- You can save up to 12 User Characterizations in a single ECal module. Previous releases allowed up to 5. There are memory limitations. The analyzer will determine if the contents of a User Characterization will fit inside the module before it is performed.

- A User Characterization can be performed remotely. See programming commands.

User Characterizations can be saved to **Disk Memory**. Learn how.

This feature provides the following benefits:

- A User Characterization using connectors that are NOT included in the supported connector table can NOT be stored to the ECal module. But when stored to disk memory, ANY connector type is allowed by firmware using a description of any length for the User Characterization.

- There is NO limit to the number of data points allowed in a User Characterization stored to disk memory. When stored in the ECal module, the number of data points is limited to a maximum of 65535 per characterization, or less as dictated by the remaining free memory in the module.

- The number of User Characterizations that can be stored to disk memory is limited only by available disk space.

- User Characterizations stored to disk memory can be freely shared between analyzers.

Learn how to Manage User Characterization in Disk Memory.
### How to Perform a User Characterization

**SUMMARY** (A detailed procedure follows.)

1. Select adapters for the module to match the connector configuration of the DUT.

2. Either calibrate the analyzer using mechanical standards or recall an existing Cal Set.

3. Measure the ECal module, including adapters, as though it were a DUT.

4. The measurement results are the characterization data that then gets stored inside the module or to disk.
Note

A 2-port analyzer can be used to perform a User Characterization on a 4-port ECal module. However, a 4-port ECal module has SIX different port pairs. The analyzer must be recalibrated for each port pair that uses unique connector types or gender.

- If all 4 ECal module ports have the same connector type and gender, then only one calibration is required to measure all six port pairs.
- If all 4 ECal module ports have different connector types or gender, then 6 calibrations are required.

When more than one calibration is required during a User Characterization, then ALL calibrations must be performed using the standard Cal Wizard, saved to Cal Sets, and then recalled from Cal Sets DURING the User Characterization.

Detailed steps to Perform a User Characterization

1. Connect the ECal module to the network analyzer with the USB cable. See Connect ECal Module USB to the analyzer USB.
2. Allow the module to warm up until it indicates READY.
3. Preset the analyzer.
4. Set up the measurement. For best accuracy, the IF bandwidth should be set to 1 kHz or less.
5. Start and complete the Characterize ECal Module Wizard:

Using Hardkey/SoftTab/Softkey

1. Press Cal > Cal Sets & Cal Kits > ECal > Characterize ECal....

Programming Commands
Select Module and Location dialog box help

**ECal Module**  Select one of the ECal modules that are connected to the analyzer.

**Detect Connected ECals**  Click to rescan the USB for ECal modules.

**Location**

- **ECal Module Memory**  Click Next to see the following dialog.

- **Disk Memory**  Enter a Characterization Name. This name appears when selecting a User Characterization to be used with subsequent calibrations.
  
  - Learn how to manage characterizations that are stored to disk memory.
  
  - See the benefits of storing the User Characterization to disk Memory.

**Keyboard**  Launches a keypad that can be used to type a characterization name from the analyzer front panel.

**Next**  Click to continue to the Select Connectors for the Characterization dialog box.

See note regarding extended frequency use.
Select User Number for new characterization dialog box help

Scroll to view all of the parameters of the stored characterizations. Select an empty location or select to overwrite an existing characterization.

Next  Click to continue to the Select Connectors for the Characterization dialog box.

See note regarding extended frequency use.

Select Connectors for the Characterization dialog box help

Connector Notes

When performing an ECal User Characterization, do NOT use a custom connector name that you added to this list. If you need to use a custom-defined connector type, select "Type B", or one of the "Type A" variations from the list of connectors for each port.

A User Characterization using connectors that are NOT included in the supported connector table can NOT be stored to the ECal module. But when stored to disk memory, ANY connector type is allowed. Learn more about storing to Disk Memory.

Select the adapters for the ECal module test ports. Select No adapter if no adapter is used on a port.

PORT A  Lists the connector types available for Port A.

PORT B  Lists the connector types available for Port B.
PORT C  Lists the connector types available for Port C (available with a 4-port ECal module).

PORT D  Lists the connector types available for Port D (available with a 4-port ECal module).

Next  Click to continue to the **Calibrations to perform or recall** dialog box.

---

**Calibrations to perform or recall** dialog box help

The analyzer must be calibrated before measuring the ECal module and necessary adapters. This dialog box displays the number and types of mechanical calibrations required for the characterization.

**Guide me through this cal now**  Click to perform a Guided calibration. A calibration kit is required for each connector type.

**Note:** Some PNA-L models cannot perform TRL calibration during the calibration portion of a User Characterization. However, this type of Cal can be performed using the Cal Wizard, saved to a Cal Set, then recalled at this point in the User Characterization.

If more than one calibration is required, the following selection is not available. See Note.

**Let me recall this cal from a cal set**  Click to select an existing Cal Set. You cannot select a Cal Set that is currently in use. Learn more about **Using Cal Sets**.

Next  Click to continue to either the **Select Cal Kits** or the **Select Cal Set** dialog box.
**Select Cal Kits** dialog box help

Provides a list of calibration kits to perform the calibration. Select the Cal Kit you will use for each port.

**Enable Unknown Thru for characterizing the module**  Check to enable. This reduces the number of steps required to characterize the THRU standard. This setting is available only on VNA models with one reference receiver per test port.

**Next**  Click to continue to the **Select Cal Set** dialog box.

---

**Select Cal Set** dialog box help

The calibration that you perform will be written to a Cal Set. This dialog box allows you to select a Cal Set to overwrite, or to write to a new Cal Set. The current choice is visible below the **Select Cal Set** button.

**Select Cal Set**  Click to open the **Select A Cal Set** dialog box.

**Create new Cal Set**  Check to create a new Cal Set to store the calibration. Clear to select and overwrite a stored Cal Set.

**Next**  Click to continue to the **Guided Calibration Steps** dialog box.

**Note:** Remember the Cal Set name for future reference.
Guided Calibration Steps dialog box help

Instructs you to connect each calibration standard to the measurement port.

**Measure**  Click to measure the standard.

**Back**  Click to repeat one or more calibration steps.

**Done**  Click **after** a standard is re-measured and all measurements for the calibration are complete.

**Next**  Click to continue to the next calibration step. (Does **not** measure the standard.)

**Cancel**  Exits Calibration Wizard.

The **Specify nominal delay** or **Guided Calibration completed** dialog box appears when the steps are completed.

Specify nominal delay dialog box help

This dialog ONLY appears when **Adapter Removal** or **Unknown Thru** calibrations are performed.

The following values were estimated from the measurement. Most of the time, they are adequate. However, for CW sweep or frequency sweep with large step sizes, the accuracy of the values may be improved.

**Nominal adapter delay**  To improve this value, measure and record the delay of the adapter with a dense step size. Enter that value here.
**Nominal phase offset** (Waveguide ONLY). To improve this value, measure and record the phase offset of the Waveguide adapter with dense step size. Enter that value here.

When one connector is coax and the other connector is waveguide, the phase offset has an ambiguity of 180 degrees. For consistency, the estimate provided here is always between 0 and 180 degrees. You can change this estimate to any value between -180 degrees and +180 degrees.

---

**Guided Calibration completed** dialog box help

Allows you to finish the calibration and continue to the next characterization steps.

**No. Finish now** Select to save Cal Set data.

**Yes** Allows selection of Save options.

**Next** Click to continue to the [Exit to Inspect Quality of Calibration](#) dialog box.

---

**Exit to Inspect Quality of Calibration** dialog box help

Allows you to exit User Characterization to validate the calibration before proceeding with the characterization.

**Back** Allows you to repeat calibration.

**Next** Click to continue to the [Characterization Steps](#) dialog box.
**Cancel** Exits the Calibration.

To return to the current step:

2. In the **Select user number for new characterization** dialog box, click Next.
3. In the **Select Connectors for Characterization** dialog box, click Next. (Previous entry is stored in memory.)
4. In the **Calibrations to perform or recall** dialog box, recall the Cal Set that you just performed.

### Characterization Steps dialog box help

Describes the instructions for each measurement required for characterization.

**Measure** Measures the ECal module.

**Next** Click to continue to the **Information for the New Characterization** dialog box when measurements are complete.

### Information for the New Characterization dialog box help

Allows you to describe the properties of the User Characterization.
Suggestions for connector abbreviations

To minimize the number of characters, we suggest using the following 3-character codes to describe the connectors listed.

A User Characterization using connectors that are NOT included on this list can NOT be stored to the ECAL module. But when stored to disk memory, ANY connector type is allowed. Learn more about storing to Disk Memory.

<table>
<thead>
<tr>
<th>Connector Type</th>
<th>3-Character Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 mm female</td>
<td>10F</td>
</tr>
<tr>
<td>1.0 mm male</td>
<td>10M</td>
</tr>
<tr>
<td>1.85 mm female</td>
<td>18F</td>
</tr>
<tr>
<td>1.85 mm male</td>
<td>18M</td>
</tr>
<tr>
<td>2.4 mm female</td>
<td>24F</td>
</tr>
<tr>
<td>2.4 mm male</td>
<td>24M</td>
</tr>
<tr>
<td>2.92 mm female</td>
<td>29F</td>
</tr>
<tr>
<td>2.92 mm male</td>
<td>29M</td>
</tr>
<tr>
<td>3.5 mm female</td>
<td>35F</td>
</tr>
<tr>
<td>3.5 mm male</td>
<td>35M</td>
</tr>
<tr>
<td>7-16 female</td>
<td>16F</td>
</tr>
<tr>
<td>7-16 male</td>
<td>16M</td>
</tr>
<tr>
<td>Type F female</td>
<td>F7F</td>
</tr>
<tr>
<td>Type F male</td>
<td>F7M</td>
</tr>
<tr>
<td>N50 female</td>
<td>N5F</td>
</tr>
<tr>
<td>N50 male</td>
<td>N5M</td>
</tr>
<tr>
<td>N75 female</td>
<td>N7F</td>
</tr>
<tr>
<td>N75 male</td>
<td>N7M</td>
</tr>
<tr>
<td>APC 7</td>
<td>7MM</td>
</tr>
<tr>
<td>K-band waveguide</td>
<td>KBW</td>
</tr>
<tr>
<td>P-band waveguide</td>
<td>PBW</td>
</tr>
<tr>
<td>Q-band waveguide</td>
<td>QBW</td>
</tr>
<tr>
<td>R-band waveguide</td>
<td>RBW</td>
</tr>
<tr>
<td>U-band waveguide</td>
<td>UBW</td>
</tr>
<tr>
<td>V-band waveguide</td>
<td>VBW</td>
</tr>
<tr>
<td>W-band waveguide</td>
<td>WBW</td>
</tr>
<tr>
<td>X-band waveguide</td>
<td>XBW</td>
</tr>
</tbody>
</table>
Next  Click to continue to the Write Characterized Data to the ECal module dialog box.

**Write Characterized Data** dialog box help

User Characterization and factory characterization data is written to either the disk memory or the ECal module memory.

**Write**  Click to write data.

The **Summary of new User Characterization** dialog box opens after data is saved to module.

- Existing data will be overwritten is you selected a User Characterization number that already has data. Learn more
- For more information, see **Restore ECal module memory**.
- The ECal Data Wipe Utility is the only way that data can be deleted from the module. Learn more at [http://na.support.keysight.com/pna/apps/applications.htm](http://na.support.keysight.com/pna/apps/applications.htm).

**Summary of new User Characterization** dialog box help

Verify the status of the ECal User Characterization.
- ECal module model number
- summary from User Characterization

**Cancel**  Click to exit (characterization complete).

**Finish**  Click to exit (characterization complete).

### Manage ECal User Characterizations in Disk Memory

Normally, User Characterizations that are stored in disk memory can be used indefinitely without needing them to be managed. However, this dialog allows you to backup the characterizations in case they are accidentally erased, or to save them to a file that can be moved to another analyzer.

### How to Manage ECal User Characterizations in Disk Memory

**Using** Hardkey/SoftTab/Softkey

1. Press **Cal > Cal Sets & Cal Kits > ECal > Manage ECal Disk Memory...**
This dialog allows you to do either of the following:

- Save an existing User Characterization in disk memory to an *.euc file.
- Load a previously saved *.euc file for use on the analyzer with the specified ECal module.

Learn more about User Characterizations stored to Disk Memory.

**ECal Module** Select an ECal Module from the list for which User Characterizations are currently stored in disk memory.

**Save As** Saves a User Characterization that is currently in disk memory to a *.euc file. This file can be used as a backup in case the archive file is accidentally deleted, or allows you to move the file to another analyzer to be used with the selected ECal Module.

**Import** Loads a previously saved *.euc file for use on the analyzer with the specified ECal module.

**Delete** Removes a User Characterization from disk memory.

**Note:** If a temperature is shown in the Temperature column, then the temperature during characterization was measured and recorded. For VNA ports on which a CalPodAsECal user characterization is used during a calibration, the cal error terms for those ports in the CalSet will have been computed with compensation for temperature at the time of calibration if the following two conditions are true: 1) The user characterization must show a temperature value in the Temperature column in this dialog box, and 2) the CalPod must be a Thermal or TVAC CalPod whose factory temperature data has been installed on the VNA by the installer package that was provided with the CalPod.

**Restore ECal Module Memory**

When user-characterized data is written to the ECal module, the entire contents of ECal memory is also written to the disk memory, including the factory ECal data. In the unlikely event that your ECal
module memory is lost, you can restore all ECal data to ECal memory.

Caution: If a new factory cal was performed after the ECal memory was written to disk memory, the new factory cal data will also be overwritten.

Note: An ECal Data Wipe Utility destroys all user data per US DoD 5220.22-M. Learn more at http://na.support.keysight.com/pna/apps/applications.htm

How to Restore ECal Module Memory
Using Hardkey/SoftTab/Softkey

1. Press Cal > Cal Sets & Cal Kits > ECal > Restore ECal Memory....

Module to be restored dialog box help

Verify the serial number of the module to be restored. If two modules are connected, choose the one to have data restored.

Next  Click to write data to the module.
Perform a 4-Port Cal with One 2-Port ECAl Module

You can perform a 4-port calibration with a 2-Port ECAl Module. When all four DUT connectors are the same type and gender, the calibration can occur with only four connections, the same number of connections you would make with a 4-port ECAl module.

- The ECAl module must span the frequency range of the measurement.
- The ECAl module must have connectors that match the DUT connectors. Because we are using a 2-port ECAl module, this means that the DUT must have only TWO unique connector types and gender. When the DUT has more than two connector types/genders, you can select a different cal kit for each port using SmartCal.

**Important Note:** DO NOT connect/disconnect USB devices during ECAl calibrations. Doing so may cause problems with the calibration.

**Procedure**

1. Connect the 2-port ECAl module to a VNA USB port.
2. Press **Cal > Main > Other Cals > Smart Cal...**
3. Select **4 Port Cal**, then click **Next** to see the following dialog:

4. Select the DUT Connectors for each port. In this example, all four DUT connectors are Type N, female.
5. Select the attached ECAl module. We are using a **85092-60007 ECAl** module.
6. Select **Modify Cal (Show Advanced Settings for ECAl)** then click **Next** to see the following dialog:

7. For the fewest number of physical connections, select the default port assignments.
● The 1st Port selection for each port pair is 1.

● For single-ended (standard) measurements, THREE is the minimum number of Thru connections. For Balanced measurements, FOUR Thru connections should be made. Learn more.

● For higher accuracy, select Add Thru. The Cal Wizard will add another port pair which results in more physical connections.

8. Select **ECal Thru as Unknown**. This is the most accurate and easiest Thru Cal Method. Learn more.

9. You may need to clear **Do Orientation** when calibrating at low power levels. Learn more. This will add additional connection steps.

10. Follow the prompts to complete the calibration:

   1. Connect ECal to ports 1 and 2. Click **Measure**.

   2. Connect ECal to ports 1 and 3. Click **Measure**.

   3. Connect ECal to ports 1 and 4. Click **Measure**.

11. At the **Specify Delay** dialogs, click **OK**. This is the measured delay for each of the Thru connections in the ECal module. Learn more.

12. Click either **Save As User Cal Set**, or **Finish**.
TRL (Thru, Reflect, Line) represents a family of calibration techniques that measure two transmission standards and one reflection standard to determine the 2-port 12-term error coefficients. For example, TRM (Thru, Reflect, Match), LRL (Line, Reflect, Line), LRM (Line, Reflect, Match) are all included in this family.

The traditional SOLT calibration measures one transmission standard (T) and three reflection standards (SOL) to determine the same error coefficients.

- Why Perform a TRL Cal?
- The TRL Calibration Process
- TRL Cal Kits
- Cal Standards Used in TRL
- TRL on 4-port PNA-L and ALL Models with an External Test Set

Why Perform a TRL Cal?

TRL calibration is extremely accurate, in most cases more accurate than an SOLT cal. However, very few calibration kits contain TRL standards. TRL Cal is most often performed when you require a high level of accuracy and do not have calibration standards in the same connector type as your DUT. This is usually the case when using test fixtures, or making on-wafer measurements with probes. Therefore, in some cases you must construct and characterize standards in the same media type as your DUT configuration. It is easier to manufacture and characterize three TRL standards than the four SOLT standards.

Another advantage of TRL calibration is that the TRL standards need not be defined as completely and accurately as the SOLT standards. While SOLT standards are completely characterized and stored as the standard definition, TRL standards are modeled, and not completely characterized. However, TRL cal accuracy is directly proportional to the quality and repeatability of the TRL standards. Physical discontinuities, such as bends in the transmission lines and beads in coaxial structures, will degrade the TRL calibration. The connectors must be clean and allow repeatable connections.

To learn more about Cal Standard requirements, see Cal Standards Used in TRL.
Note: Virtual Device describes a non-physical (connect the two test port reference planes together) type of connection description during the calibration. So, in a cal kit definition, you should not define more than one Thru standard with the same connector/gender pairing to each Virtual Device. This could cause those Thru standards to all be treated as the same physical connection step during a calibration, which would especially be a problem for TRL calibrations if a Thru standard and Line standard were measured as the same connection step.

The TRL Cal Process

Although TRL can be performed using the Cal Wizard Unguided Cal selection, the following process uses the easier SmartCal selection. Both selections require that you already have TRL calibration standards defined and included in a VNA cal kit.

1. Preset the VNA
2. Set up a measurement and the desired stimulus settings.
3. Press Cal > Main > Other Cals > Smart Cal....
4. Select the DUT connectors and Cal Kit for each port. The LOWEST port number of each port pair MUST include TRL standards. TRL appears as the Cal Method.
5. Check Modify Cal, Next, then View/Modify to change default TRL options if necessary.
6. Follow the prompts to complete the calibration.
7. Check the accuracy of the calibration

TRL Cal Kits

Keysight Technologies offers two cal kits that include the required standards to perform a TRL calibration: 85050C (APC 7mm) and 85052C (3.5mm). Both kits include the traditional Short, Open, and Load standards. (The Thru standard, not actually supplied, assumes a zero-length Thru). In addition, the kits include an airline which is used as the LINE standard. To use the airline, the kits include an airline body, center conductor, and insertion / extraction tools. The APC 7 kit includes an adapter to connect the airline to the APC connector.

Cal Standards Used in TRL

These standards must be defined in your TRL cal kit:

THRU
Note: All **THRU calibration methods** are supported in a TRL Cal **EXCEPT** Unknown Thru.

- The THRU standard can be either a zero-length or non-zero length. However, a zero-length THRU is more accurate because it has zero loss and no reflections, by definition.

- The THRU standard cannot be the same electrical length as the LINE standard.

- If the insertion phase and electrical length are well-defined, the THRU standard may be used to set the reference plane.

- Characteristic impedance of the THRU and LINE standards defines the reference impedance of the calibration.

- If a THRU standard with the correct connectors is NOT available, an adapter removal cal can be performed.

**REFLECT**

- The REFLECT standard can be anything with a high reflection, as long as it is the same when connected to both VNA ports.

- The actual magnitude of the reflection need not be known.

- The phase of the reflection standard must be known within 1/4 wavelength.

- If the magnitude and phase of the reflection standard are well-defined, the standard may be used to set the reference plane.

**LINE**

The LINE and THRU standards establish the reference impedance for the measurement after the calibration is completed. TRL calibration is limited by the following restrictions of the LINE standard:

- Must be of the same impedance and propagation constant as the THRU standard.

- The electrical length need only be specified within 1/4 wavelength.

- Cannot be the same length as the THRU standard.

- A TRL cal with broad frequency coverage requires multiple LINE standards. For example, a span from 2 GHz to 26 GHz requires two line standards.

- Must be an appropriate electrical length for the frequency range: at each frequency, the phase difference between the THRU and the LINE should be greater than 20 degrees and less than 160 degrees. This means in practice that a single LINE standard is only usable over an 8:1 frequency range (Frequency Span / Start Frequency). Therefore, for broad frequency coverage, multiple lines are required.

- At low frequencies, the LINE standard can become too long for practical use. The optimal length of the LINE
standard is 1/4 wavelength at the geometric mean of the frequency span (square root of f1 x f2).

**Note:** The TRL LINE standard must have a delay that is greater than 0 (zero) ps. Otherwise, calibration correction calculations will contain unpredictable results.

**MATCH**

If the LINE standard of appropriate length or loss cannot be fabricated, a MATCH standard may be used instead of the LINE.

- The MATCH standard is a low-reflection termination connected to both Port 1 and Port 2.
- The MATCH standard may be defined as an infinite length transmission line OR as a 1-port low reflect termination, such as a load.
- When defined as an infinite length transmission line, both test ports must be terminated by a MATCH standard at the same time. When defined as a 1-port load standard, the loads are measured separately. The loads are assumed to have the same characteristics.
- The impedance of the MATCH standard becomes the reference impedance for the measurement. For best results, use the same load on both ports. The load may be defined using the data-based definition, the arbitrary impedance definition, or the fixed load definition.

**See Also**

- See Modify Calibration Kits for detailed information about creating and modifying Calibration kit definitions.
- For more information, read Specifying Calibration Standards and Kits for Keysight Vector Network Analyzers (Application Note 1287-11)

**TRL on a 4-port PNA-L and ALL VNA Models with an External Test Set**

Beginning with the VNA code revision 5.25, TRL CAN be performed on a 4-port PNA-L and ALL VNA Models with an External Test Set enabled. Previously, a TRL calibration required a VNA with a reference receiver for each test port. With the new TRL method, a Delta Match Calibration is first performed and applied.

**Note:** See Delta Match Calibration to learn which models require this.

The accuracy of this TRL cal greatly depends on the accuracy of the Delta Match Calibration. With an accurate Delta Match Calibration, the difference in accuracy between a traditional TRL cal and this TRL cal is negligible.

**How to Perform a TRL Cal in these cases**
1. Press **Cal > Main > Other Cals > Smart Cal...**

2. Select a TRL cal kit for the ports to be calibrated.

3. During the calibration, the Cal Wizard prompts you for a valid Delta Match Cal.
CalPod is a system that simplifies the process of recalibrating the VNA without requiring the removal of the DUT or the physical connection of standards. This allows recalibration from a remote location such as when the DUT is in a temperature chamber.

**Note:** This feature is available to GCA, GCX, IMS, IMD, NF, NFX, SMC + Phase, and standard (S-Parameter) channels.

**Note:** Before using a CalPod module, ensure that Factory Cal is set to OFF (Cal > Main > Factory Cal OFF).

In this topic:

- Sweep Averaging Versus CalPod Averaging
- Process Overview
- How to start the CalPod dialog
- CalPod dialog
- CalPod Setup dialog
- CalPod Operational Check

**See Also**

CalPod as ECal

**Other Calibration topics**

**Sweep Averaging Versus CalPod Averaging**

Averaging helps to find the mean of an incoherent variation, but the source and “coherence time” of the variation are important.

There are three sources of variation in a CalPod corrected measurement:

1. Noise due to receiver noise floor. This tends to be fast variation and IFBW reduction (IFBW reduction is really just more averaging of the ADC readings, and point averaging, since it occurs in a similar time scale, is very similar to IFBW reduction) will reduce this to a mean value.
2. Variation due to instability in the cables, connectors, etc. This can be due to things like vibrational stability due to fans, air currents moving past, ringing from cars, trucks, and people moving about. These tend to be longer in time scale (a few seconds) and so sweep-to-sweep averaging will remove this long-time-scale variation.

3. Variation due to slight imperfections in the setting of the CalPod states, due to changes in line-voltage, regulation, and timing. This is helpful in the reduction of these imperfections (plus it also has a longer time scale than sweep-to-sweep averaging) as it accounts for very minor random variations in the setting of the states of the CalPod. By averaging several states (each state containing an average of the sweeps, each sweep contains an average of the IFBW acquisitions, each IFBW acquisition contains an average of the ADC readings, each ADC reading being independent), another source of variation is removed. Generally sweep averaging is not needed with CalPod averaging. But sweep averaging during calibration can help remove stability variation during calibration; so turn on sweep-averaging during calibration; but turn off sweep averaging during Initialization and Recorrection, and use CalPod averaging instead. Since initialization is done only once, double its averaging factor relative to that used for Recorrection.

Process Overview

**Note:** The following overview assumes the CalPod system has been installed and configured. See the CalPod Operations and Service Guide for installation instructions at: [http://na.support.keysight.com/calpod/85523-90005](http://na.support.keysight.com/calpod/85523-90005).

The following process assumes a 2-port DUT connected to the VNA ports 1 and 2 through CalPod modules as follows:

![Diagram of CalPod setup](image)

*The Blue boxes represent CalPod modules with internal Thru and Reflection states.*

1. After configuring and assigning CalPod modules to VNA ports 1 and 2, connect the CalPod modules to the VNA, directly or using short cables. Learn how to configure CalPod.

2. Setup measurements on a channel.
Note: CalPod does not support measurements at frequencies below 100 MHz.

3. An IFBW of 1 kHz or lower with eight averages is recommended.

4. Press Avg BW > Main > Averaging and set averaging as follows:
   - Averaging Factor: 8
   - Average Type: SWEEP
   - IF Bandwidth: 1 kHz
   - Turn sweep averaging on

5. Perform a full 2-port calibration for the channel with the CalPod outputs as the reference plane.

   Note: Be sure to save the calibration to a user calset.

6. Turn off sweep averaging. It will not need to be used during Initialize Channel nor Recorrect Channel because the OSL Averages will be used for those.

7. Set OSL Averages number to 2x the value that was used for sweep averaging. This will provide a more-than-adequate averaging for the Initialize Channel, which is only performed once for each calibration.

8. Click Initialize Channel to automatically perform the following steps:
   a. The Reflection states of both Calpod modules are switched in and S11/S22 are measured.
   b. The resulting measurements are stored in the channel’s Cal Set as additional standard measurements. These measurements are used to characterize the Calpod states - they are NOT used at this time to change the error correction.

   Notes:
   - Because the Reflection states in the CalPods are measured, it is not important what is connected to the CalPod when Initialize is pressed. Therefore, for highest accuracy, click Initialize IMMEDIATELY and ONLY ONCE after performing the calibration - before causing ANY cable movement.
   - If an adapter is required to connect the DUT to a CalPod, use a high-quality adapter. Any temperature drift due to the adapter is NOT recorrected.
   - Always connect the DUT as close as possible to the CalPod modules.

9. Connect the DUT to the CalPod outputs.
10. **Set OSL Averages** number to the value that was used for sweep averaging. Use this value for all subsequent Recorrections of this calibration.

11. Click **Recorrect Channel** or **Recorrect All Channels** whenever necessary. Any of the following actions will cause the current calibration to become invalid and require recorrection:

   a. Moving the CalPod modules to the ends of long cables.
   
   b. Changing the cables.
   
   c. Extreme temperature variations.
   
   d. Measurement drift over long time periods.

The following steps occur automatically during recorrection for the active channel:

   a. The Reflection states of both CalPod modules are switched in and S11/S22 are measured.
   
   b. Additional (de-embedded) error terms are computed to compensate for changed conditions from the Initialize measurements.
   
   c. Another Cal Set is created using the original name with the CalPod number appended. The modified error terms are saved to that Cal Set and applied to the channel. The measurements are now fully corrected.

**Note:** Re-correction is not supported with a descending list of frequencies, such as the output from a frequency-down converter with LO frequency higher than input frequency.

### How to start the CalPod dialog

**Using Hardkey / SoftTab / Softkey**

1. Press **Cal > Cal Sets & Cal Kits > Cal Pod...**

**CalPod** dialog box help

Learn all about the CalPod process. (Scroll up)
**Initialize Channel**  Calibrated measurements of the CalPod states are performed as initial reference data points for the active channel.

**Initialize All Channels**  Calibrated measurements of the CalPod states are performed as initial reference data points for all current channels. This command is not recommended, it is generally preferable to initialize each channel immediately following calibration.

**Recorrect Channel**  Recorrects the active channel Cal Set to match the initial reference.

**Recorrect All Channels**  Recorrects the Cal Sets on ALL channels that were initialized.

**Correct Power**

This checkbox causes power to be recorrected ONLY when source power correction data is stored as error terms in the CalSet. This occurs only when a Guided Power Cal is performed (on the standard channel) OR when an app channel is calibrated such as a FCA, GCA, IMD, and Noise Figure channel. This checkbox has NO effect when a S-parameter (only) Cal or a standard Source Power Cal has been performed, because source power correction data is not stored in the CalSet. To see if the source power cal can be modified, examine the Calset for "SourcePowerCorrection(n)" terms where "n" is the port number.

When any of the above power cals have been performed, and when this box is checked, the power output at the VNA port is adjusted to compensate for any change in path loss when Recorrect is performed. For example, if the path loss between the VNA port and the CalPod was increased by two dB following initialization, then the VNA output power will be increased by two dB upon recorrection. Do this when you add a significant amount of loss in the calibration path, or when the power level at the DUT is important.

When a significant amount of loss is introduced in the calibration path, it may not be possible to increase the source power enough to overcome the loss. In this case, an **Unleveled source** message may appear on the VNA screen.

When the checkbox is cleared, the source power level is not corrected.
**OSL Averages** Controls the number of sweeps worth of raw measurements to be measured and averaged together for the recorrection computations for each state of each CalPod.

**Assignment of CalPods to VNA Ports**

For each VNA port, select a CalPod module.

**Note:** A CalPod can be assigned up to 16 test ports in each measurement channel in multiport PNA mode.

**CalPod Setup** Starts the CalPod Setup dialog

**Delete All CalPod Cal Sets** Deletes all recorrection Cal Sets and reinstates the Initialization Cal Set.

---

**CalPod Setup dialog box help**

To start this dialog, click **CalPod Setup** in the CalPod dialog box.
**CalPod Serial Number**  Type the CalPod module (without 'sn'), then click **Add CalPod**. The new module is added to the list of available CalPod modules.

**Serial # and CalPod Types**

Shows the list of available CalPod modules. A CalPod module type may be STANDARD or THERMAL (include temperature correction). A CalPod module will be listed as a STANDARD type unless the thermal characterization data was previously loaded into the PNA from the USB flash drive that came with the CalPod. Once thermal characterization data has been loaded into the PNA, the CalPod is automatically listed in the CalPod Setup dialog.

**Note:** Loading thermal characterization data from the USB flash drive is the only method of setting up a THERMAL CalPod. Simply entering the serial number in the CalPod Setup dialog will set the **CalPod Type** to be **STANDARD**.

STANDARD and THERMAL data files are stored in **C:\e-trak\adapters**.

**Buttons**

**Utilities**  Launches the VNA CalPod Utilities used to configure the CalPod Controller and VNA over LAN.

**Note:** Before using a CalPod Controller, the LAN MUST be set up using the CalPod Utilities or an error message will be displayed indicating that the VNA is unable to communicate with the CalPod Controller.

---

**USB Ethernet Adapter Driver Install**  Installs the USB 2.0 to 10/100 Ethernet Adapter driver on the VNA.
USB Ethernet Adapter set to Static IP  Sets the USB 2.0 Ethernet Adapter to a static IP address of 192.168.0.101. This sets the VNA secondary LAN to the same subnet as the CalPod Controller.

VNA Primary LAN to Static  Sets the Primary (built in) LAN adapter to a Static IP address of 192.168.0.102.

VNA Primary LAN to DHCP  Returns the Primary (built in) LAN adapter to DHCP. This option is used when the CalPod Controller is connected directly to the VNA's LAN adapter using an RJ-45 LAN crossover cable. CalPod Controllers do not support DHCP networking.

Test  Click to test the connection between the controller and the selected CalPod module. The message box displays the connection status and temperature for both Ambient and Thermal modules. Only the Thermal module will apply test temperature for recorrection.

IP Setup  Starts the IPSetup dialog box to confirm the CalPod controller settings.

Delete  Removes the selected STANDARD CalPod module from the list.

To delete a THERMAL CalPod from the list:

1. Navigate to the c:/e-trak/adapters/itm directory.
2. Delete the .xml file associated with the CalPod serial number.
3. Exit all CalPod dialog boxes and restart the CalPod dialog.
4. The CalPod may now be removed using the Delete button.

About  Shows the CalPod software version information.

Controller Addresses

Each controller can support 4 modules directly, and up to 48 modules using external splitters. Additional controllers may be required if more than 48 CalPod modules are needed.

IP Address  Enter the IP Address of the Controller, then click Add. The IP address is configured using the IPSetup Utility. The default IP=192.168.0.100, but different static network settings can be configured if required.

Delete  Select the Controller Address, then click Delete to remove the address from the list.
CalPod Operator’s Check

This program is provided as a convenience to help determine the operational status of each 855xxA Series CalPod and its associated CalPod Controller. While this check is not intended to be a complete test, it does check each unit enough to provide greater than 95% confidence that the CalPod is functioning properly.

- When the max frequency of the CalPod is higher than the max frequency of the VNA, the full frequency range of the CalPod is not tested.
- Up to four CalPod modules may be checked at once. All four devices must be of the same frequency range.
- The software revision for the Operator’s Check code is displayed in the upper left-hand corner of the window.

Before running Op Check

The CalPod system must be installed and configured on the VNA.


Required equipment:

- An appropriate ECal or mechanical Cal Kit.
- A high-quality cable.
- A female-female adapter of the calibration connector type.
- A fixed attenuator up to 10 dB (3 dB preferred) or other frequency insensitive device with similar loss.

How to perform CalPod Operators Check

Click Utility, then System, then Service, then Verification, then Operator's Check.
Click **Setup Info** to learn more about this dialog.

Also, click **Cal Method** or **Connector** for additional explanation for these areas.

**Configure**

1. Enter information in the “Configure” area.

2. Each time a 2-port cal is performed, the results are saved in a file. The “Use Prior” selection uses the saved calibration.

3. When the calibration connector type does not mate with the CalPod connectors, perform the calibration and then use adapters to connect to the CalPod module.

4. Click **Begin** to start the Op Check.

5. Follow the prompts in the gray box.

**Op Check Results**

- The Results area shows Op Check progress.
- Click a test label for test information.
- When the check has finished, the results are saved to a text file. The default path and filename is: C:\Program Files(x86)\Keysight\Network Analyzer\Service\calpodopchklog.txt. To save multiple results, rename the file or save it to a different location.
- For assistance in troubleshooting CalPod Operator's Check failures or for additional information, see the appropriate FAQ at the CalPod web site: http://na.support.keysight.com/calpod

589
A CalPod module can be used as 1-port ECal module to calibrate channels for all measurement classes. In addition, the CalPod module can be left in place during measurements in order to refresh calibrations.

**Note:** Before using a CalPod module, ensure that Factory Cal is set to OFF (Cal > Main > Factory Cal OFF).

In this topic:

- Overview
- Characterizing the CalPod module
  - Summary
  - Detailed Steps
- Calibrating with CalPod

**More Calibration topics**

**Overview**

To use a CalPod module as a calibration device:

1. The CalPod module must be configured using the CalPod software. Learn how.

2. Unlike an ECal module, a CalPod module has no internal memory and therefore, no internal characterization data. Before a CalPod can be used as an ECal module, its internal standards must first be characterized using the ECal User Characterization wizard. In addition, the characterization data can ONLY be stored in VNA disk memory.

3. After these steps have been performed, connect the CalPod modules to the controller to perform a VNA calibration. The CalPod modules will appear in Cal Wizard dialogs just like an ECal module.
This image shows TWO CalPod modules after both have been characterized. An additional thru connection is required to complete a 2-port calibration. Subsequent ‘refresh’ calibrations can occur without making additional connections.

**Characterizing the CalPod module - Summary**

When performing a User Characterization for a CalPod module, you can establish the Calibration reference plane at the input connector, or at the output connector, depending on how you plan to use CalPod.

1. Initially calibrate the VNA.
2. After the calibration, the internal CalPod OPEN, SHORT, and LOAD standards are measured.
3. The measurements are saved to VNA disk memory. They are used when performing calibrations using the CalPod module.
Detailed steps to Perform a User Characterization

**Note:** If you have more than one CalPod module, each module must be characterized separately.

1. Perform a one-time CalPod configuration. [Learn how.](#)
2. Connect the CalPod module to the CalPod controller.
3. **Preset** the analyzer.
4. **Set up the measurement.** For best accuracy, the IF bandwidth should be set to 1 kHz or less.
5. Start the User Characterization Wizard as follows:

**Using Hardkey/SoftTab/Softkey**


**Select Module and Location** dialog box help

**ECal Module** Select a CalPod from list of connected modules and registered CalPods.

**Detect Connected ECals** Click to rescan CalPod/ECal modules.

**Location**

- **ECal Module Memory** NOT available for CalPod modules.
- **VNA Disk Memory** Enter a Characterization Name. This name appears when selecting a User Characterization to be used with subsequent calibrations.
  - Learn how to manage characterizations that are stored to VNA disk memory.
  - See the benefits of storing the User Characterization to VNA Disk Memory.
**Keyboard** Launches a keypad that can be used to type a characterization name from the VNA front panel.

**Next** Click to continue to the **Select Connectors for the Characterization** dialog box.

See note regarding extended frequency use.

---

**Select Connectors for the Characterization** dialog box help

![Select Connectors for the Characterization dialog box](image)

**In situ characterization**

When Checked:

- This initial calibration, and all DUT measurements are performed WITH the CalPod module THRU path in place (In Situ).

- Only the port with the switchable states is characterized (open, short, load, and offset short). The THRU state is not characterized.

- Select the RF2 (output) connector type and Cal Kit.

- The Cal Kit standards are connected to the RF2 Output connector, making it the reference plane. This calibration removes everything in front of the output connector from subsequent measurements.

- The User Char description will show "In Situ".

- The CalPod's states physically switch in at its RF1 port, but for an in-situ characterization the calibration used for measuring the characterized data for those states has its reference plane at-or-beyond the CalPod's RF2 port. So, for all calibrations subsequently performed using the in-situ CalPodAsECal user characterization, those calibrations also establish the reference plane at that same connector interface at-or-beyond that CalPod's RF2 output port.

When Cleared:

- This initial calibration, and all DUT measurements are performed WITHOUT the CalPod module in place. This is similar to an ECal calibration.

- Select the RF1 (input) connector type and Cal Kit. The Cal Kit standards are connected at this location,
making it the reference plane.

Next  Click to continue to the Calibrations to perform or recall dialog box.

Calibrations to perform or recall dialog box help

Perform or load a 1-port cal.

Guide me through this cal now  Click to perform a Guided calibration. A calibration kit is required for each connector type.

Note: Some PNA-L models cannot perform TRL calibration during the calibration portion of a User Characterization. However, this type of Cal can be performed using the Cal Wizard, saved to a Cal Set, then recalled at this point in the User Characterization.

Let me recall this cal from a cal set  Click to select an existing Cal Set. You cannot select a Cal Set that is currently in use. Learn more about Using Cal Sets.

Next  Click to continue to either the Select Cal Kits (Perform Cal) or Select Cal Set (Recall Calset).
When 'In situ' is selected, a cal kit is selected for port "RF2", as this will be the calibration reference plane (where standards are connected).

Provides a list of calibration kits to perform the calibration. Select the Cal Kit you will use for the port.

**Next**  Click to continue to the Select Cal Set dialog box.

The calibration that you perform will be written to a Cal Set. This dialog box allows you to select a Cal Set to overwrite, or to write to a new Cal Set. The current choice is visible below the Select Cal Set button.

**Select Cal Set**  Click to open the Select A Cal Set dialog box.

**Create new Cal Set**  Check to create a new Cal Set to store the calibration. Clear to select and overwrite a stored Cal Set.

**Next**  Click to continue to the Guided Calibration Steps dialog box.

**Note:** Remember the Cal Set name for future reference.
Guided Calibration Steps dialog box help

The first Measure page.

Connect each calibration standard to the location in the prompt.

**Measure**  Click to measure the standard.

**Back**  Click to repeat one or more calibration steps.

**Done**  Click **after** a standard is re-measured and all measurements for the calibration are complete.

**Next**  Click to continue to the next calibration step. (Does not measure the standard.)

**Cancel**  Exits Calibration Wizard.

Guided Calibration completed dialog box help

Allows you to finish the calibration and continue to the next characterization steps.

**No. Finish now**  Select to save Cal Set data.

**Yes**  Allows selection of Save options.

**Next**  Click to continue to the Exit to Inspect Quality of Calibration dialog box.
Exit to Inspect Quality of Calibration dialog box help

Allows you to exit User Characterization to validate the calibration before proceeding with the characterization.

**Back**  Allows you to repeat calibration.

**Next**  Click to continue to the Characterization Steps dialog box.

**Cancel**  Exits the Calibration.

To return to the current step:

2. In the Select Connectors for Characterization dialog box, click Next. (Previous entry is stored in memory.)
3. In the Calibrations to perform or recall dialog box, recall the Cal Set that you just performed.

Characterization Steps dialog box help

After the initial VNA calibration, the CalPod internal states are measured.

Connect the CalPod if not already connected.

**Measure**  Measures the CalPod module.
Next  Click to continue to the **Information for the New Characterization** dialog box when measurements are complete.

**Information for the New Characterization** dialog box help

Enter descriptive information that will be stored in the characterization file. This description will be viewable in the several VNA dialogs when the CalPod is selected.

Next  Click to continue to the **Write Characterized Data** dialog box.

**Write Characterized Data** dialog box help

The VNA writes User Characterization data to the VNA disk memory.

Write  Click to write data.
Summary of new User Characterization dialog box help

Summary page of characterization wizard after characterization is complete.

**Cancel**  Click to exit (characterization complete).

**Finish**  Click to exit (characterization complete).

---

**Calibrating with CalPod**

The CalPod module is now characterized and ready to be used in a VNA calibration.

When CalPod modules are selected for ALL ports of an N-port cal, Unknown Thru is the only selectable Thru method for all paths. [Learn more about Unknown Thru.](#)

The above image is the **Connectors and Cal Kits** page as seen during a SmartCal on a standard channel. A similar version of this page is shown in the Calibration Wizard for all Application channels (FCA, GCA, NFA and so forth).
Calibration Preferences

Cal type preferences are set from this dialog.

How to change Cal Preferences

Programming commands are NOT available for the preference settings discussed in this topic, although there are other Cal Preferences that can be set remotely.

Using a mouse

1. Click Response
2. Select Cal
3. Select Cal Sets & Cal Kits
4. Select Cal Preferences...

Cal Type Preferences dialog box help

This dialog is used to set which Cal Types are available, and the order in which they are selected as the default choice, during a SmartCal with Mechanical Standards. This setting is also used to set the default Cal Type for Guided calibrations using SCPI or COM.

Note: Your Cal Type settings are saved only until the NA application is closed. When re-opened, the factory default settings are restored.

The specified Cal Type order should allow you to make fewer changes to the Cal Type during a SmartCal with Mechanical Standards.

For example, in the above image, the first Cal Type on the list is TRL. When doing a SmartCal with Mechanical Standards:
If a TRL Cal Kit is available for the specified DUT connectors, then TRL becomes the default Cal Type.

If a TRL Cal Kit is NOT available, then the second Cal Type on the list (SOLT) is evaluated for compatibility with the available Cal Kits, and so forth with the Cal Types that remain on the list.

If TRL is removed from the list, that Cal Type is NOT available for selection during a SmartCal with Mechanical Standards.

Learn more about Cal Types.

See where you choose Cal Type during a SmartCal

Prioritized list of choices for default Cal Type  Shows the current list of Cal Types and the order in which they will be selected for Mechanical calibrations.

Change  Click to invoke the Modify list of default Cal Types dialog.

Restore factory defaults  Returns the list to the original selections and order. The factory defaults are in order of accuracy from highest (TRL) to lowest (QSOLT).

Cancel  Closes the dialog without making changes.

Modify list of default Cal Types dialog box help

Use this dialog to Add, Remove, and re-order the available Cal Types. There must be at least ONE selected Cal Type to perform a SmartCal with Mechanical Standards.

Unselected Cal Types  Cal Types in this list will not be presented as a choice during a Calibration.
**Selected Cal Types**  Cal Types in this list will be presented, in order, as the default choice during a Calibration. Click a Cal Type to select it, then click the following buttons to perform that operation.

**Add / Remove buttons**  Click to Add and Remove the selected Cal Types from the Selected Cal Types list.

**Move Up / Down**  Click to re-order the Selected Cal Types list.
Adapters, fixtures, and probes are often used for DUTs that have non-coaxial interfaces. This could make it difficult to calibrate with traditional cal standards. Cal Plane Manager (CPM) allows you to mathematically remove (de-embed), a characterized adapter, test fixture, or probe head from measurements.

In this topic:

- Features
- Using Cal Plan Manager
  - Cal Plane Manager
  - Characterize Adapter/Fixture and Apply
  - Calset Selection
  - Port Selection
  - Phase Pivot
  - Select Files
  - Apply De-embedding
  - Select Channels to De-embed
  - Select Calsets to De-embed
- Direct Receiver Calibration (separate topic)
- Other Actions
  - Reverse Port Order
  - Create a Transmission Only S2P File
  - Cascade Two S2P Files
  - Cascade an ENR file with an S2P file

Other Cal Topics
Features

- Characterizes adapters and fixtures in SnP files.
- Applies the characterizations to existing Cal Sets and channels.
- Writes to VNA power loss table using the S2P files of fixtures/adapters.
- Reverses the port order of an existing S2P file.
- Creates a forward-only S2P file from an existing S2P file.
- Cascades two S2P files.

Important Notes

- **Adapter/fixture** definition: Any physical 2-port device or component that is to be mathematically removed from channel measurements or Cal Sets.
- The adapter/fixture to be characterized MUST be reciprocal (S21 = S12).
- Two Tier-1 cals must be performed and saved to Cal Sets BEFORE performing the CPM characterization.

Using Cal Plan Manager

**How to start Cal Plane Manager**

**Using** **Hardkey / SoftTab / Softkey**

1. Press **Cal > Fixtures > Cal Plane Manager...**

**Using a mouse**

1. Click **Response**
2. Select **Cal**
3. Select **Fixtures**
4. Select **Cal Plane Manager**
Choose from the following, then click **Next >**:

- **Characterize Adapter/Fixture and Apply** - Given that you have already performed calibrations both before and after the adapter/fixture, SNP files are generated which characterize the adapter/fixture. These files are then used to de-embed the fixture from the channel and a new calset. Learn how.

- **Apply Adapter/Fixture** - The *.SNP files are already saved. Use these files to de-embed the fixture from the channel and a new Cal Set.

- **Direct Receiver Calibration** (separate topic)

- **Other Actions**

**Requirements**

- You must have already performed a Tier 1 calibration at the input of the fixture/adapter(s) AND a Tier 2
calibration at the output of the fixture/adapter(s) as in the above image.

- In addition, the calibrations must have been saved to Cal Sets on the analyzer.

**Note:** The mechanical switch / attenuator settings of the Tier 1 and Tier 2 cals for CPM MUST be the same settings. Also, when the span or number of points are different between the two cal tiers, there must be sufficient data points to ensure that phase wrapping does NOT occur. This is accomplished when the delta frequency for either calset is less than 12/combined length of the test port cables in meters.

**Applications**

The following images show the calibration planes of the Tier 1 and Tier 2 calibrations:

<table>
<thead>
<tr>
<th>Probe Head on Wafer</th>
<th>Probe Head on Fixture</th>
<th>Waveguide Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Probe Head on Wafer" /></td>
<td><img src="image2" alt="Probe Head on Fixture" /></td>
<td><img src="image3" alt="Waveguide Adapter" /></td>
</tr>
</tbody>
</table>

**Procedure**

**Configuration**  Select the number of adapter/fixtures to be characterized and de-embedded.

- Choose **One fixture** when you have a single fixture/adapter on either the input or output of the DUT.
- Choose **Two fixtures** when you have a fixture/adapter on BOTH the input AND output of the DUT.

**Browse** - Starts the following Calset Selection dialog.

**DC Phase** - Starts the Phase Pivot dialog.

Click **Next >**
Calset Selection dialog box help

Choose from the listed Cal Sets on the analyzer to use for the Tier 1 and Tier 2 calibrations.

- The Tier-1 calset MUST be from a calibration that was performed at the input to the adapter/fixture.
- The Tier-2 calset MUST be from a calibration that was performed at the DUT reference plane.

Click **Filter Calsets** to start the following dialog.
Check Enable Filter, then provide advanced filter requirements to narrow the search for appropriate Cal Sets.

**Filter Options**

**Measurement Class** - The classes listed are those that are enabled on the analyzer.

**Cal Type** - Filter for 1P (one-port) or 2P (two-port) Cal Sets.

**Calset Name Includes** - Filter to include any text that appears in the calset name.

**Calset Description Includes** - Filter to include any text that appears in the calset description.

**Cal Ports** - Filter to include only the analyzer ports to be de-embedded.

**Active Cals** - Filter to include only the Cal Sets that are currently in use on the analyzer.

**Last Modified Between** - Filter to include only the Cal Sets that were last modified between the two specified dates.

---

**Port Selection** dialog box help

In the previous dialog, when a multiport calset is selected for a characterization that involves fewer ports, then select the port in the calset that is used to characterize the fixture/adapter.
**Phase Pivot dialog box help**

![Phase Pivot dialog box](image)

For most devices, the projected phase of S21 at DC crosses the X-axis between 0° and -180°.

The phase pivot point specifies the center of the phase window. It is normally 1 Pi wide. The default value of 0° should be adequate for the majority of adapters.

However, when characterizing electrically long cables, cables with significant mismatch, or high noise in the measurements, it is possible that the projection of phase goes above 0°. This results in a 180° phase difference between the results computed by CPM versus the results you might get by measuring the same adapter with a 2-port calibration.

In these cases, you may have to change the default value to capture the projected phase of S21 at DC.

---

**Select Files dialog box help**

For each Fixture (A and B):

- Click **Browse**, then navigate to the folder to where the S2P files are to be saved.
- Enter a filename, then click **S2P Format**, then choose the format in which the data is to be saved:
Click **Apply** to continue to de-embed the fixture.

Click **Finish** to end with the characterization and close the dialog.

---

### Apply De-embedding dialog box help

Given one S2P file for each fixture/adapter, this dialog will remove the effects of the fixture/adapter from either:

- one or more Calsets
- or one or more channels.

**For each Fixture (A and B)**

1. Check to enable fixturing.

2. Select the VNA port the fixture is connected to.

3. Click **Browse** , then navigate to the S2P file that represents the fixture/adapter.

4. **Reverse Ports** - S2P files that are created using CPM ALWAYS reference port 1 of the fixture/adapter on the side closest to the analyzer and port 2 of the fixture/adapter ALWAYS on the DUT side of the device as in the following image. The application of the S2P file (this dialog) assumes this same orientation. If your S2P files were created using a different (external) application, check the orientation and check **Reverse Port Order** if necessary.
5. Choose one of the following De-embed Options

- **De-embed from Calsets** - Starts the Select Calsets to De-embed dialog box.
- **De-embed from Channels** - Starts the following Select Channels to De-embed dialog.

**Select Channels to De-embed** dialog box help

De-embedding is performed and applied to specified channels on the analyzer.
Select one more channels currently displayed on the analyzer from which to de-embed the adapter/fixture.

**Select Calsets to De-embed** dialog box help
This dialog appears when **De-embed from Calsets** is selected in the previous dialog.

De-embedding is performed and applied to specified Cal Sets. This allows you to easily apply de-embedding in the future by simply applying the de-embedded calset to any channel.

**Select Calsets**: Select the Cal Sets to which de-embedding will be applied.

**Properties**: View information about the corresponding calset.

- **Overwrite Selected Calsets** - The selected Cal Sets are overwritten with the adapter/fixture de-embedded.

- **Create New Calsets** - Select the Cal Sets from which new Cal Sets will be created.

Click **Options** to start the following dialog.
**Auto Name** - By default, a new calset will be created using the old calset name with the specified text ("CPM" by default) appended to the beginning of the name. You can change the specified text.

**Ask me for each name** - Starts the following dialog when **OK** is pressed.

**Preserve Active Cal and Channel associations** - When checked (default) the new de-embedded Cal Sets will be used to correct the same displayed channels as the current Cal Sets.

The Existing Calsets that you selected for de-embedding appear in the left column.

The proposed New Calset names appear in the right column.

To change the new Calset name, select, then edit the name.

When finished, click **OK**.
Select one of the actions:

- Reverse the Port Order of an S2P File.
- Create a Transmission Only S2P File.
- Cascade two S2P files.
- Cascade an ENR file with an S2P file

**Write to VNA power sensor loss table.** Loads the S2P Frequency / Loss pairs into the VNA Power Loss Compensation table to compensate for losses that occur when using the device to connect a power sensor to the measurement port during a Source Power Cal.

- **Enable Power Sensor Loss Table**

  Then choose from the following:

  - **Compute fixture from Calsets.** Computes the transmission loss of the fixture based on the selected Cal Sets. This choice is NOT available until two valid Cal Sets are selected.

  - **Use an S2P file.** Uses the S21 data in an existing S2P file to build the VNA’s power loss table. Select, then click **Browse**, then navigate to the S2P file, then click **Next >**.

  **Note:** In the VNA Power Loss Compensation table, loss is expressed as a positive number. CPM assumes that any negative S21 value in the S2P file is a loss and therefore multiplies the S21 values in the file by -1 to express that value as a positive number. This ensures proper handling of the offset during a source power cal.
S2P files that are created using CPM ALWAYS reference port 1 of the fixture/adapter on the side closest to the analyzer and port 2 of the fixture/adapter ALWAYS on the DUT side of the device as in the following image.

This action causes ports to be reversed on an existing S2P file.

- The data for S11 becomes the data for S22 and vice versa.
- The data for S21 becomes the data for S12 and vice versa.

The resulting file is written in the standard S2P file format.

1. **Original** - Navigate to the S2P file to be reversed.

2. **Reversed** - Navigate to the folder where the new reversed S2P file will be saved. Enter a filename. By default, the file is saved to the same folder using the filename: `<old filename>_Reversed.s2p`

3. Click **Finish**. The Reversed file is saved to the specified location.
From an existing S2P file, this feature allows you to zero the S11, S22, or both data columns. The original S21 and S12 data are preserved. This is useful for Enhanced Response calibration / de-embedding.

**Original** - Click **Browse**, then navigate to the file to be modified.

**Modified** - Click **Browse**, then navigate to the folder and enter or change the filename of the resulting S2P file. The file select dialog allows you to change the format of the data. Click **Format**, then choose from the following:

- Log Magnitude & Angle (default)
- Lin Magnitude & Angle
- Real & Imaginary

**Modifications**

Choose to Zero the S11, S22, or both data columns.

Click **Finish**. The transmission only file is saved to the specified location.
Cascade Two S2P Files dialog box help

This dialog combines the losses and phase shift of two S2P files into a single S2P file.

The stimulus settings of the two input S2P files need not be identical. The frequency range of the cascaded S2P file will be the frequency range that is common between the two input files. In addition, the cascaded S2P file will use the data points of the input file with the denser data points.

For example:

**S2P #1**: Frequency range = 1 GHz to 5 GHz; 201 pts.

**S2P #2**: Frequency range = 2 GHz to 6 GHz; 1001 pts.

**Cascaded S2P**: Frequency range = 2 GHz to 5 GHz using the data points of S2P #2.

**S2P #1** - Click **Browse**, then navigate to one of the S2P files to be cascaded.

**S2P #2** - Click **Browse**, then navigate to the other S2P file to be cascaded.

**Cascaded S2P** - Click **Browse**, then navigate to a folder and enter the filename of the resulting S2P file.

Click **Finish**. The cascaded file is saved to the specified location.
This dialog generates a new ENR file by embedding an adapter to an existing ENR file.

**Input ENR** - Click **Browse**, then navigate to ENR files to be cascaded.

**Input S2P** - Click **Browse**, then navigate to the S2P file to be cascaded.

**Output ENR** - Click **Browse**, then navigate to a folder and enter the filename of the resulting ENR file.

Click **Finish**. The cascaded file is saved to the specified location.
Port Sub-Setting Examples

S-Parameter error correction is described by the number of ports contributing to the measurement: 1, 2, 3 or 4 port corrections are common. They can be represented in a visual shorthand like the matrix below.

Source ports occupy each column. Receiver ports occupy each row.

**Example 1:** this table shows four independent 1 port calibrations. (F1) All reflection measurements S11, S22, S33, S44 will all be Full 1 Port corrected.

<table>
<thead>
<tr>
<th>Source Port</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>F1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>F1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>F1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>F1</td>
<td></td>
</tr>
</tbody>
</table>

**Example 2:** this table shows two independent 2 port calibrations. The parameters that occupy the blank cells (S13, S14, S23, S24, S31, S32, S41, S42) are uncorrected.

<table>
<thead>
<tr>
<th>Source Port</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>F</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>4</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

**Example 3:** Full 3P correction for ports 1, 3 and 4. Any port 2 parameters will be uncorrected.

<table>
<thead>
<tr>
<th>Source Port</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>3</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>4</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>
Example 4: Full 4P correction. All S-Parameters across the first four ports are corrected.

<table>
<thead>
<tr>
<th>Source Port</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>2</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>3</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>4</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

Calibrating all ports of the network analyzer allows measurement flexibility but it may not be appropriate for all test conditions. Port Subsetting can be used to devolve a multi-port to fewer ports as needed. This is useful in the following situations:

- when a calibration includes ports that are not connected to the DUT.
- When it is safer or more accurate to only acquire data in one direction which can be the case with high power amplifiers and when there is substantial loss at the DUT output port.

By default, Port Subsetting is disabled and all ports are selected as denoted by the “X” in the port indicators. Enabling port subsetting with all ports included is the same as turning port subsetting OFF. Below we show some examples of subsetting a full 4P correction.
Example: Active 4-Port Calibration

The following examples assume there is an active 4 Port calibration.

An active 4 Port calibration:

<table>
<thead>
<tr>
<th>Source Port</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>2</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>3</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>4</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

Example 5: To subset the calibration to ports 1, 3 and 4, de-select port 2 and Enable port subsetting.

This yields a correction grid like Example 3 above, except the shaded cells are no longer empty.

The multiport correction is devolved from 4 ports to 3. But what happens when measurements involving port 2 are created?

<table>
<thead>
<tr>
<th>Source Port</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>F</td>
<td>eR</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>2</td>
<td>eR</td>
<td>F</td>
<td>eR</td>
<td>eR</td>
</tr>
<tr>
<td>3</td>
<td>F</td>
<td>eR</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>4</td>
<td>F</td>
<td>eR</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

Because the calset already contains error terms for port 2, the best result for port 2 measurements is as follows:
• Enhanced Response (eR) correction to transmission terms (S21, S32, S42, S12, S32, S42)

• 1 Port correction (F1) to S22.

These corrections give a much better result than leaving the parameter uncorrected while not requiring any extra acquisition time. This is sometimes referred to as “Best Effort”.

Example: Subsetting by De-Selecting All Ports

This is a convenient way to reduce a multiport correction to a set of single port corrections.

This example yields the following correction grid.

**Example 6:** Our original 4 port correction has been devolved to a set of one port and enhanced response corrections. Again, because we have the error terms this grid produces the best measurement results without sweeping multiple source ports. (Best effort).

<table>
<thead>
<tr>
<th>Source Port</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive Port</td>
<td>1</td>
<td>F1</td>
<td>eR</td>
<td>eR</td>
</tr>
<tr>
<td>2</td>
<td>eR</td>
<td>F1</td>
<td>eR</td>
<td>eR</td>
</tr>
<tr>
<td>3</td>
<td>eR</td>
<td>eR</td>
<td>F1</td>
<td>eR</td>
</tr>
<tr>
<td>4</td>
<td>eR</td>
<td>eR</td>
<td>eR</td>
<td>F1</td>
</tr>
</tbody>
</table>

Port Sub-Setting SCPI Commands

**Note:** With SCPI you have more control over the grid.

SENS:CORR:METHod:PORT:SUBSet:FULL:VALue
This command sets the ports that are included in the subsetting.


This command sets which ports are included the "best effort" treatment.

**Note:** The ports lists for these two commands must be mutually exclusive.

**SCPI Example: Assuming a 4 port calset, port subset to a 3 port cal on ports 1,3,4**

This set of commands yields the following grid.

SENS:CORR:METH:PORT:SUBSet:FULL:VAL 1,3,4

SENS:CORR:METH:PORT:SUBSet:RESP:VAL 0

<table>
<thead>
<tr>
<th>Source Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4</td>
</tr>
<tr>
<td>1 F F F</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3 F F F</td>
</tr>
<tr>
<td>4 F F F</td>
</tr>
</tbody>
</table>

This set of commands yields the following grid.

The same result occurs when you subset to ports 1,3,4 from the dialog.

SENS:CORR:METH:PORT:SUBSet:FULL:VAL 1,3,4


<table>
<thead>
<tr>
<th>Source Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4</td>
</tr>
<tr>
<td>1 F eR F F</td>
</tr>
<tr>
<td>2 eR F1 eR</td>
</tr>
<tr>
<td>3 F eR F F</td>
</tr>
<tr>
<td>4 F eR F F</td>
</tr>
</tbody>
</table>
Use this selection to modify an existing Cal Set with a calibration path through the front-panel jumper loops (direct access receiver configuration).

For example, this might occur on a Differential IQ channel when using more than four ports. If only the source is being used for the measurement, the receiver can be accessed and calibrated using the front-panel jumper cables.

**Note:** Only corrected scalar magnitude measurements are possible at the direct-access receiver. The original phase data is preserved in the Cal Set and is NOT modified by the Direct Access Receiver Cal.

See the Cal Plane Manager main help page.

Choose Direct Receiver Calibration, then click Next >
Select a Cal Set to modify. By default, the Cal Set for the active channel is selected.

A valid Cal Set has the following attributes:

- The Cal Set has at least one reference receiver Response Tracking error term AND the test port receiver Response Tracking error term in it. These must cover the same frequency range. The Differential IQ Cal All Cal Set will meet this requirement.

- If the active Cal Set does not meet the above criteria the Next button is NOT available.

Select a Source Port to be used for Direct Receiver Calibration.

The ports that are listed have a calibrated reference receiver in the selected Cal Set.

Select (one or more) Receivers to be calibrated through a front-panel access port.

The Receivers that listed are have the same corrected frequency ranges as the selected Source Port.

For convenience, both standard and logical receiver notation are used in the list of receivers.
Using the same cable for this calibration that is used in the measurement, connect the cable as shown in the diagram.

If an additional adapter is used to connect the cable to the source port, the very small magnitude effects of that adapter will not be removed from the Cal Set.

**Power Level:** If needed, change the power level of the source for this measurement only.

The Cal Set that you selected in the first dialog appears.

You can modify that Cal Set, or enter a new Cal Set name to be created.

Click **Finish**. The following warning message appears.
Modify existing calset "2014-05-22_01-26:50"?
All VNA measurements have some level of uncertainty. The Keysight Uncertainty Calculator has always provided a method to calculate the uncertainty of your measurement based on measurement settings, connectors, and a specified Cal Kit. Now, with S9x015A/B, you can display the measurement uncertainty dynamically ('real-time') on the same screen as the measurement trace and get more reliable uncertainty estimates.

- **Overview**
  - Summary: How to show Uncertainties
  - Limitations
  - The Uncertainty Manager
    - Uncertainty Workspace
    - Port Noise Characterization
    - Cable Repeatability Characterization
    - Uncertainty Cal Kits
    - Options
      - Plot Characterizations (separate topic)
  - Perform an Uncertainty Calibration
  - Display Uncertainties
  - Save Uncertainty Measurement Data
  - Set Up Power Sensor Uncertainty (separate topic)

### Other Calibration Topics

#### Overview

The following three sources of uncertainty are characterized and accounted with Dynamic Uncertainty for S-Parameters:

- **Calibration Standards** - For select Keysight Cal Kits, the standards have already been characterized to a known degree of uncertainty. Alternatively, custom Cal Kits with custom uncertainty can be used. Learn more.
- **Noise** - The noise at each test port must first be characterized using LOAD and SHORT standards at measurement settings (frequencies, RF power, IF bandwidth, averaging factor) that are similar to the final uncertainty measurements. These characterization measurements are stored with workspace files and can be included in subsequent uncertainty measurements. [Learn more](#)

- **Cable and connector repeatability** - The repeatability of test port cables must first be characterized using LOAD and SHORT standards as terminations. These characterization measurements are stored with workspace files and can be included in subsequent uncertainty measurements. [Learn more](#).

**Summary: How to measure and display Uncertainties**

Highlighted steps are required ONLY for noise and repeatability characterizations. They are NOT required if already completed prior to calibration.

1. Setup the instrument for characterization.
2. In Uncertainty Manager, assign Cal Kits, ports, and cables for measurement.
3. Perform noise characterization.
4. Complete a standard calibration of the system.
5. Perform repeatability characterization.
6. Setup the VNA for the measurement.
7. Perform an uncertainty calibration using an Uncertainty Cal Kit. [Learn how](#).
8. Set Uncertainty Trace Properties to control what and how uncertainties are displayed. [Learn how](#).

**Note:** When recalling a Uncertainty Cal Set, it is also necessary to turn Correction ON.
**Filter S21 measurement with uncertainty bars.**

**Limitations**

- ONLY single-ended S-parameter and Power (Standard class) measurements are supported.
- SOLT calibrations are NOT supported. Instead, the preferred choices are SOLR (unknown THRU) and QSOLT.
- Multiple Cal Kits per calibration are NOT supported. For multiple Cal Kit scenarios, create a custom Cal Kit using existing standards. Learn how.
- ONLY the following ECal families are currently supported:
  - N4433A 3.5 mm 20 GHz 4-port
  - N4690C Type-N 300 kHz to 18 GHz 2-port
  - N4691B 3.5 mm 26.5 GHz 2-port
  - N4692A 2.92 mm 40 GHz 2-port
  - N4693A 2.4 mm 50 GHz 2-port
  - N4694A 1.85 mm 70 GHz 2-port
  - All N469xD ECal modules are supported, except all DC options (0DC)
- ECal User Characterizations are NOT supported.
- Sliding Load Cal standards are NOT supported, even though some Uncertainty Cal Kits contain sliding load standards.
- Multiport Testsets are NOT supported.
Cal All Channels is NOT supported.
This application does not support banded millimeter systems.

**When Programming Dynamic Uncertainty**

- Calibrations can be performed for ONLY ONE channel at a time.
- Putting Error Term data into Uncertainty Cal Sets using remote commands is NOT supported.

**Uncertainty Manager**

Uncertainty Manager is used to perform Noise and Cable Repeatability characterizations, to edit Uncertainty Cal Kits, and make settings that are used during a subsequent uncertainty calibration. Once completed, the characterizations and settings can be saved to a workspace file for future calibrations.

Uncertainty Manager is a separate application that runs at the same time as the VNA application. It is closed ONLY when the VNA application is closed.

**Important:** To return to the VNA from Uncertainty Manager, click **File**, then **Switch to VNA**.

**How to start Uncertainty Manager**

**Using Hardkey/SoftTab/Softkey**

1. Press **Cal > Cal Sets & Cal Kits > Uncertainty Setup...**

**Using a mouse**

1. Click **Response**
2. Select **Cal**
3. Select **Cal Sets & Cal Kits**
4. Select **Uncertainty Setup...**
Most Uncertainty Manager settings can be made both from the Menu and by right-clicking an item.

Uncertainty Manager Workspace

The Uncertainty Manager Workspace contains all of the settings and characterizations that are used in an Uncertainty Calibration.

- The title bar on the Uncertainty Manager dialog shows the current workspace (*.ml4) filename.
- When an uncertainty calibration is performed, the workspace is automatically saved. This file is automatically reloaded when Uncertainty Manager is started.
- To save the current workspace, click File, then Save, then enter a name. The factorydefault.ml4 (workspace) file can NOT be overwritten. If you do NOT change the filename while saving with factorydefault.ml4, then _copy is appended to the filename before it is saved.
- Plot settings are NOT saved to the workspace file.

Perform Port Noise Characterization

The Noise characterization measures the VNA source and receiver noise for a pair of test ports.

Measurement Settings

- When included in uncertainty analysis, the Noise Characterization is interpolated over frequency, but NOT
extrapolated. Therefore, the characterization should be performed over the entire frequency span of the VNA with a point spacing of about 1600 points for the entire span. Avoid very dense point spacing as this slows the characterization and puts a burden on VNA memory.

- The IFBW for the characterization should be within 10x of the IFBW of the uncertainty calibration. For example, a characterization performed at 10 kHz will allow the IFBW for the uncertainty calibration to be between 1 kHz to 100 kHz.

- The power level is generally tolerant of variation, so performing the characterization within +/- 10 dB of the power level of the uncertainty calibration is acceptable.

- The averaging factor of the characterization should be within a ratio of 1:100 of the uncertainty calibration. For example: an averaging factor of 100 for the characterization would allow an averaging factor of between 1 and 1000 for the uncertainty calibration.

**How to Perform Noise Characterization**

It is NOT necessary to perform or recall a VNA calibration before performing a noise characterization.

Start the noise characterization by doing either of the following:

- From the menu, select **Noise**, then **Measure Noise**
- Right-click a VNA Port, then select **Measure Noise**

If the current VNA settings (frequencies, RF power, IF bandwidth) are NOT the same as that with which the current noise data was measured, then the following dialog appears:

Select one of the following:

- **Clear noise data on all ports** - Use the current VNA settings to perform a new Noise characterization.
Switch to VNA (screen) - Manually change the VNA settings to match those with which the current noise data was measured. Do this to recharacterize the noise using the previous settings.

Click (Start) Measure. The VNA port noise characterization is performed using a 2-step measurement process involving two ports at the same time:

1. The noise floor is characterized with a LOAD (or low-reflective device) connected to each test port. If test cables will be used in your uncertainty measurement, connect them for the noise characterization to get a better noise estimate, especially when the cables are lossy.

2. Trace noise is characterized with a SHORT (or high-reflective device) connected to each test port.

Press OK when finished.

Save the Noise Characterization

The noise data can now be saved and later recalled for use with uncertainty measurements.

- To store the entire workspace which includes the noise characterization, click File, then Save.
- To store the JUST noise data, in the menu bar, click Noise, then Save Noise. Enter a descriptive filename. The noise data is saved to a *.ndf file. You would do this if you plan to share the data with another VNA. This ASCII (csv) file can be accessed using Windows Explorer.

Cable Repeatability Characterization

Uncertainty Manager comes preloaded with a database of uncharacterized cables. In addition, there are also some 'Example' cables which include repeatability characterization data. However, you should perform your own repeatability characterization for cables which you intend to include in the uncertainty analysis.

It is likely that you will need to add new cables to the list. For example:

- If the cable requires an adapter to connect your DUT, you should create a new cable entry and characterize it. Adapters do have an impact on how the repeatability propagates.
- To connect your DUT directly to the VNA front panel connector or connector saver, create and then assign a "zero-length Test Port" cable.
- Learn how to add new cables and connectors.

BEFORE performing the repeatability characterization

1. Assign the cable to at least one VNA port. This is done ONLY to perform the characterization. During uncertainty measurements, the cable can be assigned to any VNA port. The characterization is performed at the lowest port number to which it is assigned.

- To associate a single cable with all ports, select the cable. Then right-click, then select Assign Cable to All Ports.
To associate a single cable with a single port, select the port. Then in the **Properties** pane, select **Cable Name**, then click next to the current cable name, then select a cable.

- To perform an independent repeatability characterization for each port, ensure that the assigned cable or test port name is unique.

2. Perform or recall a valid calibration at the test port. To achieve the best characterization:

- Select the entire frequency span over which the cable is specified or the frequency span of the VNA if it is smaller. When performing the uncertainty calibration over a smaller span, it will be interpolated.
- Select a point spacing or about 1600 points. Avoid very dense point spacing as this slows the characterization and puts a burden on VNA memory.
- Select an IFBW of 1 kHz or less.
- Select a high power level without exceeding the VNA linearity specification.

**Perform the repeatability characterization**

1. Select the cable to characterize.

2. If there is already repeatability data associated with the cable, right-click the cable, then click **Clear Repeatability**.

3. At the bottom of the **Properties** pane, click **Measure Repeatability**. The following dialog appears:

![Repeatibility Characterization Dialog](image)

4. Choose **Number of Iterations** (20 is recommended). An Iteration is a measurement with each termination. A set of iterations are performed first with the LOAD (or low-reflective device), and then again with the SHORT (or high-reflective device). For 20 iterations, a total of 40 measurements are performed.

5. Connect the cable to the specified test port, connect the terminating LOAD, and click **Start Measurements**.

6. Between each measurement, either move the cable, or remove and reconnect the standard, or both
5. Do NOT remove and reconnect the cable from the VNA test port as this will impair the calibration. Use the same torque wrench as you use during ordinary calibrations and measurements.

6. Follow the prompts at the top of the dialog. When finished, press **OK**.

**Save the repeatability characterization**

The repeatability data can now be saved and later recalled for use in an uncertainty calibration. The repeatability data can be used over any subset of the characterized frequency range. When used in an uncertainty calibration, measurement settings such as power and IFBW are NOT relevant.

- To store the entire workspace which now includes the repeatability measurements, click File, then Save.
- To store JUST the repeatability data, click Save Repeatability and enter a descriptive filename. The repeatability data is saved to a *rep file. You would do this if you plan to share the data with another VNA. This file can be accessed using Windows Explorer.

**Add New Cables and Connectors**

- Click Cables, then New Cable. 'New Cable' appears in the list of connectors.
- Use the Properties pane to change the Name and assign a Connector Type and Gender.
- To add a new connector, add it using the Uncertainty Manager **CalKit Editor**.

---

**Uncertainty Cal Kit Editor**

From Uncertainty Manager, you can access the Uncertainty Cal Kit Editor by clicking the **CalKit Editor** tab. Also from the **VNA Cal Kits dialog**, click **Uncertainty Cal Kits** button.
**The Uncertainty Cal Kit Editor**

**Uncertainty Cal Kits and Standards**

To see Cal Kit uncertainty contributions in an Uncertainty calibration, the standards must include uncertainty data. This is best done with 'databased' standards. This means that the standard definitions include characterized data, not just polynomial values as in typical Keysight Cal Kits. Polynomial standards can include mechanical dimension uncertainties. Learn more.

As shown in the above image, three of the Keysight 8505x Cal Kits have already been 'databased' for use with Dynamic Uncertainty. In addition to being databased, the standards include uncertainty data.

- Keysight Uncertainty Cal Kits can NOT be edited.
- No Keysight Waveguide Cal Kits have been databased.
- If you own a Cal Kit other than those that have been databased, you can either:
  - Characterize your own uncertainty standards. Learn how (internet connection required).
  - Add uncertainty values to represent the mechanical uncertainties of your cal standard. Learn how.

All standards have properties as shown in the Properties pane above.

All standards have the General information shown, and Connector Type and Gender.

All standards have an icon preceding its Name in the Items list. This gives a hint as to the type of standard.

- Signifies a databased standard. The characterization data is stored in the workspace file. See How to
Characterize Uncertainty Standards.

- Signifies an adapter. Adapters are NOT databased. They include only nominal 'delay'. This delay value can be edited if it is in a custom Cal Kit.

- Signifies an OPEN. All other icons represent a specific type of polynomial-based standard.

Connectors

To add a new connector, right-click on **Connectors**, then **New Connector**. "New_Connector" appears in the list of connectors. Edit the name in the Properties pane. It will now appear in the Connector Type selector for a standard.

Create a Custom Cal Kit

You may need to create a custom Cal Kit for one or more of the following reasons:

- When performing an uncertainty calibration, only ONE Cal Kit can be selected for all ports. For multiple Cal Kit scenarios, create a custom (combined) Cal Kit which contains standards with different connector types using the uncertainty Cal Kit Editor.

- Create a Cal Kit to contain new standards.

Procedure

1. On the CalKit Editor tab, click **CalKits**, then **New CalKit**.
2. In the Properties pane, change the **Name** and **PartNumber**.
3. Fill the new Cal Kit with standards the following methods. There can be more than one connector type in a Cal Kit.

   a. **Copy standards from other Cal Kits.** Standards can simultaneously belong to different Cal Kits.
      i. Select the standard in a different Cal Kit to be copied.
      ii. Click **Edit**, then **Copy** (or Ctrl+C)
      iii. Select the new Cal Kit.
      iv. Click **Edit**, then **Paste** (or Ctrl+V)
      v. Make edits to the standard as necessary. You can not edit databased standards that have been copied from Keysight standards.

   b. **Create new databased or polynomial standards.**
      i. Click **CalKits**, then **New standard**
      ii. Select the standard type. Choose from: Short, Open, Load, Line, Thru (defined), Reciprocal (Unknown Thru), DatabaseStd.

         - **DatabaseStd** is the ONLY standard type for which you can upload uncertainty characterization data. These Database standards can be renamed and associated with a Equivalent Model. See the following section **Characterize Uncertainty Standards**.

         - All other standard types allow only polynomial data. These standards can include dimension
4. Click Edit **CalKits**, then **Edit Class**. Learn more about editing Cal Kit Classes from the VNA Cal Kit Editor.

**Characterize Uncertainty Standards**

To be used in an Uncertainty calibration, the standards must include uncertainty data. This is best done with a 'databased' standard. This means that the standard definitions include characterized data, not just polynomial values.

Select Keysight Cal Kits have already been characterized with uncertainty data. If you own one of these Cal Kits, shown by name in the Cal Kit editor, there is nothing more required.

If you do NOT own one of these Cal Kits, you can characterize your own uncertainty standards.

**Procedure**

1. Perform an uncertainty calibration using a Keysight Uncertainty Cal Kit.
2. Measure the device to be used as a standard.
3. Save the data as *.dsd file Learn how.
4. Import this file at the New Standard dialog.

**Mechanical Dimension Uncertainties**

For Cal Kits that include polynomial standards to be used for PCB, on wafer, or coaxial applications, you can describe the mechanical dimensions and tolerances. From these, Uncertainty Manager will calculate the electrical uncertainties. An example of this type of standard is provided in the Uncertainty Manager **Microstrip Example** Cal Kit.

**How to enter physical dimensions and tolerances**

1. Select the polynomial standard.
2. In the Properties pane, change **Type** any of the selections other than Ideal or Waveguide.
3. Enter the relevant physical dimensions accordingly to the manufacturer's specifications.
4. For each dimension, in the Uncertainty section, provide reasonable estimates of the standard parameters. By default, the uncertainty probability density function is assumed to be Gaussian. All uncertainty values are assumed to be standard deviations (1 sigma uncertainties).

**Calibration Options**
**Calibration Options** dialog box help

To start this dialog, in Uncertainty Manager, click **File**, then **Uncertainty Options**.

![Uncertainty Options dialog box]

Check each to include Noise, Cable Repeatability and Standard Definition in the uncertainty calibration. You can later decide to NOT include these in the measurements using the **Trace Properties** dialog.

**Max Uncertainty Pts** - Because of limited VNA memory, uncertainty is computed for a maximum of 501 equally-spaced data points. When applied to a measurement, the data points within each step is assumed to have the same uncertainty. To save VNA memory, reduce this number.

**Important Notes**

- These settings control the contributions used ONLY during the uncertainty calibration. They are independent from the trace properties settings, which control the contributions used during the DUT measurement.
- These settings are NOT saved with the workspace file, but to a hidden 'global' file.

**Service Mode** - When requested by Keysight service personnel, check this box before calibration to save uncertainty data for troubleshooting purposes.

**Perform an Uncertainty Calibration**
Important: BEFORE starting an uncertainty calibration, ensure that the current Uncertainty Manager workspace contains ALL of the settings that will be used for the uncertainty calibration. Here are the critical settings:

- For each port to be calibrated, select the **Connector Type** and **Port / Cable** assignment.
- To include Noise data and Cable Repeatability data in the calibration:
  - The data must be available in the current workspace.
  - Be sure that the data covers the frequency range of the calibration.
  - Check the appropriate 'include' boxes in Options dialog.

Start **Cal Wizard**, then select **Guided Cal**.

1. At the Select Ports for Guided Cal dialog, select the ports to be calibrated, then check **Use Uncertainties**, then **Next >**.

2. At the Select Cal Kit dialog:
   - The connectors that you chose from the Uncertainty Manager are selected automatically and can NOT be changed.
   - Select the (Uncertainty) Cal Kit if more than one exists.

3. Optionally, click **View/Modify Cal** to view / change the Cal Types and connection steps for the pending calibration. The following dialog appears:

   **Uncertainty Calibration Properties** dialog box help
The port configurations with selected Cal Types appear in a matrix.

- 1-port Cals appear at the intersection of the same port number. For example Port 1 and Port 1.
- 2-port 'Thru' Cal Types appear at all other intersections.
- The dark-shaded areas are a mirror image of the selectable 2-port intersections. They can NOT be edited.

**How to change Cal Types**

- Click in any white box, then make a selection.
- The Connection List (cal steps) change accordingly.
- **Invalid** appears when the Cal Types are NOT sufficient to perform a valid calibration. One or more of the Cal Types must be changed.
- **None** appears when no Cal Type is required at that port intersection.

**2-port (Thru) Cals** are selected by default based on the following (in order):

1. The standards in the selected Cal Kit. **ECal** always appears when a supported ECal module with the corrected connector type and gender is connected to USB. NOT all ECal modules have been databased. Learn more.
2. The most recent multiple-port calibrations.
3. If the most recent calibration is NOT possible, then priority is given to Cal Types in the following order:

   1. **LRM** (Line-Reflect-Match) The 'Line' (L) standard is used as a Thru by defining it in the Thru Class.
   2. **TRL** (Thru-Reflect-Match) - Learn more about TRL
   3. **SOLR** (Un-Thru) - Learn more about Unknown Thru
4. **QSOLT** (Quick SOLT) - Only ONE 1-port Cal is performed (on the lower port number by default). Then a Defined Thru. To perform the 1-port cal on the higher port number, select 'Thru' between the ports, then change the higher port number to 'One-port Cal'.

**1-port Cals**

- **One-port Cal** appears at the intersection of the same port numbers. The above image shows this for all four ports.

- **Automatic** indicates that the selection is flexible. The connection list reflects the port numbers on which to perform 1-port cals. However, any of the Automatic selections can be changed to 'One-port Cal'.

**Default Cals**

Select your Cal Type preference for the current calibration.

4. Follow the remaining connection prompts to complete the calibration.

5. Save the calibration and apply the resulting Cal Set with uncertainties as you would any Cal Set.

**Important:** Add a descriptive prefix to the Uncertainty Cal Set name such as "UNC<calset name>". This will help identify it as an Uncertainty Cal Set. This does not occur automatically.

**Set Uncertainty Trace Properties**

You have performed Noise and Cable Repeatability (optional).

You have performed a calibration using an Uncertainty Cal Kit.

Now, you want to show the uncertainties on the displayed trace.
Start the **Trace Uncertainty Properties** dialog by doing one of the following:

1. Right-click on the trace status above the grid, then click **Uncertainty**.

**Using Hardkey/SoftTab/Softkey**

1. Press **Math > Analysis > Uncertainty Analysis**...

**Using Menus**

1. Click **Response**
2. Select **Math**
3. Select **Uncertainty Analysis**...

---

**Uncertainty Trace Property** dialog box help

**Trace** - Choose the displayed trace to which the settings on this dialog will be applied. Or check **Apply to all traces** to override this selection.

**Trace Type**

ONLY Ellipse is supported in Smith Chart or Polar display format.

- **Normal** - Display the trace without uncertainties.
- **UMax** – Display the trace as the uncertainty maximum (measured or memory data PLUS upper limit uncertainty values).
- **UMin** - Display the trace as the uncertainty minimum (measured or memory data MINUS lower limit uncertainty values).
**UBars** – Display the uncertainties as “error bars” around the trace.

**Shade** – Display the uncertainties as a shaded region around the trace.

**Ellipse** – Display the uncertainties in ellipse form. Supported ONLY in Smith Chart or Polar display format.

**Settings**

**Coverage Factor** - Sets the level of confidence used in computing the specified measurement uncertainties. The higher the coverage factor, the higher the confidence that the computed uncertainty region includes the "true" measured data of the DUT.

To include **Noise** and Cable **Repeatability** uncertainties, the characterization must have already been performed. Learn how.

- **Noise** - Check to include characterized noise data contribution in the displayed uncertainties.
- **Repeatability** - Check to include cable and connector repeatability contribution in the displayed uncertainties.
- **Calibration** - Check to include Cal Standard uncertainty contribution in the displayed uncertainties.

**Apply to all traces** - Check to apply the above settings to ALL traces. Clear to use the 'Trace' selector to choose traces to which settings will be applied.

**Add Trace**

A trace is added to the channel. Choose from the following:

- **UMax** - The uncertainty maximum data (measured or memory PLUS upper limit uncertainty values).
- **UMin** - The uncertainty maximum data (measured or memory PLUS upper limit uncertainty values).

**Cancel** - Closes the dialog just like clicking **OK**. All changes are applied.

**Saving Uncertainty Data**

Once an Uncertainty Calibration has been performed and you are making measurements, you can then save Uncertainty data.

Before saving, an Uncertainty Calibration MUST be applied to the active measurement.

Click **File**, then **Save Data As**. (See the Save Data dialog).

Then choose from the following formats:

- **(*.u*p) S-parameter Uncertainty File** - Very similar to *.SnP files except this format includes uncertainty
Sample header:

!Coverage factor: \( k=1 \)

Freq(Hz), S1_1 mag (linear), S1_1 phase (deg), USl_1 mag (linear), USl_1 phase (deg)

- **(.dsd) S-parameter Data Standard Definition file** - Used by Dynamic Uncertainty (Opt S93015A/B) as the file type that is used to characterize standards.

- **(.sdatcv) METAS S-parameter Covariance File** - Used by Swiss Federal Institute of Metrology software for save/recall of measurement uncertainties.

- **(.unc) Uncertainty Parametric Data** - Contains the uncertainty contributions of the measured quantities (power and S-parameters).

Choose ports for Uncertainty Data dialog help

This dialog appears when any of the following conditions exist while attempting to save data to any of the Uncertainty data types:

- when requesting less data than is available.
- when requesting data for more than 4 ports.

**Number of ports** Select the number of ports for which data will be saved.
Notes

- When saving a *.dsd file, only data for 1-port or 2-ports per file are allowed. An error is returned when attempting to save data for more than 2 ports per file.

- When saving a *.up file, with Number of ports = 2, *.u2p data is saved; with Number of ports = 3, *.u3p data is saved, and so forth.

Arrow buttons  Click to Add or Remove ports from or to the following columns:

Available Ports  All test set ports are listed. There may NOT be valid data available for all of these ports. Learn more.

Chosen Ports  When OK is clicked, uncertainty data is saved for these ports.

OK  Becomes available when the number of Chosen ports = the Number of ports to save. Click to save the uncertainty file.
Power Calibration

Note: It is recommended that SmartCal be used instead of the following Power Calibration procedures. The procedures in this topic are for those who cannot change their existing test program/procedure with their installed-base PNAs.

Note: Source and Receiver Power Calibrations are NOT available in M937xA/P937xA PXI.

Source and Receiver Power Calibrations work together to provide very accurate power levels from the source, and very accurate power measurements from the VNA receivers.

- Source Power Calibration Overview
- Supported Power Meters and Sensors
- How to perform Source Power Calibration
- Setup
- Source Power Cal dialog
  - Source Power Calibration Options dialog
  - Power Meter Settings dialog
  - Power Loss Compensation dialog
  - Power Sensor Settings dialog (Zero / Calibrate)

- Copy a Source Power Calibration to other Channels
- Saving a Source Power Calibration
- Reducing Time to Complete a Source Power Calibration
- Receiver Power Calibration
- Saving Receiver Cals

Other Source Power Cal choices

- Guided Power Cal can be performed during an S-parameter Guided Calibration. Learn more.
- Receiver Leveling can be used to provide 'real-time' source power cal. Learn more.
See Also: Configure an Power Meter As a Receiver (PMAR)

See other Calibration Topics

Source Power Calibration Overview

Note: Source and Receiver Power Calibrations are NOT available in M937xA PXI.

Perform Source Power Calibration when you need accurate power levels at some point in the measurement path between the VNA test ports. For example, you need to characterize the gain of an amplifier across a frequency range at a specified input power. You would perform a source power cal at the input of the amplifier to ensure the exact power level into the amplifier across the frequency range.

Using a Source Power Cal, you can expect the power at the point of calibration to be within the range of the uncertainty of the power meter and sensor that is used.

Source Power Calibration...

- Is independent of measurement type. It corrects the VNA source regardless of which receivers are being used in a measurement. Therefore, it can be used with both ratio or non-ratio measurements.

- Applies ONLY to those measurements on the selected channel that use the test port that was specified as the Source for the calibration. For example, if you specify Channel 1 and Port 1 as the source to be calibrated, only those measurements on channel 1 that use port 1 as the source will be corrected.

- Can be used in conjunction with other measurement calibrations, such as a full 2-port calibration. For highest accuracy, perform the measurement calibration AFTER the source calibration.

- Can be used with Power Sweep type. Source Power Cal will correct the power at all power levels across the power sweep.

- Can be used with Port Power Uncoupled.

- Forces sweep mode to Stepped on measurements with source power correction turned ON.

- An external source can be calibrated using Source Power Cal.

Overview of How it works:

See Important First-time USB connection note.

Click to see the detailed procedure

1. Specify the measurement settings (frequency range, IFBW and so forth).

**Note:** When using an Keysight 848X power sensor (sensors that do NOT have built-in calibration factors), enter the Cal Factors using the *Power Sensor Settings* dialog, because the VNA instructs the power meter to NOT use the Cal Factor tables internal to the power meter.

3. Connect a power meter sensor to the point at which you want a known power level. This may be at the input or output of your device, or some other point between the test ports.

4. The VNA source is stepped through the specified frequency range, and power is measured with the power meter. At each data point, the source power is adjusted until the measured power is within your specified accuracy level.

5. When complete, the power meter is preset. The source power calibration can be saved as part of the instrument state.

6. The power meter is removed and the measurement path reconnected.

7. The calibration is automatically applied to the channel. All measurements on that channel using that source port benefit from the source power cal.

8. Perform an S-parameter calibration AFTER a Source Power Cal. The S-parameter cal is performed using the corrected stimulus power levels for the relevant ports.

**Verify** the source power calibration using the following procedure.

1. Connect the power meter as it was during the source power calibration.

2. Set the VNA to *Point Trigger* mode.

3. Trigger the VNA across the trace. Read about the behavior of the *sweep indicator*.

4. At each data point, the power meter should read the corrected power level within the specified tolerance.

**Supported Power Meters and Sensors**

See Keysight’s Power Meters and Sensors Webpage.

**USB Power Sensors**

- U848x Series USB Thermocouple Power Sensors (A.09.90.08 and later).
  - These include the following models: U8481A, U8485A, U8487A, U8488A, U8489A
  - External Calibration (connecting the sensor to the 1 mW ref port) is NOT supported.
• **IMPORTANT**: See http://na.support.keysight.com/pna/pseriesmeter.html

- U202x X-Series USB Peak and Average Power Sensors.
  - The VNA does NOT support peak mode in these sensors, but measures average power.


- U204x X-Series USB and LAN Power Sensors.

- U205xA/U206xA Series USB Power Sensors.

**USB Notes:**

- From a standard power cal (this topic), only one USB power sensor can be used to cover the entire frequency span. To use multiple power sensors, perform a Guided Power Cal. Learn how.

- To select a USB power sensor for a standard power cal:

  1. Connect the sensor directly to one of the VNA USB ports.
  2. From the Source Power Cal dialog, click **Power Meter Config**.
  3. On the Power Meter Settings dialog, select **USB**.

  - See Important First-time USB connection note.

  - See note about **Zeroing USB Power Sensors**.

  - See also: **Power Meters as Receivers (PMAR)**

**LAN Notes:**

- LAN power sensors can only be controlled via LAN.

- Typical LAN ports found on a PC or Keysight instrument are used for data transfer and communication only and will not power up a U2049XA LAN Power Sensor.

- LAN power sensors must connect to a PoE port (Power over Ethernet), which will supply DC power required to power up the sensor and to transfer data.
To select a LAN power sensor for a standard power cal:

1. Connect the sensor to a PoE/LAN connection.
2. From the Source Power Cal dialog, click Power Meter Config.
3. On the Power Meter Settings dialog, select LAN and enter the host name of the power sensor.

**Power Meters**

- P Series power meters (N1911A and N1912A) and all supported sensors.
- EPM Series power meters (N1913A and N1914A) and all supported sensors.
- EPM-P Series power meters (E4416A and E4417A) and all supported sensors.
- E Series power meters (E4418 and E4419) and all supported sensors.

**Power Meter Notes:**

- N1911A, 12A, 13A, and 14A power meters have a device-side USB connector and are controlled by the VNA exactly like a USB sensor. See USB Power Sensors (above). Although these meters may also have a front-panel USB port, USB power sensors must be connected directly to one of the VNA USB ports.

- Source Power Calibration operates slowly with the Keysight E930x and E932x power sensors.

- Some Keysight power meters have a mode that emulates the command set of the 437B or 438A power meter. The VNA does NOT support this emulation mode.

- The 82357A USB/GPIB Interface can be used to control power meters.

- Create a Custom Power Meter Driver for use with other power meters.

**Non-Keysight Power Sensors**

- Rohde and Schwarz NRP-Z power sensors (limited support). Learn how to install the drivers.

- VDI PM5 power meter (limited support). Learn how to install the driver and set up the power meter.

**How to perform Source Power Calibration**

**Note:** Guided Power Cal can be performed during an S-parameter Guided Calibration. Learn more.
1. Setup your measurement (sweep type, frequency range, IFBW, and so forth). By default, a Source Power Cal is performed on the source port of the active measurement.

2. Connect coax cable, GPIB cable, and power sensors to the VNA as shown in graphic below.

This image does NOT apply to USB power sensors, which are connected directly to a VNA USB port.

See Important First-time USB connection note.

3. Apply power to the power meter and allow 30 minutes warm-up time before beginning calibration.

4. Select Source Power Cal as follows:

Using Hardkey/SoftTab/Softkey

1. Press Cal > Main > Other Cals > Source Power Cal....

5. Complete the Source Power Cal dialog box (below), including Options, Loss Compensation and Power Sensor Settings, as needed.

   Note: When using an Keysight 848X power sensor (sensors that do NOT have built-in calibration factors), enter the Cal Factors using the Power Sensor Settings dialog, because the VNA instructs the power meter to NOT use the Cal Factor tables internal to the power meter.

6. When complete, click Take a Cal Sweep in the Source Power Cal dialog box.

7. Follow the prompts to connect the sensors as required.

8. At this time you can change the Source Port setting and perform a Source Power Cal on a different port.

9. When calibration is finished, click OK. Correction is then applied and turned ON for the relevant ports on
the active channel.

10. Remove sensor.

11. **SrcPwrCal** is displayed in the status bar when Source Power Correction is applied to the Active Measurement.

12. Perform a S-parameter calibration, which would use the corrected stimulus power levels for the relevant ports.

---

**To turn Source Power Correction OFF:**

- On the Calibration menu, then click Source Power Correction on/OFF.
- ONLY correction for the source port of the ACTIVE MEASUREMENT is turned OFF (regardless of port power coupling setting.)

---

**Interpolation or Extrapolation**

If the original stimulus settings are changed, Interpolation or EXTRAPOLATION is applied and **SrcPwrCal** is displayed in the status bar. This is different from measurement calibration interpolation. For example, if the frequency span is increased, the VNA will extrapolate new correction values rather than turn correction off. This is to protect your test device from being overpowered by the source. If the original settings are restored, then source power calibration returns to full correction.

---

**Source Power Cal** dialog box help
Note: Be sure that the frequency range of your power sensor covers the frequency range of your measurement. This does NOT occur automatically.

Cal with Frequency Extenders

If you are using a power sensor or power table to measure power over the frequency range of a frequency extender (mmWave module), the trace displays the measured power. This is done at the maximum RF input power setting for the frequency extender you are calibrating. This could be +10 dBm or +2 dBm depending on the power range you set during the source power calibration process. The default is +11 dBm to -30 dBm with a 0.5 dB step size. This can be changed as needed.

Though the displayed frequency range is the range of the RF input, it is actually calibrating over the frequency range of the frequency extender being used. For example, the display shows 12.5 GHz to 18.333 GHz (RF input) but it is being calibrated from 75 GHz to 110 GHz.

**Power**

**Cal Power** The calculated power (in dBm) at the calibration point. This value is the specified VNA source power plus the Power Offset value.

**Power Offset** Allows you to specify a gain or loss (in dB) to account for components you connect between the source and the reference plane of your measurement. These components will remain during a measurement. For example, specify 10 dB to account for a 10 dB amplifier in the path to your DUT. Following the calibration, the VNA power readouts are adjusted to this value.

To account for components that will be removed when the calibration is complete, use the Loss Compensation table.
Channel and Port Selection

**Channel**  Specifies the channel on which to perform the calibration. This setting defaults to the active channel.

**Source Port**  Specifies the source port to be corrected. This setting defaults to the source port for the active measurement.

**Note:** External sources can be calibrated using this dialog. Learn more.

Accuracy

At each data point, power is measured using the specified Power Meter **Settling Tolerance**, then adjusted until the reading is within this Accuracy **Tolerance** or the **Max Number of Readings** has been met. The last power reading is plotted on the screen against the Tolerance limit lines.

**Tolerance**  Sets the maximum desired deviation from the specified **Cal Power** level in 0.005 dB increments from 0 to 5 dB.

**Max Number of Readings**  Sets the maximum number of readings to take at each data point for iterating the source power. Enter a value between 1 and 1000.

Calibration Status

Allows you to turn Source Power Cal ON | OFF and view Cal data for each port, regardless of the active measurement. This feature allows the **Internal Second Source** to be calibrated and turned ON | OFF, even when being used as an incidental source in a measurement, such as an LO.

**Calibration ON**  Check to turn Source Power Calibration ON for the specified source port.

The displayed text indicates when interpolation is applied for the calibration.

Buttons

**Options**  Invokes the Source Power Cal Options dialog. Label to the left of the button displays the current 'Options' setting.

**Power Meter Config**  Invokes the Power Meter Settings dialog box

**Take Cal Sweep**  Begins source power calibration measurement.

**OK**  Applies calibration. This button is disabled until the Take Cal Sweep has been pressed.

**Cancel**  If a sweep is in progress, cancels the sweep. Press again to close the dialog.
Attention please: the power meter is operating in 200 r/s mode.

During a measurement, some Keysight power meters may display this message on the screen: It means that the meter is operating in 200 readings/sec which is the fastest speed setting for this meter. This is normal operation.

**Pass / Fail Limits**

Limit lines are drawn on the Source Power Cal measurement graticule area. These lines are at the Cal Power +/- the current setting of Accuracy Tolerance. A FAIL during the Source Power Cal sweep means that the VNA was unable to measure power to within the Accuracy Tolerance. Tight tolerances are more difficult to achieve at lower Cal Power levels. When a FAIL indication appears, increase the Max Number of Readings. If this does not cause a PASS condition, then decrease the Accuracy Tolerance value.

**See Also**

- Learn more about Source Power Cal
- Learn about External Testsets and Source Power Cal.

---

**Source Power Calibration Options** dialog box help

Provides options for measurement of the source power.

**Note:** At low power levels (less than -30 dBm) most power meters are not as accurate as a VNA receiver.
**Calibration Reference**  Choose power meter/VNA receiver to use to measure power.

- **Use a power meter.** Traditional source power calibration using only a power meter to measure the source power at each data point. Most accurate (at higher power levels) and slowest method.

  **Note:** Because the following two settings use VNA receivers to make power measurements, they do NOT work correctly when a Frequency Offset value is being used.

- **Use a power meter once, to calibrate receiver, then use receiver.** When checked, the first reading at each data point uses a power meter to calibrate the reference receiver. Subsequent readings, if necessary to meet your accuracy requirement, are measured using the reference receiver. This technique is much faster than using the power meter, and more accurate when measuring low power levels.

  **Note:** Do NOT use this setting if there is a component before the power sensor that exhibits non-linear behavior, such as a power amplifier in compression. Use a power meter and Calibrate the source at multiple power levels.

- **Use this receiver.** Select a VNA Receiver or a PMAR (Power Meter as Receiver).

  **VNA receiver** - For highest accuracy, first calibrate the receiver by performing a source power cal using a power meter, then a receiver cal. That receiver can then be used to quickly calibrate other VNA source ports, or used on another channel with different stimulus settings. This would be useful, for example, if the power level of the measurement was below the sensitivity of the power sensor. Calibrate the VNA receiver using a source power cal that is within the sensitivity of the sensor. Then, use the calibrated receiver to perform a second source power cal at the reduced power level.

  - The VNA receiver is specified using either standard receiver notation or logical receiver notation.

  - It is best to use the reference receiver for the source port to be calibrated. For example, if calibrating source port 2, specify "R2" or "a2" which is the same port 2 reference receiver using logical receiver notation.

  - To ensure an accurate source power cal, the frequency range over which the receiver was calibrated must be the same or larger than the "receiver only" source power calibration.

  - All accuracy and settling tolerance and number of reading settings apply just as they do with a power meter reading.

  **PMAR Device** - The power meter/sensor must first be configured. Learn how to Configure a PMAR device.
- **Use a power table and the analyzer reference receiver** Used to provide power leveling with mmWave test set and modules. Learn more.

**Calibrate the source at multiple power levels** Used primarily with mmWave measurements.

This feature can also be used with standard VNA measurements when a component is used in the source path such as a booster amp which does NOT have linear gain or loss over frequency. If this is not true for your setup but want to improve your source power accuracy, consider using the Receiver Leveling feature.

When checked, source power is measured using the specified 'Cal Reference' device (power meter/sensor or VNA receiver) and iterated on a sweep-to-sweep basis to construct a 2-dimensional power table: Power IN, Power OUT, over all frequencies.

- Click **Power Levels** to launch the Source Cal Power Levels dialog box to set the power levels at which source power is to be measured.

- The source power cal is saved, but the power table is NOT accessible.

**Calibrate the analyzer reference receiver** Check to calibrate the appropriate reference receiver to the power level that is measured at the calibration plane. Do this to make very accurate measurements using the calibrated reference receiver. This cal is done in addition to the standard source power cal using the any of the methods listed above. At the end of the source power cal measurement sweep, you can optionally save the reference receiver cal to a Cal Set to be recalled at a later time. The Cal is saved when the **OK** button is clicked to close the Source Power Cal dialog.

---

**Source Cal Power Levels** dialog box help

![Source Cal Power Levels dialog box](image)

This dialog appears when you click **Power Levels** on the Source Power Cal Options dialog.
Specify the power levels at which the Source Power will be calibrated. These values should be set to a few dB more or less than the measurement power levels.

**Max Power** - The highest power level at which to calibrate. This value should be a few dB higher than the highest power level of your measurement.

**Note:** Setting the **Max Power** will override power settings entered manually in a Power Table (InputPower). Therefore, when using a power table, set the **Max Power** value to the same value shown in the Power Table (InputPower).

**Min Power** - The lowest power level at which to calibrate. This value should be a few dB lower than the lowest power level of your measurement.

**Power Step** - Calibrate at every incremental power level, between the Max and Min Power settings.

---

**Power Meter Settings dialog box help**

This dialog appears when you click the **Power Meter Config** button on many dialog boxes.

**Communication**

- **GPIB / Address** Select GPIB power meter. Then select the address for the power meter. Default is 13. The VNA will search VISA interfaces that are configured in the Keysight IO Libraries on the VNA. **Note:** Use this selection when using a 82357A USB/GPIB Interface.
**USB** VNA scans for USB power sensors or N191x device-side USB power meters. Select a power sensor from the list. Only ONE USB power sensor can be configured to cover the entire frequency range of the calibration. To use multiple power sensors, perform a Guided Power Cal.

**LAN** Specify the Hostname or IP address of the Power Meter.

**Any** This can be used if you wish to spell out the exact VISA resource string/alias for your power meter’s I/O connection, for GPIB, USB, LAN or any other I/O protocol supported by VISA. For example, you must use this to use a VDI PM5 power meter, and the VISA resource string/alias in that case must be `ASRL3::INSTR`.

**Note:** The VDI PM5 driver software must be installed to use VDI power sensors. Refer to Install VDI PM5 Driver.

**Sensors** Invokes the power sensor settings dialog box.

**Settling**

These Settling settings do not apply when a VNA receiver is the power measurement device. Each power meter reading is "settled" when either:

- two consecutive meter readings are within this **Tolerance** value or
- when the **Max Number of Readings** has been met.

The readings that were taken are averaged together to become the "settled" reading. The settled reading is then compared to the Accuracy Tolerance requirements (tolerance and max readings) specified on the Source Power Cal dialog box.

**Tolerance** When consecutive power meter readings are within this value of each other, then the reading is considered settled.

**Max Number of Readings** Sets the maximum number of readings the power meter will take to achieve settling.

**Sensor Loss Compensation**

**Use Loss Table** Select this checkbox to apply loss data to Source Power calibration correction (such as for an adapter on the power sensor).

**Edit Table** Invokes the Power Loss Compensation dialog box.
To Add a Row to the table, click on a row in the table and press the down arrow on either the VNA front panel or keyboard.

To Edit a value, double-click in the cell to be edited.

Compensates for losses that occur when using an adapter or coupler to connect the power sensor to the measurement port. These components will be removed when the calibration is complete. To account for components that will remain during the measurement, use the Power Offset setting.

The Frequency / Loss pairs define the amount of loss for the entire frequency range. For example, using the entries in the above dialog image:

- 0.5 dB is used to compensate power sensor measurements up to 1 GHz.
- Each data point between 1 GHz to 2 GHz is linearly interpolated between 0.5 dB and 1 dB.
- 1 dB is used above 2 GHz.
- A single frequency/loss segment is applied to the entire frequency range.

Beginning with A.09.80, enter up to 9999 segments to achieve greater accuracy. Previously the limit was 100.

Note: Large segment counts with one or more power sensors can result in long load and close times for the VNA Application.

**Frequency**  Enter a frequency in Hz.

**Loss**  Enter a loss as a POSITIVE value in dB. To compensate for gain, use NEGATIVE values.

**Delete Table Segment**  Deletes row indicated in the field.
**Delete All** Deletes all data in the table.

The Power Loss Compensation table survives VNA Preset and Power OFF. To NOT use Loss compensation, clear the Use Loss table checkbox on the Power Meter Settings dialog.

---

**Power Sensor Settings** dialog box help

This dialog appears when you click the **Sensors** button on the Power Meter Settings dialog.

**Note:** Be sure that the frequency range of your power sensor covers the frequency range of your measurement. This does NOT occur automatically.

**Sensor A (B)** Displays one of the following messages depending on type of sensor.

- **Not connected** The VNA is not detecting a power sensor.

- **Cal factors are contained within this sensor** This message is displayed when the Internal Reference Cal Factor and Cal Factor data are contained in the sensor and automatically accessed.

**Sensor Data** Allows the following entries for power sensor data:

- **Reference Cal Factor** Specifies the sensor's Reference Cal Factor.

- **Cal Factor Table** Specifies the frequency and corresponding Cal Factor for the sensor.

- **Delete Cal Factor** Deletes the indicated row in the table.

- **Delete All** Deletes all data in the table.
• **To Add a Row** to the table, click on a row in the table and press the down arrow on either the VNA front panel or keyboard. A row is added to the bottom of the table. The table is automatically sorted by frequency when OK is pressed.

**Load Cal Factors** Click to load cal factors from a *.csv file that you create from the cal factors that appear on the sensor. The first line of the file MUST have the reference Cal Factor (typically 100), followed by Freq / Cal Factor pairs as shown in the following image:

![Cal Factor Table](image)

**Save Cal Factors** Click to save the cal factor table to a *.csv file.

**Use this sensor only** Check this box to use this sensor over the entire frequency span of the measurement, even if two sensors are connected to power meter.

Clear this box to allow entry of minimum and maximum frequencies for the sensor. Only ONE of the two sensors can have this box checked. You will be prompted to connect the appropriate sensor during the power calibration.

**Minimum Frequency** Specifies the minimum frequency range for the sensor when using dual sensors.

**Maximum Frequency** Specifies the maximum frequency range for the sensor when using dual sensors.

**Zero and Calibrate the Power Sensor**

For highest accuracy, Zero AND Calibrate the power sensor before measuring data. Follow prompts that may appear.

**Zero** - If the following settings are 'greyed', Internal or External zeroing is selected automatically based on the power meter/sensor model. Otherwise, select the appropriate type of zeroing to perform, then press Zero.

- **Internal Zero** - A switch inside the power sensor removes the sensor from the incident power.

- **External Zero** - Requires that you physically remove the sensor from incident power.
### Note: For the U2000 Series USB power sensors

Calibration is NOT available. Select External Zero ONLY when the power to be measured is *below* the specified level. Otherwise, the U2000 series performs internal zeroing automatically when needed. See your power sensor documentation for more details.

- U200xA - below -30 dBm
- U200xH - below -20 dBm
- U200xB - below 0 dBm

If your U2000 power sensor 'hangs' when external zeroing, upgrade the power sensor firmware to Rev. A.01.02.00 or higher to fix this problem.

### Note: For the U2020 X-Series USB power sensors

The U2020 X-Series support only internal zeroing. But like the U2000 series, they default to performing zeroing automatically when needed.

**Calibrate** - Available when the selected sensor has calibration capability. Calibration involves measuring an internal 1 mW source.

- Keysight P-Series sensors and U2020 X-Series USB sensors have an internal reference so you can calibrate them without connecting to a meter’s reference port.
- Keysight U2000 USB power sensors do not require calibrating.
- For other sensors, refer to the documentation to determine if it has calibration capability.

Press **Calibrate**, then follow the prompts.

### Copy a Source Power Calibration to other Channels

A macro application is now available that copies a Source Power Calibration to other channels. Once downloaded and installed on a VNA, the macro is automatically configured up. To learn more, click **Help** on the application main dialog. Get the application from [http://na.support.keysight.com/pna/apps/applications.htm](http://na.support.keysight.com/pna/apps/applications.htm).
Saving a Source Power Calibration

Because Source Power Cal calibrates source hardware, the calibration data is saved as part of the Instrument State, in either a .sta file or a .cst file. This correction is applied to all measurements on the channel that uses the calibrated source. See Save Instrument State.

Reducing Time to Complete a Source Power Calibration

The time required to perform a Source Power Calibration depends on source power, number of points, and number of readings taken. You can reduce this measurement time with the following methods:

- **Reduce number of points before calibration.** You can reduce the number of points before the measurement, then return the number of points to its original value after calibration is complete and correction is ON. The analyzer will perform a linear interpolation, although with some loss in accuracy.

- **Use an Keysight E-Series sensor.** You can obtain 40+ readings per second over GPIB with this type of sensor on the VNA.

- **Increase power to the sensor.** Lower power may have longer settling time with some sensors.

- **Check Use Reference Receiver for Iteration.**

Receiver Power Calibration

**Note:** Source and Receiver Power Calibrations are NOT available in M937xA PXI.

**Note:** A Guided Power Cal can be performed during an S-parameter Guided Calibration. [Learn more.](#)

Receiver power calibration mathematically removes frequency response errors in the specified VNA receiver, and adjusts readings to the same, or a value offset from, the source power calibration level. It is the same as doing a Response Cal or Data / Memory, (Normalization) but with the data shifted to the Cal Power value.

Use Receiver Power Calibration to make very accurate absolute power (amplitude) measurements.

**Receiver Power Calibration:**

- Is ONLY allowed when making absolute power (unratioed) measurements.
- Is most accurate when a source power calibration was performed first.
- Applies to all unratioed measurements in the active channel using that receiver.
- Can be saved in a Cal Set and later reapplied to a like measurement.
Interpolation

Like other calibration types, if the original stimulus settings are narrowed, interpolation is applied and \textbf{C* Rcvr Pwr} is displayed in the status bar. If the original stimulus settings are made wider, the VNA will turn Receiver Power Correction \textbf{OFF}.

If the original settings are restored, then receiver power calibration returns to full correction.

\textbf{How to perform a Receiver Power Calibration}

1. Perform a \textbf{Source Power Calibration}.
2. Set the active measurement to unratioed. \textbf{Learn How}.
3. Connect a THRU line from the source port to the receiver port.

   - When performing a receiver power cal on a reference receiver (source 1 and receiver R1), no connection is necessary as the receiver is internally connected to the source.
   - When the source port and receiver port are the same (receiver A, source port 1), then connect an open or short to get maximum power to the receiver. This practice is not recommended. It is best to use different ports for the source and receiver.

4. Ensure correction for Source Power Calibration is ON as indicated by \textbf{Src Pwr Cal} or \textbf{Src Pwr Cal*} in the status bar.
5. Start the \textbf{Calibration Wizard}

\textbf{Using Hardkey/SoftTab/Softkey}

1. Set the active measurement to unratioed. \textbf{Learn How}.
2. Press \textbf{Cal > Main > Other Cals > Receiver Power Cal}....
Cal Type Selection  Select Receiver Power

Receiver Power Configuration

**Cal Power**  Specifies the power level to be displayed on the measurement when complete. (Source Port Power + Power Offset).

**Source Port Power**  Test port Power set for the measurement. [Learn how to change Test Port Power](#)

**Power Offset**  Allows you to specify a gain or loss (in dB) to account for components you connect between the source and the reference plane of your measurement AFTER a source power cal has been performed. Following the calibration, the VNA power readouts are adjusted to the Cal Power value.

**Next**  Click to continue the Calibration Wizard.

**Notes:**

- When Receiver Power Cal is finished, 'Response' is displayed in the status bar and correction data is applied to subsequent sweeps. This is done because Receiver Power Cals are essentially Response Cals once they are stored and applied. See Saving a Receiver Power Cal below.

- To turn correction OFF, click Cal > Main > Correction > Channel Correction OFF.

Learn more about Receiver Power Cal (scroll up).

**Saving a Receiver Power Calibration**

Beginning with VNA Revision 5.0, Receiver Power Cal is saved to a Cal Register and optionally to a User Cal Set. It can be applied to measurements in the same way as other Cal Types. Previously, Receiver Power Cal data was saved as part of an Instrument State and was only applied to the measurement on which it was performed.

Learn more about Saving VNA files types.
Fixture Simulator

This features allow you to mathematically add (embed) or remove (de-embed) circuits to, or from, your measurements. The mathematical models are applied to specific ports for all measurements on the channel.

New fixture simulator provides more flexible and intuitive GUI for multi-port measurements. The multiple fixturing elements can be combined in any order, creating infinite combinations.

Notes

- The following features are available in GCA, GCX, Swept IMD, Swept IMDX, Noise Figure, NFX, Diff IQ, and Spectrum Analyzer Apps:
  - Port Extensions (Not available in Swept IMD, IMDX, Noise Figure, NFX, or Diff IQ)
  - SnP file Embedding/De-embedding
  - Port Matching
  - Port Z Conversion
  - Power Compensation

- All other Fixturing features are available ONLY in a standard channel.

- SNP files can be used in many fixture features, including SNP file deembedding, ground loop embedding/de-embedding, port matching, and differential port matching. The contents of any SNP files used for fixturing are read and stored in memory when the file is selected. After selection, the file is not locked and contents may be changed. In order to pick up any updated content, it is necessary to re-apply the file through these means:
  - GUI: Re-select the file in the dialog.
  - SCPI: Re-select the file and apply updates:
    ```scpi
    calc:fsim:draft:circ1:file "myFile.s2p"
    calc:fsim:apply
    ```
    - Legacy SCPI: Cycle the deembedding state for the feature using the file. For example:
See Also

- **Procedures: To Embed or De-embed?**
- "De-embedding and Embedding S-Parameter Networks Using a Vector Network Analyzer" App note for more conceptual information on Fixture Simulation.
- See an example of how these functions can be used to de-embed unwanted effects of a test fixture, and then mathematically embed the DUT in the circuit in which it is used.

Notes

- The fixturing operations are applied to the measurement results.
- In the Data processing chain, the Fixture Simulator functions occur at the same time as the **Apply Error Terms** block.
- When fixturing is enabled, all of the enabled fixturing features are applied when snp files are saved.
- **Source power compensation** is then optionally applied to compensate for the aggregate loss through all enabled fixturing operations.

How to select Fixturing Simulator

**About Fixturing ON/off**

**BOTH** of the following must occur to turn a fixturing selection **ON**.

**EITHER ONE** will turn a fixturing selection **OFF**.

1. Turn **Apply Fixtures ON/off**
   - Port Extensions is NOT affected by Fixturing ON/off.

2. Check **Enable** on the individual fixturing selection dialog box.

**Using** **Hardkey/SoftTab/Softkey**

1. Press **Cal > Fixtures > Apply Fixtures**.
**Fixture Generator** dialog box help

You can configure your virtual fixtures on GUI. The calculation order is from right to left.

**Note:** The calculation order was from left to right for A.14.70.01 to A.15.20.05.

---

**File:**

**Save Topology...** : Save the current setup into a topology setting file (.topo). You can recall it by Load Topology.

**Save Topology as SnP...** : Save the whole embedding, de-embedding, and port extension blocks as a SNP file. The portion of Port Z conversion is not included. The dimension N of the SNP file will be the total number of VNA's test ports. This saved SNP file can be used later in the Fixture Generator as a de-embedding block.

**Load Topology...** : Load the topology setting file (.topo). You can load the .totp file which was saved with the VNA of equal or less number of test ports.

**Load Default Topology** : Reset the topology setting.
**Edit:** Allow to enable/edit/cut/copy/paste/delete/add for a block.

**Active Channel:** Select the channel to apply the fixture simulator.

**Apply Fixtures:** Apply all fixture blocks and all impedance transforms that are enabled. It does not affect port extensions or balanced definitions.

**Add Block...** Opens the **Add Block** dialog to add and define a new block. See the [block types](#) below.

Right Click on each block shows a pop-up menu.

- **System Z0...**
- Port Z Conversion...
- Balanced Z Conversion...
- SnP file
- Ground Loop
- Port Matching
- Differential Matching
- Ideal Line
- Impedance Transformer

**Disable (Draft) State**

**Enable (Active) State**
Blue: Embedded, Green: De-embedded

SnP Embedding/De-embedding dialog box help
This function specifies a SnP file to embed (add) or de-embed (remove) from the measurement results. Computation takes place BEFORE Balanced conversion.

The SnP file S-Parameter data is normalized to a single File-Zo impedance as defined in the file. The VNA will re-normalize the S-Parameter data from File-Zo to the VNA System-Zo. The VNA will interpolate if the number of data points that are read is different from the current VNA setting.

**Note:** De-embedding a component with more than 20 dB of loss becomes impractical because of an inability to accurately measure the match of the DUT through such a device.

The de-embedding operation recalls an .s2p file (Touchstone format) which includes the electrical characteristics of a 2-port fixture or device. The file can be in any standard format (real-imaginary, magnitude-angle, dB-angle).

**Enable block as** Check to apply the settings to the measurement results. Must also enable *Fixturing ON/off*.

**De-Embed/Embed** Select the embed (adding the snp device virtually) or de-embedding (removing the snp device).

**Note:** Port Matching circuits can only be "embed".

**File Name:** Select the snp file for embeding or de-embeding

**Modify:**
If the block has high loss, then it may cause transmission measurement errors when full S-parameter correction is applied. The reason is that the DUT output match cannot be measured accurately through a high-loss path; if the fixture has a large mismatch then this error will cause errors in the transmission measurement. Zeroing the fixture reflection at the DUT will reduce this error. This problem is described in "Handbook of Microwave Component Measurements with Advanced VNA Techniques", Joel Dunsmore, John Wiley & Sons, page 592 in the First Edition and page 775 in the Second Edition.

- **None**: does not modify the block.
- **Set Snn=0 @ DUT** - will set all reflection parameters on the DUT-side to zero.
  
  - For the 4-terminal fixture shown above, it will set S33=s44=0.
- This is useful for fixtures that have low crosstalk between DUT-side ports.

- Set \( S_{nn} = X_{\text{Talk}} = 0 \) @ DUT - will set all reflection and crosstalk parameters on the DUT-side to zero.

- This selection is not available for a 2-terminal fixture.

- For the 4-terminal fixture shown above, it will set \( S_{33} = S_{44} = S_{43} = S_{34} = 0 \).

- This is useful for multiport fixtures which have a lot of crosstalk between the DUT-side ports.

**Enable Extrapolation** Check to apply a simple extrapolation when the SnP file has a narrower frequency range than the channel. The values for the first and last data points are extended in either direction to cover the frequency range of the measurement. The frequency ranges of the SnP file are displayed at the right of the dialog.

When extrapolation is necessary and enabled, a message is displayed showing the frequency range to be extrapolated. When extrapolation is necessary and disabled, a message is displayed offering to enable extrapolation.

**Note:** For IMD and IMDX channels, you will be prompted for extrapolation as described above if the S2P file does not meet the port frequency conditions. However, if the file is valid, extrapolation will be enabled automatically. This is for compatibility reasons with the IMD/IMDX calsets, which list all frequencies for all ports. This applies only for the SnP section, NOT the ground loop section. Ground loop is not for IMD/IMDX.

**General** (Right side of Dialog box)

**Block ID** Block ID. This is the same number as the circuit number in SCPI.

**F\text{max}** show the maximum frequency of imported snp file

**F\text{min}** show the minimum frequency of imported snp file.

**Number of Port** show the number of port of imported snp file

**Reverse Ports** Reverses the ports on an existing S2P file.

- The data for S11 becomes the data for S22 and vice versa.
- The data for S21 becomes the data for S12 and vice versa.
Ground loop de-embedding removes the effect of a non-ideal ground connection between the DUT’s ground and the analyzer’s ground reference. Typically, the non-ideal component is the parasitic inductance of the ground contacts.

Ground loop embedding adds the effect of a non-ideal component on the ground contacts. The Ground Loop De-embedding / Embedding can be specified by circuit model type or touchstone file.

**Enable block as** Check to apply the settings to the measurement results. Must also enable **Fixturing ON/off**.

**De-Embed/Embed** Select the embed (adding the snp device virtually) or de-embedding (removing the snp device)
Ground Loop (RL) or (GC)

**Circuit** (Right side of Dialog box)

- **Inductance (L), Resistance (R) / Conductance (G), Capacitance (C)** Values for the specific components of the circuit type that models your fixture.

**General** (Right side of Dialog box)

- **Block ID** Block ID. This is the same number as the circuit number in SCPI.
- **Fmax** Fixed at 9.9999 THz
- **Fmin** Fixed at 0 Hz
- **Number of Ports** Define the number of ports. Once the number of ports is set, it cannot be changed.
- **VNA Fmax** Maximum frequency of the VNA.
- **VNA Fmin** Minimum frequency of the VNA.

**Ground Loop (file)**

**Enable Extrapolation** Check to apply a simple extrapolation when the SnP file has a narrower frequency range than the channel. The values for the first and last data points are extended in either direction to cover the frequency range of the measurement. The frequency ranges of the SnP file are displayed at the right of the dialog.

When extrapolation is necessary and enabled, a message is displayed showing the frequency range to be extrapolated. When extrapolation is necessary and disabled, a message is displayed offering to enable extrapolation.

**General** (Right side of Dialog box)

- **Block ID** Block ID. This is the same number as the circuit number in SCPI.
- **Fmax** show the maximum frequency of imported snp file
- **Fmin** show the minimum frequency of imported snp file.
- **Number of Ports** Define the number of ports. Once the number of ports is set, it cannot be changed.
**Reverse Ports** Reverses the ports on an existing S2P file.

- The data for S11 becomes the data for S22 and vice versa.
- The data for S21 becomes the data for S12 and vice versa.

**Port Matching** dialog box help

This function specifies a circuit to embed (add) to the measurement results. See Order of Fixture Operations.

Enable block as  Check to apply the settings to the measurement results. Must also enable Fixturing ON/off.

De-Embed/Embed  Select the embed (adding the snp device virtually).

**Type:** Choose a circuit model that best emulates your fixture at the selected VNA port: The circuit diagram is shown in the dialog box.

**Circuit** (Right side of Dialog box)

**Capacitance (C), Inductance(L), Resistance(R), Conductance(G)**  Values for the specific components of the circuit type that models your fixture.
**General** (Right side of Dialog box)

- **Block ID**  Block ID. This is the same number as the circuit number in SCPI.
- **Fmax**  Fixed at 9.9999 THz
- **Fmin**  Fixed at 0 Hz
- **Number of Ports**  Fixed at 2.
- **VNA Fmax**  Maximum frequency of the VNA.
- **VNA Fmin**  Minimum frequency of the VNA.

**Differential Port Matching** dialog box help

This function allows the embedding/de-embedding of a differential matching circuit at a balanced port.

- **Enable block as**  Check to apply the settings to the measurement results. Must also enable Fixturing ON/off.
- **De-Embed/Embed**  Select the embed (adding the snp device virtually) or de-embedding (removing the snp device)

- **Differential Matching**  Predefined circuit.

![Differential Matching Circuit Diagram](image)

- **Circuit**  (Right side of Dialog box)  Define the values
  - **C**  Capacitance value
  - **G**  Conductance value
  - **L**  Inductance value
  - **R**  Resistance value
**General** (Right side of Dialog box)

**Block ID**  Block ID. This is the same number as the circuit number in SCPI.

**Fmax** Fixed at 9.9999 THz

**Fmin** Fixed at 0 Hz

**Number of Port** Fixed at 2

**VNA Fmax** Maximum frequency of the VNA.

**VNA Fmin** Minimum frequency of the VNA.

- **Diff. Matching (s2p)**  Select an *.S2P file that represents the matching circuit. Then click **Browse** to navigate to the *.S2P file.

  **Note:** For the *.S2P file:
  Port 1 of the circuit is assumed to be connected to the VNA
  Port 2 of the circuit is assumed to be connected to the DUT.

**Block ID**  Block ID. This is the same number as the circuit number in SCPI.

**Fmax** show the maximum frequency of imported snp file

**Fmin** show the minimum frequency of imported snp file.

**Number of Ports** Fixed at 2

**VNA Fmax** Maximum frequency of the VNA.

**VNA Fmin** Minimum frequency of the VNA.

---

**Ideal Line** dialog box help
This function allows the embedding/de-embedding of an ideal line.

**Enable block as** Check to apply the settings to the measurement results. Must also enable **Fixturing ON/off**.

**De-Embed/Embed** Select the embed (adding the snp device virtually) or de-embedding (removing the snp device)

- **Electrical Parameter**
  - **Delay** Delay in second
  - **Line Z0** Line Characteristic Impedance
  - **LineZ0_enable** Enable/Disable Line Z0
  - **Loss** Line Loss

- **Diff. Matching (.s2p)** Select an *.S2P file that represents the matching circuit. Then click **Browse** to navigate to the *.S2P file.
  - **Length** Length in meter
  - **Line Z0** Line Characteristic Impedance
- **LineZO_enable** Enable/Disable Line Z0
- **Loss** Line Loss
- **Vel_Fact** Velocity Factor

**General** (Right side of Dialog box)

**Block ID** Block ID. This is the same number as the circuit number in SCPI.

**Fmax** Fixed at 9.9999 THz

**Fmin** Fixed at 0 Hz

**Number of Port** Fixed at 2

**VNA Fmax** Maximum frequency of the VNA.

**VNA Fmin** Minimum frequency of the VNA.

---

**Impedance Transformer** dialog box help

This function allows the embedding/de-embedding of an impedance transformer.
Enable block as  Check to apply the settings to the measurement results. Must also enable Fixturing ON/off.

De-Embed/Embed  Select the embed (adding the snp device virtually) or de-embedding (removing the snp device)

- Circuit
  - Rin, Rout  Impedance for in and out

General  (Right side of Dialog box)

  Block ID  Block ID. This is the same number as the circuit number in SCPI.

  Fmax  Fixed at 9.9999 THz

  Fmin  Fixed at 0 Hz

  Number of Ports  Fixed at 2

  VNA Fmax  Maximum frequency of the VNA.

  VNA Fmin  Minimum frequency of the VNA.

---

**Power Compensation** dialog help

![Power Compensation Dialog](image)

**Note:** This feature is available in ALL measurement classes.
This function adjusts the source power at the specified port to compensate for the combined amount of gain or loss through specific fixturing operations. Use this function to set the power level at the DUT input.

Power Compensation adjusts the source power for the gain/loss through 2-port de-embedded fixture components.

- It does not compensate for any port matching networks.
  - To work around this limitation, compute the anti-network of the matching circuit and use that data to generate an S2P file. Change the fixture type from port matching to De-embed. This should result in the same measurement data but will also enable the port power compensation feature.
- It does not compensate for any fixture networks with more than 2 ports.
  - There is no work around for this limitation. There is a coarse adjustment for port power using Power Offsets. Calculate the loss through the fixture, use that number to set the power offset, and then set the desired power level at the port.

For example:

- Your DUT requires a fixture on the input port which is connected to VNA port 1.
- The fixture description (such as an S2P file at the De-embed function) indicates the fixture has approximately 2 dB of loss across the frequency span.
- You set source power to 0 dBm. But you want 0 dBm at the DUT input (the fixture output).
- Check Power Compensation on Port 1 and enable Fixturing.
- Power Compensation causes the source power to be increased by approximately 2 dB so that the power at the fixture output plane will remain at 0 dBm.

Power Compensation affects all measurements in the channel.

Enable Fixturing to use Power Compensation.

**Note:** Use caution when applying power compensation. Always test your setup without a DUT in a place. If you are using S2P files, Recall your S2P file into the VNA so you can verify that the device your S2P file describes is what you intended it to be. It is too easy to misalign data in S2P files if they are constructed by hand.

**Fixture Simulator Example**

The following example shows a DUT and the matching circuit with which the DUT will be used in its intended application. When the DUT is tested in a high-volume manufacturing environment, multiple
test fixtures are often required. The most accurate way to test the DUT and ensure measurement consistency between the different test fixtures is to use a simple, repeatable, test fixture without the actual matching elements.

To get the desired performance data, the parasitic effects of the fixture must first be removed (de-embedded) from the measured data. Then a perfect "virtual" matching circuit must be simulated and added mathematically (embedded) to the corrected, measured data. The result is an accurate display of the DUT as though it was actually tested with a physical matching circuit, but without the uncertainties of using real components.

Test Device and the circuit in which it will be used.

This diagram does NOT refer to the order in which operations are performed.

1. Create a balanced measurement using single-ended to balanced (SE-Bal) topology. Include all relevant measurement settings (IFBW, number of points, and so forth). Once the measurement is created and calibrated, the measurement parameter can be easily changed. For example, Sdd22 to Sds21.

2. Calibrate the measurement at the point where the simple test fixture is connected to the VNA. Use accurate calibration standards and definitions.
3. Remove the effects of the three uncalibrated transmission lines of the simple test fixture. This can be done in several different methods. The easiest is to use manual or automatic Port Extensions to move the calibration reference plane to the DUT. This removes the electrical length and loss of the fixture’s transmission lines, but does not account for fixture mismatch. Another method is to de-embed previously-created *.S2p files of the 3 transmission lines. The files can be created using external ADS modeling software. Another alternative is to create the *.S2P files by independently measuring all 3 ports of the test fixture and saving the results of each to an S2P file.

4. With the test fixture connected to the VNA and a DUT inserted, the measurement results now appear as though calibration was performed at the connections to the DUT, and the device was measured in a 50-ohm single-ended test environment. The following steps will cause the results to reflect the performance of the device as though the device is embedded in the circuit in which it will be used.

5. Port 1 of the device is a single-ended port and sees a source impedance the same as the VNA system impedance, so no change is required. However, if Rs were a value other than 50 ohms, Port 1 Impedance Conversion would be used to simulate the different impedance.

6. Port Matching is used to simulate L1 inductance. Select any of the Shunt L circuits to embed (add) to the measurement results. Enter the value of L and R. The C and G values can be entered as 0 (zero).

7. Port Matching is used to simulate C1 and C2 capacitance. For both port 2 and port 3, select any of the Series C circuits to embed (add) to the measurement results. Enter the value of C and G. The L and R values can be entered as 0 (zero).

8. Balanced Conversion mathematically simulates the measurement in balanced mode.

9. Differential Port Matching is used to simulate L2 inductance. Select Shunt L- Shunt C and enter the inductance / resistance value. The C and G values can be entered as 0 (zero).

10. Finally, Differential Z Conversion is used to simulate a circuit termination of 200 ohms. If you are making Common Mode measurements, specify Common Mode Z Conversion.
The following features allow you to mathematically add (embed) or remove (de-embed) circuits to, or from, your measurements. The mathematical models are applied to specific ports for all measurements on the channel.

### Notes

- The following features are available in GCA, GCX, Swept IMD, Swept IMDX, Noise Figure, NFX, Diff IQ, and Spectrum Analyzer Apps:
  - Port Extensions (Not available in Swept IMD, IMDX, Noise Figure, NFX, or Diff IQ)
  - 2 Port De-embedding
  - Port Matching
  - Port Z Conversion
  - Power Compensation
- All other Fixturing features are available ONLY in a standard channel.

### See Also

- **Procedures:** To Embed or De-embed?
  - "De-embedding and Embedding S-Parameter Networks Using a Vector Network Analyzer" App note for more conceptual information on Fixture Simulation.
  - See an example of how these functions can be used to de-embed unwanted effects of a test fixture, and then mathematically embed the DUT in the circuit in which it is used.

### Order of Fixture Operations

Click to learn more about each operation.

First, the following **Single-ended** measurement functions are processed in this order:

1. Port Extensions
2. 2-Port De-embedding
3. Ground loop de-embedding / embedding
4. Port Matching Circuit Embedding
5. Port Z (Impedance) Conversion
6. 4-Port Network (single-ended) Embed/De-embed

Note: The operation for ground loop embedding and ground loop de-embedding will always occur as the 3rd step. It cannot be moved. By default, this is after the 2-Port DeEmbedding operation.

Then, **Balanced** measurement functions are processed in this order:

7. Balanced Conversion
8. Differential / Common Mode Port Z Conversion
9. Differential Port Matching

- **Source power compensation** is then optionally applied to compensate for the aggregate loss through all enabled fixturing operations.

Notes

- The fixturing operations are applied to the measurement results.
- The order of operations 1 through 5 can be changed using the SCPI command: `CALC:FSIM:SEND:OORD`. Learn how to send this command from the GPIB Command Processor Console.
- The order of the operations 6 through 9 can NOT be changed.
- In the Data processing chain, the Fixture Simulator functions occur at the same time as the **Apply Error Terms** block.
- When fixturing is enabled, all of the enabled fixturing features are applied when snp files are saved.
How to select Fixturing Simulator

About Fixturing ON/off

BOTH of the following must occur to turn a fixturing selection ON.

EITHER ONE will turn a fixturing selection OFF.

1. Turn Apply Fixtures ON/off
   Port Extensions is NOT affected by Fixturing ON/off.

2. Check Enable on the individual fixturing selection dialog box.

Using Hardkey/SoftTab/Softkey

1. Press Cal > Fixtures > Apply Fixtures.

Port Matching dialog box help

Note: This feature is available in the following measurement classes: GCA, GCX, Swept IMD, Swept
IMDx, Noise Figure, NFX, Diff IQ, Spectrum Analyzer, and standard (S-Parameter) channels.

This function specifies a circuit to embed (add) to the measurement results. See Order of Fixture Operations.

**Enable Port Matching** Check to apply the settings to the measurement results. Must also enable Fixturing ON/off.

**Port** - Select Port in which to apply simulation.

**Circuit Model for Matching** - Choose one of the following that best emulates your fixture at the selected VNA port:
User Defined (S2P File)  Load a file that is specified with User S2P File button.
None  Use no circuit model.

User S2P File  Click to specify an S2P file of the circuit model to embed at the selected port. If the normalized impedance value in a recalled User .S2P file is different from the port reference impedance setting of the VNA, the VNA setting is used.

**Circuit Values**

**Capacitance (C), Inductance(L), Resistance(R), Conductance(G)**  Values for the specific components of the circuit type that models your fixture.

**Reset**  Restores the default values.

**2 Port De-embedding** dialog box help

**Note:** This feature is available in the following measurement classes: GCA, GCX, Swept IMD, Swept IMDx, Noise Figure, NFX, Diff IQ, Spectrum Analyzer, and standard (S-Parameter) channels.
De-Embed when you have performed a calibration and then added a fixture (an adapter, an attenuator, a longer cable, etc.) that connects between the Cal reference plane and your DUT. This function removes the effects of a component or test fixture from the measurement results.

**Note:** De-embedding a component with more than 20 dB of loss becomes impractical because of an inability to accurately measure the match of the DUT through such a device.

The de-embedding operation recalls an .s2p file (Touchstone format) which includes the electrical characteristics of a 2-port fixture or device. The file can be in any standard format (real-imaginary, magnitude-angle, dB-angle).

**Enable De-embedding** Check to apply the settings to the measurement results. Must also enable Fixturing ON/off.

**Enable Extrapolation** Check to apply a simple extrapolation when the S2P file has a narrower frequency range than the channel. The values for the first and last data points are extended in either direction to cover the frequency range of the measurement. The frequency ranges of both the channel and the S2P file are displayed at the bottom of the dialog.

When extrapolation is necessary and enabled, a message is displayed showing the frequency range to be extrapolated. When extrapolation is necessary and disabled, a message is displayed offering to enable extrapolation.

This setting also causes 4-port Extrapolation to be enabled and disabled.

**Port** The VNA port to which the recalled de-embedding file is applied.
From the drop-down menu, select **User Defined (S2P File)**.

**Reverse Adaptor Ports**  Check to cause the Fixture/Adapter to be configured with Port 2 connected to the VNA and Port 1 to be connected to the DUT. The image in the dialog reflects that change.

**User S2P File**  Click to specify an existing .S2P file. If the normalized impedance value in a recalled User .S2P file is different from the port reference impedance setting of the VNA, the VNA setting is used.

---

**Port Z (Impedance) Conversion dialog box help**

![Port Z Conversion dialog box](image)

**Note:** This feature is available in the following measurement classes: GCA, GCX, Swept IMD, Swept IMDx, Noise Figure, NFX, Diff IQ, Spectrum Analyzer, and standard (S-Parameter) channels.

This function corrects the measurement and displays the results as if the measurement had been made into the specified impedance value. However, the physical port termination is still approximately 50 ohms.

The specified impedance value is applied to all of the measurements on ONLY the active channel.

See **Order of Fixture Operations**.

**Enable Port Z Conversion**  Check to apply the settings to the measurement results. Must also enable **Fixturing ON/off**.

**R**  Real part of the impedance value.

**jX**  Imaginary part of the impedance value.

**Close**  Applies the entries and closes the dialog box.
Note: Port Z (Impedance) conversion uses values in the following prioritized order:

1. Balanced (Differential or Common Mode) - if enabled, these values are always used.
2. Single Port Impedance - if enabled, this value is used if Balanced is not enabled.
3. System Impedance - if neither balanced or single port is enabled, this value is used.

4/6/8-Port Embed/De-embed dialog box help

This function specifies a single-ended 4-port circuit (*.S4P file) to embed (add) or de-embed (remove) from the measurement results. Computation takes place BEFORE Balanced conversion. See Order of Fixture Operations.

There is a single normalized impedance value for each port in the *.S4P file. This impedance value must match the impedance of the previous Port Z setting, or the VNA port impedance.

The VNA will interpolate if the number of data points that are read is different from the current VNA setting.

Enable 4-Port Embed/De-embed Check to apply the settings to the measurement results. Must also enable Fixturing ON/off.

Enable Extrapolation Check to apply a simple extrapolation when the S4P file has a narrower
frequency range than the channel. The values for the first and last data points are extended in either
direction to cover the frequency range of the measurement. The frequency ranges of both the
channel and the S4P file are displayed at the bottom of the dialog.

When extrapolation is necessary and enabled, a message is displayed showing the frequency range
to be extrapolated. When extrapolation is necessary and disabled, a message is displayed offering to
enable extrapolation.

This setting also causes 2-port Extrapolation to be enabled and disabled.

**Topology**

Select a DUT topology. Refer to the images on the 4-port embed/De-embed dialog box.

- **A** - Network 1
- **B** - Network 1/3
- **C** - Network 1/2/4

**NA Ports** - Select the VNA Port that is connected to each circuit port.

**Network Ports** Select the network ports that represent the configuration of the S4P file. By default,
ports 1 and 2 are connected to the VNA and ports 3 and 4 are connected to the DUT.

**None, Embed, De-embed** For Network1 and Network2, select:

- **None** - The same as disabling.
- **Embed** - Add the specified network circuit to the measurement results. See 2-port Embed image.
- **De-embed** - Remove the specified network circuit from the measurement results. See 2-port De-embed
  image.

**Browse** For both Network1 and Network2, navigate to find the .S4P file to embed or de-embed.

**OK** Applies the changes and closes the dialog box.

**Cancel** Does NOT apply the changes and closes the dialog box.
Differential Impedance Conversion dialog box help

This function sets the Differential impedance value for each balanced port.

The default value for R: is the SUM of the impedance values for both ports that make the logical port. If Port Z Conversion is not enabled, then System Z0 values for both ports are summed.

See Order of Fixture Operations.

Enable Differential Z Conversion Check to apply the settings to the measurement results. Must also enable Fixturing ON/off.

Logical Port Select the logical (balanced) port to receive impedance value. To see logical port numbers, see the measurement topology.

R Real part of the impedance value.

jX Imaginary part of the impedance value.

Close Closes the dialog box.

See note about Port Impedance priority.

Common Mode Impedance Conversion dialog box help

This function sets Common Mode Impedance value for each balanced port.

The default value for R: is calculated as follows.

\[(Z1 \times Z2) / (Z1 + Z2)\]

Where ports 1 and 2 comprise the logical port:

Z1 = the Port Impedance values for port 1
Z2 = the Port Impedance values for port 2

If Port Z Conversion is not enabled, then System Z0 values for port 1 and 2 are used in the calculation.

See Order of Fixture Operations.

Enable Common Mode Z Conversion  Check to apply the settings to the measurement results. Must also enable Fixturing ON/off.

Logical Port  Select the logical (balanced) port to receive impedance value. To see logical port numbers, see the measurement topology.

R  Real part of the impedance value.

jX  Imaginary part of the impedance value.

Close  Closes the dialog box.

See note about Port Impedance priority.

---

**Differential Port Matching** dialog box help

This function allows the embedding of a differential matching circuit at a balanced port.

See Order of Fixture Operations.

Enable Differential Port Matching  Check to embed the selected matching circuit to the measurement results. Must also enable Fixturing ON/off.
**Logical Port**  Choose Logical DUT port to receive the selected matching circuit. To see logical port numbers, see the measurement topology.

**Select Circuit**  Select a matching circuit. Choose from:

- **Shunt L - Shunt C**  Predefined circuit.

![Diagram of Shunt L - Shunt C circuit](image)

**Circuit Values**  Choose from:

- **C**  Capacitance value
- **G**  Conductance value
- **L**  Inductance value
- **R**  Resistance value

- **User defined**  Select an *.S2P file that represents the matching circuit. Then click **Browse** to navigate to the *.S2P file.

**Note:** For the *.S2P file:
Port 1 of the circuit is assumed to be connected to the VNA
Port 2 of the circuit is assumed to be connected to the DUT.

- **None**  No embedded circuit on selected port.

**Close**  Closes the dialog box.
This function adjusts the source power at the specified port to compensate for the combined amount of gain or loss through specific fixturing operations. Use this function to set the power level at the DUT input.

Power Compensation adjusts the source power for the gain/loss through all fixture components.

- It compensates for all port matching (embedded) networks unless "Compensate Only For De-Embeds" is selected.

For example:

- Your DUT requires a fixture on the input port which is connected to VNA port 1.
- The fixture description (such as an S2P file at the 2-port De-embed function) indicates the fixture has approximately 2 dB of loss across the frequency span.
- You set source power to 0 dBm. But you want 0 dBm at the DUT input (the fixture output).
- Check Power Compensation on Port 1 and enable Fixturing.
- Power Compensation causes the source power to be increased by approximately 2 dB so that the power at the fixture output plane will remain at 0 dBm.

Power Compensation affects all measurements in the channel.

Enable Fixturing to use Power Compensation.

Note: Use caution when applying power compensation. Always test your setup without a DUT in a place. If you are using S2P files, Recall your S2P file into the VNA so you can verify that the device
your S2P file describes is what you intended it to be. It is too easy to misalign data in S2P files if they are constructed by hand.

**Ground Loop De-embedding / Embedding**

Ground loop de-embedding removes the effect of a non-ideal ground connection between the DUT’s ground and the analyzer's ground reference. Typically, the non-ideal component is the parasitic inductance of the ground contacts.

Ground loop embedding adds the effect of a non-ideal component on the ground contacts.

The Ground Loop De-embedding / Embedding can be specified by circuit model type or touchstone file.

Ground loop de-embedding / embedding is only available from SCPI remote interface.

**Fixture Simulator Example**

The following example shows a DUT and the matching circuit with which the DUT will be used in its intended application. When the DUT is tested in a high-volume manufacturing environment, multiple test fixtures are often required. The most accurate way to test the DUT and ensure measurement consistency between the different test fixtures is to use a simple, repeatable, test fixture without the actual matching elements.

To get the desired performance data, the parasitic effects of the fixture must first be removed (de-embedded) from the measured data. Then a perfect "virtual" matching circuit must be simulated and added mathematically (embedded) to the corrected, measured data. The result is an accurate display of the DUT as though it was actually tested with a physical matching circuit, but without the uncertainties of using real components.
This diagram does NOT refer to the order in which operations are performed.

See Order of Fixture Operations.

1. Create a balanced measurement using single-ended to balanced (SE-Bal) topology. Include all relevant measurement settings (IFBW, number of points, and so forth). Once the measurement is created and calibrated, the measurement parameter can be easily changed. For example, Sdd22 to Sds21.

2. Calibrate the measurement at the point where the simple test fixture is connected to the VNA. Use accurate calibration standards and definitions.

3. Remove the effects of the three uncalibrated transmission lines of the simple test fixture. This can be done in several different methods. The easiest is to use manual or automatic Port Extensions to move the calibration reference plane to the DUT. This removes the electrical length and loss of the fixture’s transmission lines, but does not account for fixture mismatch. Another method is to de-embed previously-created *.S2p files of the 3 transmission lines. The files can be created using external ADS modeling software. Another alternative is to create the *.S2P files by independently measuring all 3 ports of the test fixture and saving the results of each to an S2P file.

4. With the test fixture connected to the VNA and a DUT inserted, the measurement results now appear as
though calibration was performed at the connections to the DUT, and the device was measured in a 50-ohm single-ended test environment. The following steps will cause the results to reflect the performance of the device as though the device is embedded in the circuit in which it will be used.

5. Port 1 of the device is a single-ended port and sees a source impedance the same as the VNA system impedance, so no change is required. However, if Rs were a value other than 50 ohms, Port 1 Impedance Conversion would be used to simulate the different impedance.

6. **Port Matching** is used to simulate L1 inductance. Select any of the Shunt L circuits to embed (add) to the measurement results. Enter the value of L and R. The C and G values can be entered as 0 (zero).

7. **Port Matching** is used to simulate C1 and C2 capacitance. For both port 2 and port 3, select any of the **Series C** circuits to embed (add) to the measurement results. Enter the value of C and G. The L and R values can be entered as 0 (zero).

8. **Balanced Conversion** mathematically simulates the measurement in balanced mode.

9. **Differential Port Matching** is used to simulate L2 inductance. Select Shunt L- Shunt C and enter the inductance / resistance value. The C and G values can be entered as 0 (zero).

10. Finally, **Differential Z Conversion** is used to simulate a circuit termination of 200 ohms. If you are making Common Mode measurements, specify **Common Mode Z Conversion**.
Automatic Fixture Removal (AFR)

Note: This feature requires Option S9x007A or 007.

In this topic:

- Overview
- Requirements
- Automatic Fixture Removal Wizard

Overview

Fixtures are often used for DUTs that have non-coaxial interfaces. This feature allows you to mathematically remove, or de-embed, a characterized test fixture from displayed measurement results of the test fixture and DUT.

Before starting the AFR process, **Perform a calibration** at the connectors of the test fixture (red lines in images below).

The AFR Wizard will guide you through these steps:

1. Press Cal > Fixtures > Auto Fixture Removal...
2. Describe your fixturing situation.
3. Specify How the Thru fixture characterization will occur.
4. Do characterization.
5. Remove the effects of the test fixture. Leaves ONLY the displayed results of the DUT.
6. Touchstone files are saved that characterize the two halves of the test fixture.

Requirements

- Test fixture and either a complete Thru **OR** Left and Right Thru halves, all shown below.
- Both the Test fixture and Thru are made of the exact same medium; preferably fabricated ON the same piece of medium.

Single-ended Fixtures
Supports both 1-port (not shown) and 2-port single-ended DUT configurations.

When used with a 4-port DUT configuration, the trace coupling is not removed.

**Differential (Balanced) Fixtures**
Left and Right halves of Thru standard

Supports 4-port DUT configuration ONLY which includes removing the effects of coupling between the differential traces.

How to start the Automatic Fixture Removal Wizard

With a calibrated measurement of the DUT in the test fixture present:

Click Response, then Cal, then Fixtures, then Automatic Fixture Removal

Note: The dialogs below show images for a Single ended DUT, but Differential works exactly the same, only with Differential S4P files.

Automatic Fixture Removal Wizard

For best results, follow the AFR Wizard tabs from steps 1 through 5 by either clicking Next > or clicking the tabs.

In this section:

1. Describe Fixture
   - AFR Configuration Dialog
2. Specify Standards
3. Measure Standards
4. Remove Fixture
5. Save Fixture
1. Describe Fixture

The choices that you make in the dialog are reflected in the diagram and text (red box in following image).

My fixture inputs are:

- **Single Ended** - The fixture and DUT have single-ended inputs and outputs.
- **Differential** - The fixture and DUT have differential inputs and outputs.

My measurement is (Single Ended):

- **1-port** - such as S11
- **2-port** - such as S21

My measurement is (Differential):

- **Multiport / Use THRUs** - select up to 64 ports. Check **Use THRUs** to use THRU fixture.

My measurement is (Differential):

- **2-port** - such as SDD11
- **4-port** - such as SDD21

- **Multiport / Use THRU** - select up to 64 ports. Check **Use THRU** to use THRU fixture.

Advanced Settings (click ^ to show and hide)

After fixture removal set Calibration Reference Z0 to: (Choose one of these settings)

- **System Z0** - Sets impedance to the System Impedance setting. Learn how to set System Z0.

- **Measured Fixture Z0** - Sets impedance to the value that is measured during the AFR process. Not allowed when 'band limited' is selected below.

- **<nn> ohms** - Sets impedance to an arbitrary value.

- **Set 'System Z0' to Calibration Reference Z0** - When the impedance is measured or set to an arbitrary value, check to also set the System Z0 to the same value.

Select all that apply:
- Check if this is true: **I want to correct for when the match** (Return Loss) of Fixture A is NOT equal to the match of Fixture B.

- Check if this is true: **I want to correct for when the electrical length** of Fixture A is NOT equal to the electrical length of Fixture B.

- Check if this is true: **My fixture is band limited**. Bandpass mode will be used during the Time Domain measurement. If NOT checked, then **Lowpass** mode is used. Because Lowpass mode includes impedance in the calculation, it renders the best accuracy. Learn more about these settings.

**Note**

When using **Lowpass** mode and an error message appears ("**Measurement settings are not adequate...**"), change the start frequency and the number of points so that the frequency span between data points equals the start frequency. This can be done by selecting values using the following logic:

- **Start freq = 10 MHz**

  then either:

  - **Stop freq = 20 GHz**
  - **Number of points = 2000**

  or

  - **Stop freq = 50 GHz**
  - **Number of points = 5000**

  In either case, the frequency span between data points equals 10 MHz, the start frequency.

**AFR Configuration Dialog**

The AFR Configuration dialog can be launched by clicking on the **AFR Config** icon (shown below).
Access the following settings in the **General** tab of the AFR Configuration dialog.
Set manual time domain start and stop settings for corner cases shown under Time Domain Settings.

The Tolerance (in ohms) and Number of Iterations can be set under Wizard Tab 3, Edit Window, Measured Fixture Impedance Iteration. Check the Auto Iterate box to automatically iterate when Impedance Method Auto is checked.

AFR Mode Conversion - AFR mode conversion will turn on or off automatically to make the AFR result more precise with non-single ended data.

---

2. Specify Standards
Selecting 2 single-ended ports, or 4 differential ports allows a 2X Thru choice.

Checking **Use THRUs** in the **1. Describe Fixture** tab allows 2X Thru choice for a Multiport measurement.
If **My characterization fixture** \`DUT measurement fixture\` is checked in the **1. Describe Fixture** tab, the **Fixtured DUT** choice is displayed and checked.

Otherwise, the fixture MUST be characterized using 1-port fixture measurements using either **OPEN** or **SHORT** terminations.

**Note:** The term 'Standards' is used here because this process can be thought of as the second in a '2-tier' calibration. The first tier of the calibration must already be performed (the VNA calibrated) before starting the AFR process. Another way of describing this step would be:

"How will you be measuring or loading the characterization of the Thru standard?"

- **2X Thru** - Both halves together in one fixture.
- Both **1X Thru** halves separately (also known as 1-port AFR). Specify either **Open** and/or **Short** at the end of each half of the standard.
Notes:

- This image: appears as the 2X Thru when you select 1. Describe Fixture, My measurement is 1 port. This means the left and right fixtures are mirror images of each other and have the same S-parameters. The cascaded combination of the two fixtures are Fixture A + Reversed Fixture A, or A'.
- If both halves are identical and you do not have a 2X Thru, then only Fixture A measurements are necessary.

Advanced Settings

This setting is used to describe any ADDITIONAL length between the halves of the Thru or added to either of the individual halves.

If the electrical length of the Thru standard is identical to the test fixture, then make no changes to the default settings (Known length = 0).

My Thru fixture has:

- **Known thru length** - Enter the length in nanoseconds. See a simulated length in the diagram between the two halves of the Thru.
- **Unknown thru length computed using reflects**. This setting requires the two halves of the Thru fixture be characterized separately with a reflect standard.
- **Unknown thru length computed using fixtured DUT measurements**. This setting requires an additional characterization of the Fixture + DUT.

3. Measure Standards
The following dialog is for Multiport 2X Thru measurements when Use THRUs is checked in the 1. Describe Fixture tab.

This step characterizes the 1X or 2X Thru standards. This is done by either performing
measurements or by loading one or more *.snp files that describe the characterization of the fixtures.

**Note:** When loading standards from files, the typical system characterization impedance (Z0) value is used, which is 50 &.

Click **Measure** to see the following dialog:

For best results, the analyzer should be calibrated. Also, the measurement Start and Step frequencies should be equal. This is necessary for TDR measurements.

Connect the specified standard at the PNA port. Then click **Measure**.

![Measure dialog](image)

OR click **Load** and navigate to the *.snp file that describes the standard.

For 2X Thru standards, the following dialog allows you to optionally remap the fixture ports.

![Load dialog](image)

**Calculated Fixture Characteristics**
**Note:** The fixture length must be 4 times the rise time. If not, an error message will be displayed. For example, a measurement at 26.5 GHz has a rise time of 37.7 ps. Therefore, the fixture length must be $4 \times 37.7 \text{ ps} = 151 \text{ ps}$.

The loaded or measured Impedance and Electrical Length of the fixture are calculated and displayed here.

The preview button allows you to visualize the time domain characteristics of the fixture model that will be used for de-embedding. By using the editing features noted below, the output match, length, and impedance of the fixture model can be optimized. This allows flexibility for including or excluding physical characteristics such as excess inductance of plug fingers, excess capacitance of receptacle pads or any other anomalies not desired in the fixture model. This advanced feature should be used with care as misuse could cause passivity or causality errors in the resultant fixture model.

Click **Edit** to start the following, interactive **Measure Fixture Impedance TDR Data** dialog.
This dialog allows you to change the Impedance and/or Fixture Length of the saved *.snp data by moving a marker to the desired location on the TDR plot.

**Tips**

- The TDR X-axis shows the **1-way** electrical length of the fixture in time.
- Right-click the plot, then click **Autoscale**.
- Click and drag to zoom the display on specific details of the plot.

**Editing Mode.** (For 2X Thru, Fixture Length can NOT be changed.)

Choose one or more of the following to edit the data in resulting *.snp files:

- **Impedance** - move the M1 marker to edit the data as though the measurement was made in the specified characteristic impedance. The Z marker is used to measure the fixture impedance.
*Gate* - move the flag to 'gate out' a part of the fixture measurement, usually close to the transition. The Z marker automatically changes to the location of the gate. The gate flag cannot move further than the fixture length diamond marker.

*Fixture Length* - move the diamond marker to change only the electrical length of the fixture. The fixture length cannot be shortened to a value less than the current gate value. In this way, the gate value will never be greater than the electrical length.

**Impedance Editing Mode**

**Select Fixture Impedance** - Select the A or B fixture.

**Impedance Method** - Select *Auto* then click *Iterate* to measure the fixture Z0 automatically and return fixture impedance back to 50 ?. If more than one fixture impedance needs to be measured, check *All* to measure all impedances listed in *Select Fixture Impedance*. Select *Marker* to measure the fixture Z0 manually using the Z marker. The *Set At* field shows the X-axis position of the Z marker in ns. The *Set Marker At Default* returns the Z marker and fixture impedance to the default value.

**Measured Fixture Z0** - Shows the measured fixture impedance. When *Marker* is selected as the Impedance Method, this field displays the current Z marker value or a value can be entered directly in this field. When *Auto* is selected as the Impedance Method, this field displays the fixture impedance value that was found automatically after clicking the *Iterate* button. In *Auto*, moving the Z marker will not change this value.

**Diff: ZA:** - Enter the AFR fixture impedance manually using the text box.

**Fixture Length Editing Mode**

**Select Fixture Impedance** - Select the A or B fixture.

**Set Marker Position, Gate Position, or Fixture Length** depending on the selected Editing Mode. Move the marker (M1, Gate flag, or diamond) to the desired response on the trace.
Set At Default - PLTS chooses an appropriate length at which to set the initial impedance or length. Click to return the marker to this location.

Preview ON/off - Select Off, then ON to preview the new measurement. Click OK when finished. When saved, the new Impedance and Length results are copied to the *.snp files.

4. Remove Fixture

![Image of the PLTS software interface for removing fixtures]

**Note:** First choose Select correction method, then make other selections, then click Apply Correction.

Both operations can be performed, but only one at a time.

- **Turning on Fixturing/de-embedding for channels.** Usually only one channel is being used on the VNA, so only one channel will appear in the choices of channels to be corrected.
  
  - **Select the channels to be corrected.**
    
    Fixturing and de-embedding will be enabled for the selected channels in the VNA. If the new fixturing UI is used, these de-embedding blocks will place to the left of any existing circuit elements. See Using Fixture Simulator for more details.
• Advanced settings

  • **Enable Extrapolation** - When fixture data is loaded from a file and the frequency span of the data is not as wide as that of the channel, check this box to calculate and use linearly-extrapolated fixture data.

  • **Compensate for power** - When checked, test port power is increased to compensate for loss in the fixture.

• **Modifying the calset(s) used on the channels.** Usually only one calset is in use on the VNA, so only one choice would be available.

  • Advanced settings

  • **Enable Extrapolation** - When fixture data is loaded from a file and the frequency span of the data is not as wide as that of the measurement, check this box to calculate and use linearly-extrapolated fixture data.

  • **Compensate for power** - When checked, test port power is increased to compensate for loss in the fixture.

  • **Prompt for new calset name(s)** - When cleared, when you apply AFR to a calset, the calset is overwritten. Once done, this process is NOT reversible. When checked, you are prompted to enter a new calset name and the original is preserved. The new calset is written with AFR correction.

  • **Apply AFR to other selected Calsets** - When checked, allows you to apply AFR correction to other calsets not currently in use on the VNA. Click Browse, then navigate to the calsets to be corrected.

---

5. Save Fixture
Select File format to save fixture data:

- Touchstone (*.snp)
- Touchstone 2 (*.ts)
- Citifile (*.cti)

Choose port assignment for save fixture files:

The port assignments are interpreted differently when the file is opened in each program. Choose which program software you will be using to open the saved file: PLTS, VNA, ADS.

Choose the directory and base names for the saved files:

Click Browse to navigate to a directory folder.

With a base file name: The resulting filename will appear as follows (assuming a Touchstone format):

- `<base file name>_1.S4P` - The left half of a Differential fixture.
- `<base file name>_2.S4P` - The right half of a Differential fixture.
- `<base file name>_1.S2P` - The left half of a single-ended fixture.
- `<base file name>_2.S2P` - The right half of a single-ended fixture.
Click **Save Fixture Files** to save the files to the specified directory.
Port Extensions

Port extensions allow you to electrically move the measurement reference plane after you have performed a calibration.

**Note:** This feature is available in GCA, GCX, Spectrum Analyzer, and standard (S-Parameter) channels.

- Why and How to use Port Extensions
- Manual Port Extensions Procedure
- Port Extensions dialog and Toolbar
- Step Size dialog
- Automatic Port Extension dialog

**See Also**

Data Flow Map

Fixture Compensation features

Phase Accuracy

Comparing the VNA Delay Functions

**Other Calibration Topics**

**Why use Port Extensions**

1. You are unable to perform a calibration directly at your device because it is in a test fixture. Perform a calibration at a convenient place, then use port extensions to compensate for the time delay (phase shift), and optionally the loss, caused by the fixture.

2. You have already performed a calibration, and then decide that you need to add a length of transmission line in the measurement configuration. Use port extensions to “tell” the analyzer you have added the length to a specific port.
Important Note: Port Extensions and VNA Data Flow

See VNA Data Flow diagram

Normally, Port Extensions are applied to individual S-parameters in the **Phase Correction** process and only applies to displayed S-parameters.

However, when **Fixturing** is ON or when making a **Balanced Measurement**, Port Extension compensation is applied in the **Apply Error Terms** process which affects **ALL** S-parameters, whether displayed or not. This allows all underlying S-parameters to have proper extensions applied.

Therefore, when using Port Extensions with features that require more than a single S-parameter (such as k-factor in equation editor), do one of the following:

- Enable **Fixturing** - Individual Fixturing features are NOT required to be enabled.
- Use **8510 Mode Data Processing**.

When Port Extension compensation is applied in the **Apply Error Terms** process, after a **Data-to-Memory** operation has been performed, further changes to Port Extensions settings will NOT be applied to the Memory trace.

---

**How to use Port Extensions**

- If you know the **electrical length** of the fixture or additional transmission line, enter the value directly to the **Time** setting.

- If you know the **physical length** of the fixture or additional transmission line, enter the value directly to the **Distance** setting.

- If you do **NOT** know either the electrical or physical length of the fixture or additional transmission line, you must be able to connect an **OPEN** or **SHORT** to the new reference plane - in place of the DUT. In most cases, removing the DUT will leave a suitable **OPEN** at the new reference plane.

- Port Extensions can then be added manually (as follows), or by using **Automatic Port Extensions**.

**Manual Port Extensions Procedure**

1. Select a calibrated S11 measurement.

2. Select Phase format.
3. With an OPEN or SHORT at the calibration reference plane, verify that the phase across the frequency span is at or near zero.

4. Connect the fixture or added transmission line and attach an OPEN or SHORT in place of the DUT. In most cases, removing the DUT will leave a suitable OPEN at the new reference plane. On the Port Extension toolbar or dialog, increase either **Time** or **Distance** until the phase response is flat across the frequency span of interest.

5. If you know the loss of the additional transmission line, enter the **Loss Compensation** values using either one or two data points.

**Note:** Most OPEN and SHORT standards have delay. Therefore, adjusting delay with this method results in a delay equal to two times the delay of the OPEN or SHORT.

---

**How to set Port Extensions**

**Using Hardkey/SoftTab/Softkey**

1. Press **Cal > Port Extension > Port Extensions...**

---

**Port Extensions dialog and Toolbar help**

**Note:** This feature is available to **GCA, GCX, Spectrum Analyzer,** and standard (S-Parameter) channels.

Port extensions settings affect all measurements on the active channel that are associated with a particular port.

Learn Why and How to use Port Extensions (scroll up).
**Port**  Select a port for delay and loss values. Port Extensions settings affect ALL measurements on the active channel that are associated with a particular port.

**Port Extension**  Turns ON and OFF port extensions on all ports.

**Show Toolbar**  Check to show the Port Extensions toolbar. The toolbar allows you to make adjustments to the port extensions while showing more of the screen. This is the only way to show or hide the toolbar.

**Delay**

Enter delay in either Distance or Time by entering a value or clicking the up/down arrows. Click to start the Step Size dialog.

**Time**  The amount of port extension delay in time. Enter a positive value.

**Distance**  The amount of port extension delay in physical length. Enter a positive value.

**Distance Units**  (Dialog ONLY) Select from Meters, Inches, or Feet. The Step Size setting will not change automatically. Learn more.

**Loss**

The following settings allow the entire frequency span to be corrected for loss.

**Loss at DC**  Offsets the entire frequency span by this value. Loss1 or Use1 must also be checked. To compensate for loss at DC, enter a positive value which causes the trace to shift in the positive (up) direction.

**Loss @ Frequency**  Check the box, and enter values for Loss and Frequency.

When Loss1 or Loss1/Loss2 are used, a curved-fit algorithm is used as follows:

**Loss1 ONLY:**

\[ \text{Loss}(f) = \text{Loss1} \times \left(\frac{f}{\text{Freq1}}\right)^{0.5} \]

**Loss1 and Loss2:**

Set the lower frequency to Loss1, and the higher frequency to Loss2.

\[ \text{Loss}(f) = \text{Loss1} \times \left(\frac{f}{\text{Freq1}}\right)^{n} \]

Where:

\[ n = \log_{10} \left[ \frac{\text{abs} (\text{Loss1/Loss2})}{\text{Freq1/Freq2}} \right] \]

**Note:** abs = absolute value
**Velocity**

**Velocity Factor**  For each port, sets the velocity factor that applies to the medium of the device that was inserted after the measurement calibration. The value for a polyethylene dielectric cable is 0.66 and 0.7 for PTFE dielectric. 1.0 corresponds to the speed of light in a vacuum.

**Couple to system Velocity Factor**  When unchecked, the Velocity Factor is set for only the specified port and only for Port Extensions. When checked, sets the Velocity Factor for all ports. In addition, changing this value also changes this setting for the Electrical Delay and Time Domain Distance Marker features.

**Media**

For each port, select the media of the added transmission line or fixtures.

**Coax**  Select when the fixture or added transmission line is coax. Also specify the velocity factor of the coax.

**Waveguide / Cutoff Frequency**  Select when the fixture or added transmission line is waveguide. Also enter cutoff (minimum) frequency of the waveguide.

**Note:**  when using a Waveguide cal Kit, set System Z0 to 1 ohm before calibrating.

**Couple to system Media Definition.**  When unchecked, the Waveguide Cutoff Frequency is set for only the specified port and only for Port Extensions. When checked, sets the Waveguide Cutoff Frequency for all ports. In addition, changing this value also changes this setting for the Electrical Delay feature.

**Reset**  All port extensions settings are changed to preset values. The Port Extension ON / OFF state is NOT affected.

**Auto Ext.**  Starts the Automatic Port Extensions dialog box

**Note:**  Individual receiver port extensions (A,B, and so forth) cannot be set.
**Step Size dialog box help**

Changes the step size that occurs when the Time or Dist up/down arrows are pressed on the Port Extension toolbar. The Units for step size are changed on the Port Extension dialog.

**Auto**  Step Size is set to the default value.

**User Defined**  Enter a step size value, then click OK.

This value remains the same when the units are changed. For example if a step size of 12 is entered on this dialog, then you change the units from Inches to Feet, the step size of 12 inches becomes 12 feet, not 1 feet. Therefore, change the units first, then set the step size.

Learn about Port Extensions (scroll up)

---

**Automatic Port Extension dialog box help**

Automatic Port Extension AUTOMATICALLY performs the same operation as Manual Port Extension. By connecting a SHORT or OPEN, the reference plane is automatically moved to the point at which the standard is connected. In addition, Automatic Port Extension will optionally
measure and compensate for the loss of the additional transmission line.

Auto Port Extension is NOT available when:

- Sweep type is set to power sweep
- Frequency Offset is ON
- Media is set to Waveguide

**Note:** Turn OFF Equations that may exist on the active trace when using Automatic Port Extensions.

### Auto Port Extensions Procedure

1. Connect the added transmission line or fixture. Attach an OPEN or SHORT to all affected ports at the new reference plane. In most cases, removing the DUT will leave a suitable OPEN at the new reference plane.

2. On the Port Extension toolbar, click **Auto Port Ext**. Click **Show Configuration** to make additional settings.

3. Click **Measure** to perform the port extension calculations. The resulting delay and loss settings are entered into the port extension toolbar. These settings are saved with Instrument Save or you can manually record the values and enter them again when required.

### Settings

**Measure either OPEN, SHORT, or both** Press a button to make the measurement of the reflection standard.

Measure either OPEN or SHORT depending on which is most convenient. An ideal OPEN and SHORT, with zero loss and delay, is assumed. Therefore, accuracy is most affected by the quality of the standard. In most cases, removing the DUT will leave a suitable OPEN at the new reference plane. When measuring both OPEN and SHORT standards, the average of the two is used and will slightly improve accuracy.

**Selected Ports** Indicates the ports that currently have automatic port extension enabled. By default, ALL analyzer ports are enabled. To disable a port, see **Measure on Port Number** below.

**Note:** Port Extensions settings affect ALL measurements on the active channel that are associated with a particular port.

**Show/Hide Configuration** Press to either show or hide the following configuration settings in the dialog box.
### Measure on Port Number

Select port number to enable or disable automatic port extension.

**Enable** Check to enable the specified port. All enabled ports will have their reference plane automatically adjusted after performing Automatic Port Extension.

**Include Loss** Check to automatically measure the loss in the additional transmission line and apply compensation. To calculate loss compensation, frequencies at 1/4 and 3/4 through the frequency range are usually used as Freq1 and Freq2 values. [Learn more about Loss Compensation.](#)

**Adjust for Mismatch** Only available when **Include Loss** is checked. Mismatch adds ripple to the S11 and S22 traces. If the ripple is large, S11 and S22 can appear greater than 0 dB which leads to numeric instabilities in using the S-parameters. Adjust for mismatch increases the loss of the fixture so that the peak of the ripples is below 0 dB. While this adds more error (all the error is negative) it does allow the S-parameters to be used in simulators without numerical instabilities.

- **Check** - Offsets the trace to cause all of the data points to be at or below zero.
- **Clear** - Most accurate application of the curve-fit calculation, but allows positive responses.

**Prompt for Each Standard** Check to invoke a prompt when the Measure OPEN or SHORT button is pressed. The prompt will indicate which standard to connect to which port.

### Method

Select the span of data points which will be used to determine correction values for phase and loss (optional). If a portion of the current frequency span does not have flat or linear response, you can eliminate this portion from the calculations by using a reduced User Span.

To calculate loss compensation, Current Span and User Span methods usually use frequencies at 1/4 and 3/4 through the frequency range as Freq1 and Freq2 values. See [Loss Compensation](#) to learn more about how loss is calculated.

**Current Span** Use the entire frequency span to determine phase and loss values.

**Active Marker** Use only the frequency at the active marker, and one data point higher in frequency, to calculate phase and loss values. If a marker is not present, one will be created in the center of the frequency span.

**User Span** Use the following User Span settings to determine phase and loss values.

### User Span

**Start** Enter start frequency of the user span.
**Stop**  Enter stop frequency of the user span.

Learn about Port Extensions (scroll up).

See also Comparing Delay Functions
Swap Adapters and Offset Delay Calibration Methods

The Swap Adapters or Offset Delay calibration method is used when you do NOT have calibration standards with the same connector type as your DUT. In this case, the Offset Delay is the preferred calibration method over the Swap Adapters method.

The Swap Adapters calibration method (also known as Swap Equal Adapters and Equal Length Adapters) was used in the past as a quick alternative to the more tedious adapter removal method. This method requires that the adapters be of equal electrical length. There are two adapters for each port. The swap equal adapter method implicitly assumes the adapters have identical return loss. The finite return loss of each adapter on each port degrades both the residual directivity and residual match terms. The offset delay calibration only has one adapter. The return loss of this one adapter will degrade the residual directivity and residual match error terms.

**Note:** For any other reason, these calibration methods are **NOT** recommended because the Unknown Thru method is more convenient AND more accurate.

The Offset Delay calibration method uses the available standards for calibration then adds offset delay to the measurement plane to account for each adapter used. This eliminates the need for adapters with equal electrical length and is preferred over the Swap Adapters method.

**Swap Adapters Procedure**

The following is an example procedure showing how to perform a Swap Adapters 2-port calibration for a non-insertable DUT. The DUT has 2.92 mm connectors. You do NOT have 2.92 mm calibration standards, but you DO have 2.4 mm standards and adapters that have the same electrical delay as the 2.92 mm adapters.

Adapters A1 and A2 = test port to 2.4 mm adapters

Adapters B1 and B2 = test port to 2.92 mm adapters
1. Start the Cal Wizard and select Guided (Smart) Cal. **Note:** The VNA will NOT prompt you to connect the adapters by name or when to swap the adapters.

2. Specify the connector type and gender and Cal Kit of the adapter that you will be using (2.4 mm) - NOT the connector type of the DUT (2.92 mm). By specifying the connector gender, you are also specifying the Thru method (flush thru for insertable and Unknown Thru for non-insertable.) For example, when both DUT ports have female connectors, we will perform an Unknown Thru cal.

3. When prompted for reflection standards on port 1, connect the Open, Short, and Load standards to Adapter A1.

4. When prompted for reflection standards on port 2, connect the Open, Short, and Load standards to Adapter A2.

5. When prompted for a Thru connection, swap Adapter A1 and A2 for B1 and B2. Connect the Thru device. This could be any device that meets the requirements of the Unknown Thru standard. In the case of a non-insertable DUT, connect B1 and B2.

6. Make DUT measurements with Adapters B1 and B2 in place.

---

**Offset Delay Procedure**

The following is an example procedure showing how to perform a 2-port calibration for a non-insertable DUT using Offset Delay to account for the added delay of two adapters. The DUT has 2.92
mm connectors. You do NOT have 2.92 mm calibration standards, but you DO have 2.4 mm standards and adapters.

Adapters = test port to 2.4 mm (female)-to-2.92 mm (male) adapters

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Start the Cal Wizard and select Guided (Smart) Cal.</td>
</tr>
<tr>
<td>2.</td>
<td>Specify the connector type and gender and Cal Kit that you will be using (2.4 mm) - NOT the connector type of the DUT (2.92 mm). By specifying the connector gender, you are also specifying the Thru method (flush thru for insertable and Unknown Thru for non-insertable.) For example, when both DUT ports have female connectors, we will perform an Unknown Thru cal.</td>
</tr>
<tr>
<td>3.</td>
<td>When prompted for reflection standards on port 1, connect the Open, Short, and Load standards.</td>
</tr>
<tr>
<td>4.</td>
<td>When prompted for reflection standards on port 2, connect the Open, Short, and Load standards.</td>
</tr>
<tr>
<td>5.</td>
<td>When prompted for a Thru connection, connect the Thru device between port 1 and port 2. This could be any device that meets the requirements of the Unknown Thru standard. In this example of a non-insertable DUT, a female-to-female 2.4 mm barrel adapter is used as the Thru device.</td>
</tr>
<tr>
<td>6.</td>
<td>Click on Response, Cal, then select Port Extension. The Port Extension dialog is used to electrically move the measurement reference plane after you have performed a calibration to account for the two adapters. Learn more about Port Extensions.</td>
</tr>
<tr>
<td>7.</td>
<td>Select Port 1 and enter the delay of the adapter connected to Port 1.</td>
</tr>
<tr>
<td>8.</td>
<td>Select Port 2 and enter the delay of the adapter connected to Port 2.</td>
</tr>
<tr>
<td>9.</td>
<td>Click OK.</td>
</tr>
</tbody>
</table>
10. Make DUT measurements with Adapters and DUT in place.
Delta Match Calibration

A TRL Cal, QSOLT, or Unknown Thru Cal requires simultaneous, valid measurements of the reference receivers for the test ports being calibrated. This is not possible for both the 2-port AND 4-port versions of the N5231B, N5232B, and N5239B PNA-L models.

This is also NOT possible with older N5230C 4-port models, nor for certain port pairs when an external test set is connected to ANY VNA model. With external test sets, some pairs of ports share the same receiver. In general these are the port pairs where one port is directly above the other. Therefore, TRL, QSOLT, and Unknown Thru are NOT possible in the context of a 2-port cal for any of those port pairs.

A Delta Match Calibration can be thought of as a software method used to overcome this hardware limitation. The Delta Match Calibration characterizes the source match and load match of the VNA test ports, and then calculates the differences, or "delta", of the two match terms. The delta term is combined with directivity and reflection tracking to compute a switch correction term. The switch correction error terms for each port are then used during subsequent TRL, QSOLT, or Unknown Thru calibrations.

There are several ways to acquire the Delta Match Calibration:

1. **From an existing User Cal Set** that meets the following Delta Match criteria:
   
   - Must have been performed using ECal or as a guided mechanical Cal (not Unguided).
   - Must have the same start frequency, stop frequency, and number of points as the channel being calibrated.
   - Must calibrate the ports that require the delta match terms.

2. **From a Global Delta Match Calibration.**

3. **From a 'Self Delta Match'** when other portions of the calibration fully characterize all ports using SOLT with Defined Thru or Flush Thru. For example, when calibrating all four ports of a PNA-L, perform a SOLT between ports 1 and 2, and also between ports 3 and 4, then Unknown Thru could be used between any combination of the remaining ports. This is allowed with an external test set.

**Which to use?** A Self Delta Match Cal will always be used when possible. Otherwise, the Cal Wizard will use a GDM Cal when available unless you select Choose Delta Match.

Global Delta Match (GDM) Cal
A GDM Cal is an "all-inclusive" calibration that can be applied whenever the delta match terms are required.

**Factory GDM Cal**

Beginning with A.09.80, a factory-performed GDM calibration is installed on 2-port PNA-L models. This GDM Cal is identical to one that you would perform, except that it is likely to be more accurate than a GDM Cal that would be performed by most PNA-L customers. The factory GDM is used when a Self Delta Match Cal is not possible, just like a user-performed GDM.

- If you choose to perform a GDM Cal, it will overwrite the Factory GDM Cal.
- A backup copy of the factory GDM Cal is saved on the PNA-L at D:\calfiles\ Global Delta Match Calset.pcs.
- To restore the original file, copy it to the original location at: C:\ProgramData\Agilent\Network Analyzer\UserCalSets\Global Delta Match Calset.pcs.
- If the original file is deleted, it will be restored automatically from the backup location.

**User GDM Cal**

When a GDM Cal is required, and your PNA-L model does NOT have a factory GDM, it must be performed at least once in order to use TRL, Unknown Thru, or QSOLT calibrations

- To attain the highest accuracy, the following settings are automatically used to perform a GDM Cal. When applied, it will likely be interpolated.
  - Performed over the entire frequency range of the VNA.
  - If using an external test set, perform GDM with the external test set connected. There will be a separate GDM calibration stored for different PNA configurations. It is possible to have a GDM for the standalone system side by side with GDM calibrations for each type of external test set used.
  - Uses very dense data points, particularly at low frequencies.
  - Uses 100 Hz IF Bandwidth.
- The measurement conditions (cabling or adapters) of the subsequent calibration do NOT have to match the conditions under which the GDM Cal was performed. Because the GDM Cal characterizes the switch correction at each VNA port, those terms are NOT affected by differences in cabling or adapters.
- For highest accuracy when using an ECal module to perform the GDM Cal, the test ports of the ECal module should permit an **insertable connection** (for example 3.5mm male on one port and 3.5 mm female on the other). The frequency range of the ECal module must cover the entire frequency range. When using mechanical standards, the GDM Cal will force an insertable connection.
- Upon completion, the GDM Cal is stored as a special type of Cal Set and should be used ONLY as a Delta
Match Cal. It provides Delta Match error terms, but does NOT provide all of the standard error correction terms.

- Experience has shown that a GDM Cal may require updating only once every year depending on environmental conditions and how heavily the instrument is used.

**How to perform a GDM Cal**

**Note:** 2-port PNA-L models have a Factory GDM Cal, which means it is NOT necessary to perform a GDM Cal. If you perform a GDM, the Factory GDM will be overwritten.

These selections will only be available if the VNA hardware requires a Delta Match Calibration.

**Using Hardkey/SoftTab/Softkey**

1. Press **Cal**
2. then **Start Cal**
3. then **Global Delta Match**

**Using Menus**

1. Click **Response**
2. then **Cal**
3. then **Start Cal**
4. then **Global Delta Match**

**Delta Match Calibration. Select DUT Connectors and Cal Kit** dialog box help

- Only one Cal Kit is specified and necessary to perform a Delta Match Cal. However, ALL of the VNA test ports are calibrated in a Delta Match Cal.

- You must configure ALL test ports to terminate in the specified connector / gender using the necessary adapters. The errors from adapters are removed during calibration, but when using mechanical standards, the GDM Cal will force an insertable connection.

- For highest accuracy when using an ECal module to perform the GDM Cal, the test ports of the ECal module should permit an insertable connection (for example 3.5mm male on one port and 3.5 mm female on the other). The frequency range of the ECal module must cover the entire frequency range. When using mechanical standards, the GDM Cal will force an insertable connection. If you select an ECal
module that does NOT cover the entire frequency range of the VNA your selection will change to a different Cal Kit.

**Guided Calibration Steps** dialog box help

Follow the prompts to connect standards to the calibration plane.

Then click **Measure**.

When all standards have been measured, click **Done** to complete the measurement steps.

**THRU Connections**

- ONLY 2 Thru connections are required for a 4-port Cal. This is less than the minimum number of Thrus of a standard 4-port Cal.

- GDM Cal is always performed using a **Flush Thru**, a **Known Thru**, or an **insertable ECal module**. You can NOT use an Unknown Thru in the calibration process. For highest accuracy, perform GDM Cal using an insertable ECal module and select Flush-thru as the Calibration Thru method. Learn about Calibration Thru methods.
Click **Finish** to store the GDM Calibration as a special type of Cal Set.

By default, it will be used when a Delta Match Calibration is required.

It can ONLY be used as a Delta Match Cal. It does NOT provide all of the standard error correction terms.
Calibration Overview

The following is discussed in this topic:

- What Is Measurement Calibration?
- Why Is Calibration Necessary?
- Conditions Where Calibration Is Suggested
- What Is ECal?

See other Calibration Topics

What Is Measurement Calibration?

Calibration removes one or more of the systematic errors using an equation called an error model. Measurement of high quality standards (for example, a short, open, load, and thru) allows the analyzer to solve for the error terms in the error model. See Measurement Errors.

You can choose from different calibration types, depending on the measurement you are making and the level of accuracy you need for the measurement. See Select a Calibration Type.

The accuracy of the calibrated measurements is dependent on the quality of the standards in the calibration kit and how accurately the standards are modeled (defined) in the calibration kit definition file. The calibration-kit definition file is stored in the analyzer. In order to make accurate measurements, the calibration-kit definition must match the actual calibration kit used. To learn more, see Accurate Calibrations.

Calibration Wizard provides the different calibration methods used in the VNA. See Calibration Wizard.

There are quick checks you can do to ensure your measurement calibration is accurate. To learn more see Validity of a Measurement Calibration.

If you make your own custom-built calibration standards (for example, during in-fixture measurements), then you must characterize the calibration standards and enter the definitions into a user modified calibration-kit file. For more information on modifying calibration kit files, see Calibration Standards.

Note: Instrument Calibration is ensuring the analyzer hardware is performing as specified. This is not the same as measurement calibration.
Why Is Calibration Necessary?

It is impossible to make perfect hardware that would not need any form of error correction. Even making the hardware good enough to eliminate the need for error correction for most devices would be extremely expensive.

The accuracy of network analysis is greatly influenced by factors external to the network analyzer. Components of the measurement setup, such as interconnecting cables and adapters, introduce variations in magnitude and phase that can mask the actual response of the device under test.

The best balance is to make the hardware as good as practically possible, balancing performance and cost. Calibration is then a very useful tool to improve measurement accuracy.

Conditions Where Calibration Is Suggested

Generally, you should calibrate for making a measurement under the following circumstances:

- You want the best accuracy possible.
- You are adapting to a different connector type or impedance.
- You are connecting a cable between the test device and an analyzer test port.
- You are measuring across a wide frequency span or an electrically long device.
- You are connecting an attenuator or other such device on the input or output of the test device.

If your test setup meets any of the conditions above, the following system characteristics may be affected:

- Amplitude at device input
- Frequency response accuracy
- Directivity
- Crosstalk (isolation)
- Source match
- Load match

What Is ECAL

ECal is a complete solid-state calibration solution. It makes one port (Reflection), full two and three-
port calibrations fast and easy. See Using ECal.

- It is less prone to operator error.
- The various standards (located inside the calibration module) never wear out because they are switched with PIN-diode or FET switches.
- The calibration modules are characterized using a TRL-calibrated network analyzer.
- ECal is not as accurate as a good TRL calibration.

For information about ordering ECal modules, see Analyzer Accessories or contact your Keysight Support Representative.
Measurement Errors

You can improve accuracy by knowing how errors occur and how to correct for them. This topic discusses the sources of measurement error and how to monitor error terms.

- Drift Errors
- Random Errors
- Systematic Errors
  - 3-Port Error Terms
  - 4-Port Error Terms
- Monitoring Error Terms

See other Calibration Topics

Drift Errors

Drift errors are due to the instrument or test-system performance changing after a calibration has been done.

Drift errors are primarily caused by thermal expansion characteristics of interconnecting cables within the test set and conversion stability of the microwave frequency converter and can be removed by re-calibrating.

The time frame over which a calibration remains accurate is dependent on the rate of drift that the test system undergoes in your test environment.

Providing a stable ambient temperature usually minimizes drift. For more information, see Measurement Stability.

Random Errors

Random errors are not predictable and cannot be removed through error correction. However, there are things that can be done to minimize their impact on measurement accuracy. The following explains the three main sources of random errors.

Instrument Noise Errors
Noise is unwanted electrical disturbances generated in the components of the analyzer. These disturbances include:

- Low level noise due to the broadband noise floor of the receiver.
- High level noise or jitter of the trace data due to the noise floor and the phase noise of the LO source inside the test set.

You can reduce noise errors by doing one or more of the following:

- Increase the source power to the device being measured - ONLY reduces low-level noise.
- Narrow the IF bandwidth.
- Apply several measurement sweep averages.

Switch Repeatability Errors

Mechanical RF switches are used in the analyzer to switch the source attenuator settings.

Sometimes when mechanical RF switches are activated, the contacts close differently from when they were previously activated. When this occurs, it can adversely affect the accuracy of a measurement.

You can reduce the effects of switch repeatability errors by avoiding switching attenuator settings during a critical measurement.

Connector Repeatability Errors

Connector wear causes changes in electrical performance. You can reduce connector repeatability errors by practicing good connector care methods. See Connector Care.

Systematic Errors

Systematic errors are caused by imperfections in the analyzer and test setup.

- They are repeatable (and therefore predictable), and are assumed to be time invariant.
- They can be characterized during the calibration process and mathematically reduced during measurements.
- They are never completely removed. There are always some residual errors due to limitations in the calibration process. The residual (after measurement calibration) systematic errors result from:
  - imperfections in the calibration standards
  - connector interface
Reflection measurements generate the following three systematic errors:

- Directivity
- Source Match
- Frequency Response Reflection Tracking

Transmission measurements generate the following three systematic errors:

- Isolation
- Load Match
- Frequency Response Transmission Tracking

Notes about the following Systematic Error descriptions:

- The figures for the following six systematic errors show the relevant hardware configured for a forward measurement. For reverse measurements, internal switching in the analyzer makes Port 2 the source and Port 1 the receiver. 'A' becomes the transmitted receiver, 'B' becomes the reflected receiver, and 'R2' becomes the reference receiver. These six systematic errors, times two directions, results in 12 systematic errors for a two port device.

- For simplicity, it may be stated that ONE standard is used to determine each systematic error. In reality, ALL standards are used to determine ALL of the systematic errors.

- The following describes an SOLT calibration. This does not apply to TRL or other types of calibration.

Directivity Error

All network analyzers make reflection measurements using directional couplers or bridges.

With an ideal coupler, only the reflected signal from the DUT appears at the 'A' receiver. In reality, a small amount of incident signal leaks through the forward path of the coupler and into the 'A' receiver. This leakage path, and any other path that allows energy to arrive at the 'A' receiver without reflecting off the DUT, contributes to directivity error.
How the Analyzer Measures and Reduces Directivity Error.

1. During calibration, a load standard is connected to Port 1. We assume no reflections from the load.

2. The signal measured at the 'A' receiver results from the incident signal leakage through the coupler and other paths.

3. Directivity error is mathematically removed from subsequent reflection measurements.

Isolation Error

Ideally, only signal transmitted through the DUT is measured at the 'B' receiver.

In reality, a small amount of signal leaks into the 'B' receiver through various paths in the analyzer.

The signal leakage, also known as crosstalk, is isolation error which can be characterized and reduced by the analyzer.

How the Analyzer Measures and Reduces Isolation Error
1. During calibration, load standards are connected to both Port 1 and Port 2.

2. The signal measured at the 'B' receiver is leakage through various paths in the analyzer.

3. This isolation error is mathematically removed from subsequent transmission measurements.

**Source Match Error**

Ideally in reflection measurements, all of the signal that is reflected off of the DUT is measured at the 'A' receiver.

In reality, some of the signal reflects off the DUT, and multiple internal reflections occur between the analyzer and the DUT. These reflections combine with the incident signal and are measured at the 'A' receiver, but not at the 'R' receiver.

This measurement error is called source match error which can be characterized and reduced by the analyzer.

**How the Analyzer Measures and Reduces Source Match Error**

1. During calibration, all reflection standards are connected to Port 1. Known reflections from the standards are measured at the 'A' receiver.

2. Complex math is used to calculate source match error.

3. Source match error is mathematically removed from subsequent reflection and transmission measurements.

**Load Match Error**

Ideally in transmission measurements, an incident signal is transmitted through the DUT and is measured at the 'B' receiver.

In reality, some of the signal is reflected off of Port 2 and other components and is not measured at the
'B' receiver.

This measurement error is called load match error which can be characterized and reduced by the analyzer.

**How the Analyzer Measures and Reduces Load Match Error**

1. The Port 1 and Port 2 test connectors are mated together for a perfect zero-length thru connection. If this is not possible, a characterized thru adapter is inserted. This allows a known amount of incident signal at Port 2.

2. The signal measured at the 'A' receiver is reflection signal off of Port 2.

3. The resulting load match error is mathematically removed from subsequent transmission and reflection measurements.

**Frequency Response Reflection Tracking Error**

Reflection measurements are made by comparing signal at the 'A' receiver to signal at the 'R1' receiver. This is called a ratio measurement or "A over R1" (A/R1).

For ideal reflection measurements, the frequency response of the 'A' and 'R1' receivers would be identical.

In reality, they are not, causing a frequency response reflection tracking error. This is the vector sum of all test variations in which magnitude and phase change as a function of frequency. This includes variations contributed by:

- signal-separation devices
- test cables
- adapters
- variations between the reference and test signal paths
Frequency response reflection tracking error can be characterized and reduced by the analyzer.

How the Analyzer Measures and Reduces Frequency Response Reflection Tracking Error.

1. During calibration, all reflection standards are used to determine reflection tracking.
2. The average 'A' receiver response is compared with the 'R1' receiver response.
3. Complex math is used to calculate Frequency Response Reflection Tracking Error (see the following diagram). This frequency response reflection tracking error is mathematically removed from subsequent DUT measurements.

Note: In reflection response calibrations, only a single calibration standard is measured (open or short) and thus only its contribution to the error correction is used.

Frequency Response Transmission Tracking Error

Transmission measurements are made by comparing signal at the 'B' receiver to signal at the 'R1' receiver. This is called a ratio measurement or "B over R1" (B/R1).

For ideal transmission measurements, the frequency response of the 'B' and 'R1' receivers would be
identical.

In reality, they are not, causing a frequency response transmission tracking error. This is the vector sum of all test variations in which magnitude and phase change as a function of frequency. This includes variations contributed by:

- signal-separation devices
- test cables
- adapters
- variations between the reference and test signal paths

Frequency response transmission tracking error can be characterized and reduced by the analyzer.

**How the Analyzer Measures and Reduces Frequency Response Transmission Tracking Error.**

1. During calibration, the Port 1 and Port 2 test connectors are mated together for a perfect zero-length thru connection. If this is not possible, a characterized thru adapter is inserted. This allows a known amount of incident signal to reach Port 2.

2. Measurements are made at the 'B' and 'R1' receivers.

3. Complex math is used to calculate Frequency Response Transmission Tracking Error (see the following diagram). This frequency response transmission tracking error is mathematically removed from subsequent DUT measurements.
3-Port Error Terms

The following flow diagram displays the 3-port error term model:

where:

E = error term
DIR = Directivity
MAT = Forward Source Match and Reverse Load Match
TRK = Forward Reflection Tracking and Reverse Transmission Tracking

4-Port error terms
A full 4-port calibration requires the following terms:

**Learn about the port numbering convention** for error terms.

<table>
<thead>
<tr>
<th>Source Port</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DIR 1,1</td>
<td>RTRK 1,1</td>
<td>SRM 1,1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LDM 2,1</td>
<td>TTRK 2,1</td>
<td>SRM 2,1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LDM 3,1</td>
<td>TTRK 3,1</td>
<td>XTLK 3,1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LDM 4,1</td>
<td>TTRK 4,1</td>
<td>XTLK 4,1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>DIR 2,2</td>
<td>LDM 2,3</td>
<td>LDM 2,4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RTRK 2,2</td>
<td>TTRK 2,3</td>
<td>TTRK 2,4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SRM 2,2</td>
<td>XTLK 2,3</td>
<td>XTLK 2,4</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>DIR 3,3</td>
<td>LDM 3,4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RTRK 3,3</td>
<td>TTRK 3,4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SRM 3,3</td>
<td>XTLK 3,4</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>DIR 4,4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RTRK 4,4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SRM 4,4</td>
</tr>
</tbody>
</table>

**Reflection terms**

- **DIR**: Directivity
How can we measure only 3 THRU connections?

On a 4-port VNA, a full 4-port cal can be performed while measuring only 3 THRU connections. Measuring more than 3 THRU connections on a VNA with four native ports can give higher accuracy under some conditions.

By measuring all of the reflection terms, and 3 transmission THRU connections, there is adequate information available to calculate the remaining transmission terms. The following is a high level explanation of the concept. The actual calculations are much more complex.

To simplify, let's substitute letters (A,B,C,D) for port numbers from the diagram above so that they can be combined without confusion. Also for simplicity, let's assume that the source match and directivity errors are zero.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>AA</td>
<td>AB</td>
<td>AC</td>
<td>AD</td>
</tr>
<tr>
<td>B</td>
<td>BA</td>
<td>BB</td>
<td>BC</td>
<td>BD</td>
</tr>
<tr>
<td>C</td>
<td>CA</td>
<td>CB</td>
<td>CC</td>
<td>CD</td>
</tr>
<tr>
<td>D</td>
<td>DA</td>
<td>DB</td>
<td>DC</td>
<td>DD</td>
</tr>
</tbody>
</table>

- The reflection errors are all measured (AA, BB, CC, DD).
- Let's assume we measure a THRU between ports AB, AC, AD. The reverse direction for these THRUs are also measured at the same time (BA, CA, DA).
- The terms left to calculate are BC, CB, BD, DB, CD, DC.

The following shows how the BC term is calculated from BA and AC:

\[
\frac{BA \times AC}{AA} = \frac{B \times A \times C}{AA} = BC
\]
Similarly:

- CB is calculated from CA and AB
- BD is calculated from BA and AD
- DB is calculated from AB and DA
- CD is calculated from CA and AD
- DC is calculated from DA and AC

**Monitoring Error Terms using Cal Set Viewer**

You can use **Cal Set Viewer** to monitor the measured data and the calculated error term. This will help to determine the health of your VNA and the accuracy of your measurements.

By printing or saving the error terms, you can periodically compare current error terms with previously recorded error terms that have been generated by the same VNA, measurement setup, and calibration kit. If previously generated values are not available, refer to Typical Error Term Data in Appendix A, "Error Terms", of the Service Guide.


- A stable system should generate repeatable error terms over about six months.
- A sudden shift in error terms over the same frequency range, power, and receiver settings, may indicate the need for troubleshooting system components. For information on troubleshooting error terms, see Appendix A, "Error Terms", of the Service Guide.
- A subtle, long-term shift in error terms often reflects drift or connector and cable wear. The cure is often as simple as cleaning and gauging connectors or inspecting cables.

**Viewing Cal Set Data**

- Existing measurement traces are unaffected by the Cal Set Viewer.
- The Cal Set data trace is presented in the highest unused channel number (usually 32) in the active window.
- The Cal Set data trace is labeled as S11 in the status bar regardless of the type of error term or standard.
- Only one Cal Set error term or standard data can be viewed at a time. However, a data trace can be stored into memory and then compared to other data traces.

*See the error terms equations.*
How to access Cal Set Viewer

Using Hardkey/SoftTab/Softkey

1. Press *Cal > Cal Sets & Cal Kits > Cal Set Viewer ON|OFF*

How to use Cal Set Viewer

1. Use the down arrow to select a Cal Set. Then click either:
   - **Error Terms** - calculated data.
   - **Standards** - the raw measurement data of the Standard. **ONLY** available with Unguided Cal (not ECal or Guided Cal).

2. Use the down arrow to select an error term or standard to view.

3. Select the **Enable** check box to view the data on the VNA screen.

**Port numbering convention** for error terms is the same as for S-Parameters:

**E Term (Receiver, Source)** with the following exceptions:

- Load Match (2,1) - The match of port 2 which is measured by making an S11 measurement.
- Load Match (1,2) - The match of port 1 which is measured by making an S22 measurement.
- Transmission Tracking (2,1) - The port 2 receiver relative to the port 1 reference. (source=port 1).
- Transmission Tracking (1,2) - The port 1 receiver relative to the port 2 reference. (source=port 2).
- And so forth for multiport calibrations.
Calibration accuracy is affected by the type of calibration, quality of the calibration standards, and the care with which the calibration is performed. This section provides additional information about how to make accurate calibrations.

- Measurement Reference Plane
- Effects of Using Wrong Calibration Standards
- Data-based versus Polynomial Calibration Kits
- Accuracy Level of Interpolated Measurement
- Effects of Power Level
- Using Port Extensions
- Isolation Portion of 2-Port Calibration
- Choosing a Thru Method

Learn how to determine the validity of your calibration.

**See other Calibration Topics**

### Measurement Reference Plane

Most measurement setups will NOT allow you to connect a device under test (DUT) directly to the analyzer front panel test ports. More likely, you would connect your device to test fixtures, adapters, or cables that are connected to the analyzer.

A calibration takes place at the points where calibration standards are connected during the calibration process. This is called the measurement reference plane (see graphic). For the highest measurement accuracy, make the calibration reference plane the place where your DUT is connected. When this occurs, the errors associated with the test setup (cables, test fixtures, and adapters used between the analyzer ports and the reference plane) are measured and removed in the calibration process.
Effects of Using Wrong Calibration Standards

Normally, a calibration is performed using a calibration kit that contains standards with connectors of the same type and sex as your DUT.

However, your calibration kit may not always have the same connector type and gender as your device. For example, suppose your device has 3.5mm connectors, but you have a Type-N calibration kit. If you use an adapter to connect the Type-N standards to the 3.5mm test port, then the adapter becomes part of the calibration and NOT part of the test setup. This will result in significant errors in your reflection measurements.

Data-based versus Polynomial Calibration Kits

The Select DUT Connectors and Cal Kits dialog box offers a data-based model and a polynomial model for the newest high-frequency cal kits. See Analyzer Accessories. The data-based models provide higher accuracy for describing calibration standards than the polynomial models. It is RECOMMENDED that the data-based model be used if the most accurate results are desired.

<table>
<thead>
<tr>
<th></th>
<th>Data-Based Model</th>
<th>Polynomial Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>How accurate is the model?</td>
<td>Provides highest calibration accuracy. Eliminates the errors that can be the result of polynomial model approximations.</td>
<td>Provides high calibration accuracy.</td>
</tr>
<tr>
<td>How does the model define calibration standards?</td>
<td>Uses S-Parameter measurements.</td>
<td>Uses traditional four-term polynomial calibration standard modeling parameters.</td>
</tr>
<tr>
<td>How do I manually edit the definitions of the calibration standards when using the model?</td>
<td>Use the Advanced Modify Cal Kit function.</td>
<td>Use the Advanced Modify Cal Kit function.</td>
</tr>
<tr>
<td>How do I use the Calibration Wizard with the model?</td>
<td>Use only the SmartCal (Guided) Calibration method.</td>
<td>Use the SmartCal (Guided) or the Unguided Mechanical Calibration methods.</td>
</tr>
</tbody>
</table>

Learn about the “Expanded Math” feature.

**Effects of Power Level**

To attain the most accurate error correction, do NOT change the power level after a calibration is performed. However, when changing power within the same attenuator range at which the measurement calibration was performed, S-parameter measurements can be made with only a small degradation of accuracy. If a different attenuator range is selected, the accuracy of error correction is further degraded.

A best practice is to calibrate so that the receiver power is the desired power you want to calibrate at when you are making a measurement. For example, if you are testing an amplifier with 20 dB of gain, and a max input of -20 dBm, then calibrate at 0 dBm (it is the power that will hit the receiver when you are testing). After the calibration, lower the source power to measure your amplifier. In this way you will have less noise in the calibration. The noise degradation by calibrating at a lower power will completely swamp the small dynamic accuracy error. For most of the time, the uncertainty calculator assumes 10 Hz BW which does effectively eliminate the noise from the uncertainty computation, but it is rarely used in practice. And, note the R receiver is more linear (by design, it has extra padding in it) compared to the test receivers so the dynamic accuracy or linearity error is more substantial on the test receivers (B receiver for example). Therefore, calibrating at near its target power will reduce the dynamic error at the test receiver. The reference receiver has almost no error at all. In fact, it is so small it cannot be measured, so the uncertainty calculator uses the specification value which is, in fact, limited by our linearity test system and not the receiver. The Reference channel linearity is less than 0.015 dB from -120 dBm to about 0 dBm, but the specification is probably 3 times higher.

To check the accuracy of a calibration, see [Validity of a Calibration](#).

**Using Port Extensions**

Use the port extensions feature after calibration to compensate for phase shift of an extended measurement reference plane due to additions such as cables, adapters, or fixtures.

Port extensions is the simplest method to compensate for phase shift, mismatch, and loss of the path between the calibration reference plane and the DUT.

Learn how to apply port extensions.

Learn about [characterizing a test fixture](#).
Isolation Portion of 2-Port Calibration

The isolation portion of a calibration corrects for crosstalk, the signal leakage between test ports when no device is present. When performing an UNGUIDED 2-port calibration, you have the option of omitting the isolation portion of the calibration.

**Note:** Isolation can be performed on a Smart (Guided) Calibration ONLY

**Note:** Isolation is supported only when the receiver gain are set at LOW for both ports.

The uncorrected isolation between the test ports of the analyzer is exceptional (typically >100 dB). Therefore, you should only perform the Isolation portion of a 2-port calibration when you require isolation that is better than 100 dB. Perform an isolation calibration when you are testing a device with high insertion loss, such as some filter stopbands or a switch in the open position.

The isolation calibration can add noise to the error model when the measurement is very close to the noise floor of the analyzer. To improve measurement accuracy, set a narrow IF Bandwidth.

**How to perform an Isolation Calibration**

Isolation is measured when the Load standards are connected to the analyzer test ports. For best accuracy, connect Load standards to BOTH test ports each time you are prompted to connect a load standard. If two Loads are not available, connect the untested analyzer port to any device that will present a good match.

**Important Notes:**

1. For best results, the average factor should be increased while measuring the isolation standards. Doing an isolation without increasing averaging you are generally adding noise to the measurement rather than correcting for isolation. This PNA automatically increases the average factor by 8 during the isolation steps of a calibration to minimize the noise degradation that would otherwise be present.
2. The PNA crosstalk is generally below the noise floor which means if you do an isolation calibration is usually not required.
3. SmartCal did not add the ability to do an isolation calibration from the GUI because it is usually only needed in special cases. The isolation calibration can be done using the following commands:
   - `SENSe<ch>:CORRection:COLLect:GUIDed:ISOLation:AVERage:INCRement` or `IsolationAveragingIncrement`
   - `SENSe<ch>:CORRection:COLLect:GUIDed:ISOLation:PATHs` or `GetIsolationPaths` and `SetIsolationPaths`
4. Doing an isolation calibration is generally only useful when crosstalk is an issue-this may happen if the fixture added contributes significant crosstalk.

**Choosing a Thru Method**

When calibrating for a non-insertable device, you must choose a method to calibrate for the THRU
error terms. This can have a significant effect on measurement accuracy. Learn more about choosing a thru method.
Validity of a Calibration

This section helps you determine if your calibration is valid and how the analyzer displays correction level information for your measurement.

- Frequency Response of Calibration Standards
- Validating a Calibration
- Quick Check
- ECal Confidence Check
- Determining Effects of Not Terminating Unused ECal Ports
- Verification Kit

See other Calibration Topics

Frequency Response of Calibration Standards

In order for the response of a calibration standard to show as a dot on the smith chart display format, it must have no phase delay with respect to frequency. The only standards that exhibit such "perfect" response are the following:

- 7-mm short (with no offset)
- Type-N male short (with no offset)

There are two reasons why other types of calibration standards show phase delay after calibration:

1. The reference plane of the standard is electrically offset from the mating plane of the test port. Such devices exhibit the properties of a small length of transmission line, including a certain amount of phase shift.

2. The standard is an open termination, which by definition exhibits a certain amount of fringe capacitance and therefore phase shift. Open terminations which are offset from the mating plane will exhibit a phase shift due to the offset in addition to the phase shift caused by the fringe capacitance.

The most important point to remember is that all standards are measured in order to remove systematic errors from subsequent device measurements. As a result, if calibration standards with
delay and fringe capacitance are measured as a device after a calibration, they will NOT appear to be "perfect". This is an indication that your analyzer is calibrated accurately and working properly.

Validating a Calibration

At the completion of a calibration or selection of a stored Cal Set, validation can accomplish the following:

**Improve Measurement Accuracy** – Once a measurement calibration has been performed, its performance should be checked before making device measurements. There are several sources of error that can invalidate a calibration: bad cables, dirty or worn calibration standards that no longer behave like the modeled standards, and operator error.

**Verify Accuracy of Interpolation** – You should validate the calibration if you are testing a device and the measurements are uncertain because of interpolation. For more information see Interpolation Accuracy.

**Verify Accuracy of Cal Standards** – To check accuracy, a device with a known magnitude and phase response should be measured.

Quick Check

For this test, all you need are a few calibration standards. The device used should not be one of the calibration standards; a measurement of one of these standards is merely a measure of repeatability.

The following reflection and transmission Quick Check tests can be applied to all test ports.

**To verify reflection measurements, perform the following steps:**

1. Connect either an OPEN or SHORT standard to port 1. The magnitude of S11 should be close to 0 dB (within a few tenths of a dB).

2. Connect a load calibration standard to port 1. The magnitude of S11 should be less than the specified calibrated directivity of the analyzer (typically less than -30 dB).

**To verify transmission measurements:**

1. Connect a THRU cable (or known device representative of your measurement) from port 1 to port 2. Verify the loss characteristics are equivalent to the known performance of the cable or device.

2. To verify S21 isolation, connect two loads: one on port 1 and one on port 2. Measure the magnitude of S21 and verify that it is less than the specified isolation (typically less than -80 dB).
**Note:** To get a more accurate range of expected values for these measurements, consult the analyzer's specifications.

**ECal Confidence Check**

ECal Confidence Check is a method to check the accuracy of a calibration performed with mechanical standards or an ECal module. The confidence check allows you to measure an impedance state in the ECal module (called the confidence state), and compare it with factory measured data stored in the module.

In order for this test to be valid, the test ports of the ECal module must connect directly to the calibration reference plane (without adapters).

**Note:** In the N469x series of 2-port ECal modules, from the module minimum frequency up to approximately 2 GHz, the confidence state has a very high amount of transmission loss. In this frequency range, calibrated measurements of transmission S-parameters for the confidence state may vary much more than expected from the Keysight-characterized data in the measurement memory trace. When comparing the measurement trace and memory trace you, ignore the data for frequencies up to 2 GHz.

**How to Perform ECal Confidence Check:**

1. Connect ECal module to the analyzer with the USB cable. See Connect ECal Module to the VNA. **Note:** Terminate any unused ECAL ports with a 50 ohm load. See below.

2. Allow the module to warm up for 15 minutes or until the module indicates READY.

   **Note:** N755xA series ECal modules do not require a warm-up period.

3. Do one of the following to start ECal Confidence Check

   **Using Hardkey/SoftTab/Softkey**

   1. Press Cal > Cal Sets & Cal Kits > ECal > ECal Confidence Check...

On the following **ECal Confidence Check dialog box**:

2. Click **Read Module Data**. The following occurs:

   - ECal module is set to "confidence state".
Analyzer reads and displays stored data.
Analyzer measures and displays "confidence state".

3. To view a different parameter, select Change Measurement and select the check box for the desired parameter. The default is the active channel parameter.

4. Select the viewing option in the Trace View Options block.

5. Compare the stored and measured data for each measurement parameter.

Notes:

- After exiting ECal Confidence Check, the ECal module remains in the same impedance state and the factory (or user-characterized) data is still stored in the memory trace. Therefore, you can save both the data and memory trace as a *.csv files and import them to a spreadsheet. Learn how.

- If the two traces show excessive difference, there may be a loose or dirty connection at the test ports or damage to the test cables. Carefully inspect the cables and connections. Then clean and gage each connector, and re-calibrate if needed.

- The User Characterization setting selects the user-characterization data instead of the factory characterization data (available when a User-Characterization is stored in the ECal module).

Determining Effects of Not Terminating Unused ECal Ports

The following procedure can be used to determine the calibration errors when unused ECal ports are not terminated.

1. Connect the ECal module to one VNA test port through an adapter (eliminates the possibility of cable movement errors being included) and leave the unconnected port(s) open.

2. Perform a 1-port cal (use 100 Hz IFBW) then save the calset.

3. Connect a load(s) to the unconnected port(s) of the ECal module.

4. Perform a 1-port cal then save the calset.

5. Compare the calset in step 2 with the calset in step 4 using Calset Viewer to evaluate the effect of leaving the unused ECal ports open.

ECal Confidence Check dialog box help
Compares the accuracy of corrected (calibrated) data with stored data in the ECal module. For the check to be valid, the module test ports must connect directly to the calibration reference plane (without an adapter). Learn more about ECal Confidence Check.

**Measurement**

**Change Measurement** Opens the Measure dialog box.

**Use ECal Module**

**Read Module Data**

- Copies stored data from the ECal module to Memory.
- Changes state of ECal module to confidence state.
- Measures and displays confidence state and Memory trace.
- Displays the factory and user characterizations data stored in the ECal module. Learn more.

**Scale** Opens the Scale dialog box.

**Show Prompts** Check to show a reminder for the connection (default).

**Trace View Options**

**Data and Memory Trace** Displays current measurement data and Memory trace.

**Data / Memory** Performs an operation where the current measurement data is divided by the data in memory.
**Data + Memory** Performs an operation where the current measurement data is added to the data in memory.

---

**Verification Kit**

Measuring known devices, other than calibration standards, is a straightforward way of verifying that the network analyzer system is operating properly. Verification kits use accurately known verification standards with well-defined magnitude and phase response. These kits include precision airlines, mismatch airlines, and precision fixed attenuators. Traceable measurement data is shipped with each kit on disk and verification kits may be re-certified by Keysight.

See Analyzer Accessories for a list of Keysight verification kits.
About Calibration Kits

A calibration kit contains a set of physical devices called standards. Each standard has a precisely known or predictable magnitude and phase response as a function of frequency. All Keysight Cal Kits and their standard definitions are stored in the analyzer. For a list of Keysight calibration kits, see Analyzer Accessories.

Calibration Standards

Calibration standards provide the reference for error-corrected measurements in the network analyzer. Each standard has a precisely known definition that includes electrical delay, impedance, and loss. The analyzer stores these definitions and uses them to calculate error correction terms.

During measurement calibration, the analyzer measures standards and mathematically compares the results with the definitions ("ideal models") of those standards. The differences are separated into error terms that are later removed from device measurements during error correction. See Systematic Errors.

Standard Type

A standard type is one of four basic types that define the form or structure of the model to be used with that standard. The following are the four basic standard types:
Standard Terminal Impedance

<table>
<thead>
<tr>
<th>Standard</th>
<th>Terminal Impedance</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHORT</td>
<td>zero ohms</td>
</tr>
<tr>
<td>OPEN</td>
<td>infinite ohms</td>
</tr>
<tr>
<td>LOAD</td>
<td>system impedance, Z0</td>
</tr>
<tr>
<td>THRU/LINE</td>
<td>no terminal impedance</td>
</tr>
</tbody>
</table>

Learn about other Calibration Standards:

- Data-Based Standard
- Sliding Load
- Offset Load
- Arbitrary Impedance Load

Standard Definitions

Standard definitions describe the electrical characteristics of the standards and the frequencies they will be used. Standard definitions can be viewed from the Advanced Modify Cal Kit menu selection. Standard definitions include:

- **Minimum Frequency**  Specifies the minimum frequency the standard is used for calibration.
- **Maximum Frequency**  Specifies the maximum frequency the standard is used for calibration.
- **Z0**  Specifies the characteristic impedance of the standard (not the system characteristic impedance or the terminal impedance of the standard).
- **Delay**  Specifies a uniform length of transmission line between the standard being defined and the actual calibration plane.
- **Type**  Specifies type of standard (SHORT, OPEN, THRU/LINE, LOAD, ARBITRARY).
- **Loss**  Specifies energy loss, due to skin effect, along a one-way length of coaxial cable.

Loss model equation:

- The value of loss is entered as ohms/second at 1 GHz.
- To compute the loss of the standard, measure the delay in seconds and the loss in dB at 1 GHz. Then use the following formula:
Capacitance model equation:

**C0, C1, C2, C3.** Specifies the fringing capacitance for the open standard.

- $C = (C0) + (C1 \times F) + (C2 \times F^2) + (C3 \times F^3)$
- $(F$ is the measurement frequency).

  The terms in the equation are defined when specifying the open as follows:
  - **C0** term is the constant term of the third-order polynomial and is expressed in Farads.
  - **C1** term is expressed in F/Hz (Farads/Hz).
  - **C2** term is expressed in F/Hz².
  - **C3** term is expressed in F/Hz³.

Inductance model equation:

**L0, L1, L2, L3.** Specifies the residual inductance for the short standard.

- $L = (L0) + (L1 \times F) + (L2 \times F^2) + (L3 \times F^3)$
- $(F$ is the measurement frequency).

  The terms in the equation are defined when specifying the short as follows:
  - **L0** term is the constant term of the third-order polynomial and is expressed in Henries.
  - **L1** term is expressed in H/Hz (Henries/Hz)
  - **L2** term is expressed in H/Hz².
  - **L3** term is expressed in H/Hz³.

Class Assignments

Once a standard is characterized, it must be assigned to a standard "class". A standard class is a group of standards that are organized according to the calibration of the network analyzer error model.

The number of classes needed for a particular calibration type is equal to the number of error terms being corrected.
A class often consists of a single standard, but may be composed of multiple standards. These may be required for accuracy or to cover a wide frequency range.

**Example:** A response calibration requires only one class, and the standards for that class may include an OPEN, or SHORT, or THRU. A 1-port calibration requires three classes. A 2-port calibration requires 10 classes, not including two for isolation.

The number of standards assigned to a given class may vary from one to seven for unguided calibrations. Guided calibrations allow as many standards as needed.

Calibration Classes are assigned in the Advanced Modify Cal Kit menu, SOLT or TRL tab.

### The different classes used in the analyzer

**S11A, S11B, S11C (S22A, S22B, S22C and so forth)**

These are the three classes for port 1-reflection calibrations (three classes also for S22 and S33). They are used in the one-port calibrations and the full two-port calibration. They are required in removing the directivity, source match, and reflection tracking errors. Typically, these classes might consist of an open, a short and a load standard for each port.

**Transmission and Match (forward and reverse)**

These classes are used to perform a full two-port calibration. The transmission class relates primarily to the transmission tracking, while the match class refers to load match. For both of these classes, the typical standard is a thru or delay.

**Isolation**

The isolation classes are used to perform a full two-port and the TRL two-port calibrations. The isolation classes apply to the forward and reverse crosstalk terms in the network analyzer error model.

**TRL THRU**

These are used to perform a TRL two-port calibration. The TRL thru class should contain a thru standard or a short line. If it contains a non-zero length thru standard, then the calibration type is called LRL or LRM.

**TRL REFLECT**

This class is used to perform a TRL two-port calibration. The TRL reflect class should contain a standard with a high reflection coefficient, typically an open or short. The actual reflection coefficient need not be known, but its phase angle should be specified approximately correctly (± 90 deg). The exact same reflection standard must be used on both ports in the TRL calibration process.

**TRL LINE or MATCH**
These are used to perform a TRL two-port calibration. The TRL line or match class should contain line standards, load standards, or both. If a line standard is used, its phase shift must differ from that of the TRL THRU standard by 20° to 160°. This limits the useable frequency range to about 8 to 1. Two or more line standards of different lengths may be specified to get broader frequency coverage. It is also common to include a load standard for covering low frequencies, where the line's length would be impractically long. When a load is used, the calibration type is called TRM or LRM.

**Note:** For more information, read *Specifying Calibration Standards and Kits for Keysight Vector Network Analyzers (Application Note 1287-11)*
Modify Calibration Kits

The following topics discuss Modifying Calibration Kits:

In this Topic

- How to Modify Cal Kits
- Manage Cal Kits dialog
- Cal Kits and Firmware Upgrades
- Import Kit dialog

Using VNA CalKit Editor

- Connectors Tab
- Standards Tab
- SOLT Tab
- TRL Tab

Concepts

- Why Modify a Cal Kit
- VNA Cal Kit File Types

Procedures

- How to Create a New Cal Kit from an Existing Cal Kit
- Creating Custom Calibration Kits using a New Connector Family
- Noise Figure and TRL Cal (separate topic)
How to Modify Cal Kits

The series of dialog boxes that follow allow you to modify the standard definitions or class assignments of calibration kit files.

**Using Hardkey/SoftTab/Softkey**

1. Press Cal > Cal Sets & Cal Kits > Cal Kit...

**Using a mouse**

1. Click Response
2. Select Cal
3. Select Cal Sets & Cal Kits
4. Select Cal Kit...

Manage Cal Kits

The Manage Cal Kits dialog allows you to define the cal kits in the active workspace. These cal kits may be edited, created, or saved to a cal kit file.
Cal Kits and Firmware Upgrades

- If the firmware upgrade includes factory cal kits that are formatted differently than the factory cal kit files in the instrument, then:
  
  - The factory cal kit files will be overwritten.
  - The custom cal kit files will not be changed.
  - A backup of the active workspace will be saved.
  - A new active workspace will be created and will contain only the new factory cal kit definitions.
  - Under these conditions, if you want to use your custom cal kit definitions, you will need to import these files into the active workspace. Learn how to import cal kit files.

Uncertainty Cal Kits (Only for PNA)

Uncertainty Cal Kits... Opens the Uncertainty Cal Kit Manager (Option S93015A/B enabled). Learn more about Dynamic Uncertainty.

Standard Cal Kits

This group box lists all cal kits in the active workspace.

Edit... Starts the Connectors tab of the Edit Kit dialog box to modify selected calibration kit definitions.

Import... Starts the Import Kit dialog box to add a cal kit definition from a file into the active workspace.

Save... Saves the selected calibration kit definition into a cal kit file (using .xkt, .ckt, or .prn file type). See VNA Cal Kit File Types.

Insert... Starts a blank Edit Kit dialog box to create a new calibration kit.

Delete Deletes selected calibration kit from the active workspace.

Print... Prints the contents of the selected cal kit to a .prn file.

Cal Kit Workspace

The active cal kit workspace is a collection of standard cal kits that are accessible by the VNA for calibrations.
Save Workspace... Saves the active workspace to a workspace file (*.xkw or *.wks).

Recall Workspace... Recalls a workspace file into the active workspace.

Restore Defaults Restores the active workspace and the factory cal kit files (*.xkt) to their factory default definitions.

For more information see Creating Custom Calibration Kits using a New Connector Family.

**Import Cal Kit dialog box help**

**Open**

![Image of Import Cal Kit dialog box]

**Files of type** Select the file type of your Cal Kit. Learn more about VNA Cal Kit File Types

**File name** Navigate and select your cal kit file name.

**Open** Loads the selected file into the active cal kit workspace.

**Note:** There is no limit to the number of cal kits that can be imported. However, during an Unguided cal, you can access ONLY mechanical cal kits #1 through 95.

Imports a cal kit file into the active cal kit workspace.

**Importing Cal Kits from "legacy" network analyzers**

Cal kit files from "legacy" network analyzers (such as the 8510 or 8753) may not contain
information that this VNA requires. Therefore, this VNA may modify the cal kit name, description, standards, and class assignments. You may need to correct these modifications after importing your legacy cal kit to meet your specific requirements.

- "Legacy" cal kit files are referenced to the VNA test port gender while modern cal kit files are referenced to the Device Under Test (DUT) connector gender. Therefore, when a legacy cal kit is imported, the genders of the standards in the legacy cal kit will be automatically reversed in the new cal kit.

- Legacy cal kits do not contain connector definitions. If a coaxial legacy kit is imported, then male and female coax connector definitions will be added to the kit. If a waveguide legacy kit is imported, then a genderless waveguide connector definition will be added to the kit.
For most applications, the default calibration kit models provide sufficient accuracy for your calibration. However, several situations may exist that would require you to create a custom calibration kit:

- Using a connector interface different from those used in the predefined calibration kit models.
- Using standards (or combinations of standards) that are different from the predefined calibration kits. For example, using three offset SHORTs instead of an OPEN, SHORT, and LOAD to perform a 1-port calibration.
- Improving the accuracy of the models for predefined kits. When the model describes the actual performance of the standard, the calibration is more accurate. For example: A 7 mm LOAD is determined to be 50.4 \( \Omega \) instead of 50.0 \( \Omega \).
- Modifying the THRU definition when performing a calibration for a non-insertable device.
- Performing a TRL calibration.
Creating a New Connector Family

To create a custom calibration kit that uses a new connector type, you must first define the connector family. The connector family is the name of the connector-type of the calibration kit, such as:

- APC7
- 2.4 mm
- Type-N (50Ω)

Although more than one connector family is allowed, it is best to limit each calibration kit to only one connector family. One exception to this is if specific thru's are required on a per-port-pair basis (see the following note).

**Note:** One way to accomplish assigning specific thru’s to specific port pairs is to define connectors on a per-port basis. For example, if it is desired for THRU1 to be used between Ports 1 and 2, and THRU2 to be used, then you can set up connectors “2.4mm Port1”, “2.4mm Port2”, “2.4mm Port3”. And then in the standards definition, create THRU1 standard with connectors “2.4mm Port1” and “2.4mm Port2” and THRU2 standard with “2.4mm Port2” and “2.4mm Port3”.

If you are using a connector family that has male and female connectors, include definitions of both genders. If you are using a family with no gender, such as APC7, only one connector definition is required.

Use the following steps to create a custom calibration kit:

1. Press **Cal > Cal Sets & Cal Kits > Cal Kit....**
2. Click **Edit....**
3. In the **Connectors Tab**, click **Add** to name the new connector family.
4. Enter the Kit Description for the custom cal kit.
5. Click **Add** in the Connectors section of the dialog box.
6. Enter a Connector Family name.
7. Enter a Description of the connector.
8. Select the Gender of one of the connectors.
9. Enter the minimum and maximum Frequency Range.
10. Enter the Impedance.
11. Click the down-arrow to select the Media.
12. Enter the cut-off frequency
13. Click **Apply**.
14. Click **OK**.
15. If you need to add another connector gender, in the **Connectors Tab**, click **Add** in the Connectors section again for the next connector gender.
16. If you are adding another connector gender, repeat step 3.

**Note**: If you have male and female versions of the connector family, you probably do NOT also have a NO GENDER version.

### Enter Standards

Now that the connector family is added to the custom cal kit, you are ready to add new calibration standards.

1. In the **Standards Tab**, under the list of standards, click **Add**.
2. Select the type of standard (OPEN, SHORT, LOAD, or THRU), then click **OK**.
3. Complete the information in the dialog box for the standard you selected. Note that for banded standards, the start and stop frequency may be different than the frequency range of the specified connector. Edit the start and stop frequencies as needed. Click **OK** when all the settings are correct.
4. Repeat steps 2 - 3, as necessary, to add all standards and definitions to the new custom cal kit.
5. Assign each of the standards to a calibration class. This is done through the **TRL Tab** or **SOLT Tab**
6. Save the Cal Kit.
How to Create a New Cal Kit from an Existing Cal Kit

You can create a new custom Cal Kit using an existing Cal Kit as a starting point.

Here is how:

1. Press Cal > Cal Sets & Cal Kits > Cal Kit....
2. Immediately click Save... and change the file name. Select either *.*kt or *.ckt, *.prn file type. Learn more about these file types.
3. Make modifications to your new custom Cal Kit as required.
4. Routinely save your work by clicking Save.

See Also

About VNA Cal Kits and Firmware Upgrades
VNA Cal Kit File Types

The VNA Cal Kit editor can open the following types of Cal Kit files:

<table>
<thead>
<tr>
<th>VNA Families</th>
<th>File Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal Kits supported by current firmware of these VNA models: PNA Series, E5080A/B, FieldFox, and PXI Series</td>
<td>* .xkt</td>
</tr>
<tr>
<td>Old PNA Series Cal Kits (PNA Firmware A.07.50 to A.09.90)</td>
<td>* .ckt</td>
</tr>
<tr>
<td>Old PNA Series Cal Kit (before A.07.50)</td>
<td>* .ck1</td>
</tr>
<tr>
<td>Previous FieldFox format Cal Kits</td>
<td>* .xml</td>
</tr>
<tr>
<td>Previous ENA format Cal Kits</td>
<td>* .ckx</td>
</tr>
<tr>
<td>8510 Cal Kit</td>
<td>.CK_ *</td>
</tr>
<tr>
<td>8753, 8752, 8719, 8720, or 8722 Cal Kit</td>
<td>* .ck</td>
</tr>
</tbody>
</table>

The current revision of Cal Kit files can be downloaded at
http://na.support.keysight.com/pna/caldefs/stddefs.html

File Save (As)

The VNA Cal Kit Editor can save Cal Kits in one of three file formats:

- * .xkt - Newer format that is based on xml and is shared among VNA families.
- * .ckt - VNA binary format, provided for backwards compatibility with older VNA firmware revisions and may not support future new cal kit capabilities, which is expected of the * .xkt format.
- * .prn - Cal kit print files. This is a text file format which can be read into spreadsheets, but the Cal Kit Editor does not read-in these files. These files are only produced as a form of documentation.

About Opening Legacy VNA Kits

Cal kit files from Keysight "legacy" network analyzers (listed above) may not contain information that the VNA requires. When loaded into the VNA Cal Kit Editor, the cal kit name and description, the cal standards, and the cal class assignments will be modified in a best effort manner. You may need to correct these modifications after importing your legacy cal kit to meet your specific requirements.

- "Legacy" cal kit files are based on the analyzer test port sex. Modern VNA cal kits are based on the Device
Under Test (DUT) connector sex. Therefore, when the kit is imported the standard's label and description are reversed and are noted as F- (female) and M- (male).

- When a Coaxial standard is detected in the kit file, a pair of male/female connectors is typically created.
- Waveguide standards that are created as connector have no gender.

**File Association**

With the exception of *.xml, the above file types are automatically associated with the CalKit Editor if they are not already associated with a different program. That means, after running CalKit Editor, double-clicking any of the above file types (except *.xml) will open the file using CalKit Editor.

If you have already associated one of these file types with a different program and would like to change it to CalKit Editor, do the following:

1. Right-click the file, then click **Open With**
2. Browse to the CalKitEditor install folder.
   - C:\Program Files (x86)\Agilent\VNA Cal Kit Editor
3. Check **Always use the selected program to open this kind of file.**
4. Select **CalKit Editor.**
**Connectors Tab**

- **Cal Kit Name**  Allows you to change the Name of the selected calibration kit.
- **Cal Kit Description**  Allows you to change the description of the selected calibration kit.

**Connector Family**  Click the down arrow to select the connector family associated with the Cal Kit.

**Add**  Starts the Add Connector dialog box which allows you to add new connector type to the calibration kit.

**Delete**  Deletes - WITHOUT WARNING - the selected connector family.

**Note:** To modify a connector family or name, Add a new connector, then delete the old connector.

The following is the list of Factory-defined connector type strings:
<table>
<thead>
<tr>
<th>Connector Family</th>
<th>Frequency</th>
<th>Gendered</th>
<th>Genderless</th>
</tr>
</thead>
<tbody>
<tr>
<td>APC 3.5 female</td>
<td>7-16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>APC 3.5 male</td>
<td>7-16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>APC 2.4 female</td>
<td>2.92 mm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>APC 2.4 male</td>
<td>2.92 mm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>APC 7</td>
<td>1.85 mm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type N (50) female</td>
<td>1.85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type N (75) female</td>
<td>1.85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type F (75) female</td>
<td>1.85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type F (75) male</td>
<td>1.85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type A (50) female</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type A (50) male</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type B</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Frequency Range**

**Min** Allows you to define the lowest frequency at which the standard is used for calibration.

**Max** Allows you to define the highest frequency at which the standard is used for calibration.

**Gender**

**Gendered** - The connector family contains both Male and Female connectors.

**Genderless** - The connector family does NOT contain Male and Female connectors. APC7 connectors are an example of this connector type.

**Impedance**

Specify the impedance of the standard.

**Media**

The medium (or 'geometry') of the connector (COAX or WAVEGUIDE).

**Cutoff Frequency** If Media is Waveguide, type the low-end cutoff frequency.

**Height/Width Ratio** Used to calculate waveguide loss. This value is usually on the data sheet for waveguide devices.

**About Waveguide Cal Kits**

If modifying or creating a waveguide cal kit, be sure to make the following settings. You can create a
custom waveguide cal kit using an existing factory waveguide Cal kit as a starting point. The factory cal kits already have these settings.

- Frequency Range: **Min. frequency = Cutoff frequency.**
- Gender: **No Gender**
- Impedance Z0: **1 ohm**
- Media: **Waveguide**

For waveguide, choose TRL (Thru-Reflect-Line) calibration type. These calibration types are more accurate and take fewer steps than SOLT.

**Add Connector Family**

Enter a name for the new connector family. Then click **OK**.

---

**Available at the bottom of every tab**

- **Save As** - Allows you to save the cal kit to a new file name and type.
- **Save** - Saves the cal kit to the same file name and type.
- **Close** - Closes the cal kit editing session. The file is NOT saved automatically.
Standards Tab

Allows you to Add, Edit or Delete cal standards in a cal kit.

**Add Standard (Open, Short, Load, Thru, or Data-based)**

Allows you to add standards to the calibration kit file.

Choose from:

- OPEN
- SHORT
- LOAD
- THRU
- DATA-BASED
- ISOLATION
Standards dialog box help

The following fields apply to ALL standard types:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard ID</td>
<td>Number in list of standards</td>
</tr>
<tr>
<td>Label</td>
<td>Type of standard. This usually appears in prompts for standards.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of standard.</td>
</tr>
</tbody>
</table>

The other areas of the dialog change depending on the type of standard selected.

Identification

- **Standard ID** Number in list of standards
- **Label** Type of standard. This usually appears in prompts for standards.
- **Description** Description of standard.

The following fields apply to ALL standard types EXCEPT the Isolation type:

Connector

Indicates the type and gender (Male, Female, None) of the standard.

Thru and Isolation standards have two connectors.

Data-Based standards MAY have two connectors.

Frequency Range

- **Min** Defines the lowest frequency at which the standard is used for calibration.
- **Max** Defines the highest frequency at which the standard is used for calibration.
The Delay Characteristics fields apply to MOST standard types:

**Delay Characteristics**

**Delay** Defines the one-way travel time from the calibration plane to the standard in seconds.

**Z0** Defines the impedance of the standard.

**Loss** Defines energy loss in Ohms, due to skin effect, along a one-way length of coaxial cable.

---

**Other fields are unique to standard type**

Choose from:

- OPEN
- SHORT
- LOAD
- THRU
- DATA-BASED
- ISOLATION

---

**Open Standard**

**C0, C1, C2, C3** Specifies the fringing capacitance.

These are the unique fields of the dialog. See the areas that are common to all standards.
Short Standard

L0, L1, L2, L3  Specifies the residual inductance.

Load Standard

Choose from the following

Fixed Load  Specifies the load type as Fixed. The fixed load is assumed to be a perfect termination without reflection.

Sliding Load

A sliding load is defined by making multiple measurements of the device with the sliding load element positioned at various marked positions of a long transmission line. The transmission line is assumed to have zero reflections and the load element has a finite reflection that can be mathematically removed using a least squares circle fitting method.

A sliding load cal can be very accurate when performed perfectly. It can also be very inaccurate when not using proper technique. For accurate results, closely follow the users manual instructions for the sliding load.

Arbitrary Impedance
Specifies the load type that has an impedance value different from system Z0. An arbitrary impedance device is similar to a fixed load except that the load impedance is NOT perfect. Early firmware releases of the VNA series used a fixed resistance value. A complex terminating impedance has been added to allow for more accurate modeling of circuit board or on-wafer devices.

The following Complex Impedance settings are available ONLY when Arbitrary Impedance is selected.

- **Real**  The real portion of the impedance value.
- **Imaginary**  The imaginary portion of the impedance value.

**Offset Load**

![Offset Load Parameters](image)

Using an Offset Load standard results in a more accurate calibration than with a Broadband Load. Therefore, when performing a calibration using one of the modified Cal Kit definitions, you may be prompted to connect more standards than before this change. To revert to using the Broadband Load Standard without offset, do the following:

1. Press **Cal > Cal Sets & Cal Kits > Cal Kit...**
2. At the **Manage Cal Kit** dialog box, click **Edit...**
3. Select the **SOLT** tab.
4. Under the Calibration Kit Classes, select **SC (Loads)**
5. Under Selected Standards, select **Broadband Load**, then click **Move Up** until the standard is at the top of the list. This will ensure that the Broadband Load is used first.

**About Offset Load**

An offset load is a compound standard consisting of a load element and two known offset elements (transmission lines) of different length. The shorter offset element can be a zero-length (Flush-thru) offset. The load element is defined as a 1-port reflection standard. An offset load standard is used
when the response of the offset elements are more precisely known than the response of the load element. This is the case with waveguide. Measurement of an offset load standard consists of two measurements, one with each of the two offset elements terminated by the load element. The frequency range of the offset load standard should be set so that there will be at least a 20 degree separation between the expected response of each measurement.

To specify more than two offset elements, define multiple offset load standards. In cases where more than two offsets are used, the frequency range may be extended as the internal algorithm at each frequency will search through all of the possible combinations of offsets to find the pair with the widest expected separation to use in determining the actual response of the load element.

The following Offset Load settings are available ONLY when Offset Load is selected.

- First Offset Standard
- Second Offset Standard
- Load Standard

---

**Thru Standard**

**Connectors** - Defines connector type at both ends of the Thru standard.

**Virtual Device**

Most cal kits have only one Thru standard definition for SOLT calibrations. For these cases, use the default selections (checked for zero-length Thrus and cleared for non-zero-length Thrus).
This checkbox is used to make forward and reverse measurements of your Thru standard for the same pair of ports in two separate steps. This is NOT common for zero-length (Flush) Thru standards.

When checked, calibration prompts involving that Thru will omit the Description. For example “Connect port 1 to port 2”. This is the common prompt for Flush-Thru standards.

When cleared, calibration prompts for that Thru will include the Description. For example “Connect <standard description> between ports 1 and 2”.

To make forward and reverse measurements of your Thru standard for the same pair of ports as two separate steps, do the following:

1. Create separate definitions of the Thru standard(s) using the same settings, except for the Label and Description. Clear this checkbox for BOTH definitions.

2. For one Thru definition, in the label and description include the word ‘FORWARD’ to prompt the operator to use this standard for the forward measurement. Assign this standard to the SOLT “FWD TRANS” and “FWD MATCH” classes of the cal kit.

3. For the Thru definition, in the label and description include the word ‘REVERSE’ to prompt the operator to use this standard for the reverse measurement. Assign this standard to the SOLT “REV TRANS” and “REV MATCH” classes of the cal kit.

4. When you perform SOLT calibrations using this cal kit, the forward measurements of the Thru will be measured in one connection step, and the reverse measurements in another.

Data-Based Standard

---

![Data-Based Standard](image)
Learn about the relative accuracy of Databased versus Polynomial Cal Kits.

The modified file can then be uploaded into the VNA.

**Upload Data From File**

Click **Load Data File**, then navigate to the *.dat or *.dsd file which is provided with the data-based Cal Kit. Both Response data and Accuracy (Uncertainty) data is provided in a single *.dat or *.dsd file.

**For Advanced Users**

Response data can be loaded from a *.s2p or *.cti file.

Accuracy data can be loaded from a *.cti file. Starting with Option S93015A/B, you can also load data from a *.dsd (S-parameter Data Standard Definition) file. The *.dsd file contains both Response data and Accuracy (Uncertainty) data where the accuracy data is in covariant form. Learn more about Dynamic Uncertainty.

**Virtual Device**

This checkbox is displayed for a Data-Based cal standard when the standard has been defined to have 2 ports.

- When Cleared (default) calibration prompts for that standard will include its Description. For example “Connect <standard description> between ports 1 and 2”.

- When Checked, calibration prompts for that standard will NOT include its Description, so the prompt will be worded as if the data-based standard is a zero-length Thru connection. For example “Connect port 1 to port 2”.

**Isolation Standard**
The pair of loads are considered one standard.

Both loads are connected to the VNA and measured with the same prompt.

---

**Available at the bottom of every tab**

**Save As** - Allows you to save the cal kit to a new file name and type.

**Save** - Saves the cal kit to the same file name and type.

**Close** - Closes the cal kit editing session. The file is NOT saved automatically.
Note: This dialog looks similar to the dialog that appears after the Cal Wizard View / Modify dialog. However, setting changes in that dialog affect ONLY the calibration that is in progress. These settings, accessed through Modify Cal Kit, changes the cal kit for all future calibrations that use this cal kit.

Allows you to assign single or multiple standards to SOLT Calibration Classes.

Click the TRL tab to assign standards to TRL Calibration Classes.

1. For each Cal Kit Class, select Available Standards from the left list, then click >> to copy the standard to the Cal Kit.

2. Use MOVE UP and MOVE DOWN to change the ORDER of the standards. The order is used in guided calibrations to determine which standards in that class will be used in calibrations that involve the frequency ranges over which the standards are defined. Guided cals will include standards in the order in which they appear in this class list, and in the case where standards in the class list have frequency ranges that overlap,
the order also determines which standard is used for frequencies in the overlap range. For example, let’s assume that you define a broadband Short from Min Freq. = 0 Hz and Max Freq. = 999 GHz, and that standard is listed first in the SB or TRL REFLECT class. If you then list a frequency-banded Short with the same connector below the broadband short in those same classes, then guided calibrations would not use the frequency-banded Shorts because the broadband Shorts would always be given priority.

SOLT <cal class> Label

The cal standard category label that appears in the VNA’s user interface during unguided SOLT calibrations.

Calibration Kit Classes

For each calibration class, select Available Standards, then click >> to move to the Selected Standards list.

- **SA** - OPEN Standards (standards in the SA class are not always Opens)
- **SB** - SHORT Standards
- **SC** - LOAD Standards
- **FWD / REV Trans and Match** - THRU Standards. Most Cal Kits do NOT include a physical THRU standard, but assume that an Insertable Thru will be used.
- **UNKNOWN THRU** - Unknown Thru Cal is the preferred THRU method of calibrating the analyzer to measure a non-insertable device. The Unknown Thru calibration is also known as Short-Open-Load-Reciprocal Thru (SOLR) calibration. Learn more.
- **ISOLATION** - Isolation standard. For VNA analyzers, ISOLATION calibration is not usually recommended. It could be beneficial in some situations where custom user-supplied test set hardware is being used.

Link FWD TRANS, FWD MATCH, REV TRANS, and REV MATCH Check to automatically assign the standard definition for FWD TRANS to FWD MATCH, REV MATCH, and REV TRANS. Clear to separately assign FWD MATCH, REV MATCH and REV TRANS classes (SOLT calibrations only).

Expanded Calibration

The following two check boxes apply ONLY during Guided Calibrations. For Unguided Calibration, these check boxes are ignored, including the case where the multiple standards dialog box is presented.

Measure all mateable standards in class Check this box to attain the very highest accuracy possible. For example, if a cal kit contains several load standards, during the calibration process you
will be prompted to measure each of the standards. This could require a significant amount of calibration time. When checked, the "Use expanded math when possible" box is also checked automatically.

**Use expanded math when possible** Some kits contain multiple calibration standards of the same type that together cover a very wide frequency range. (For example: multiple shorts, or a lowband load and a sliding load.) If a calibration requires more than one standard to cover the calibration frequency range, there can be regions of overlapping measurements. When this checkbox is selected, the VNA automatically computes the most accurate measurement in the overlap regions using a "weighted least squares fit" algorithm. This function improves accuracy without slowing the calibration speed.

- Manually select this checkbox only when using a cal kit that contains multiple standards of the same type. (For example: multiple shorts, or a lowband load and a sliding load.)
- The checkbox is cleared by default when a polynomial model is selected from the cal kit menu.
- The checkbox is selected by default when the 85058B or 85058E data-based model is selected from the cal kit menu.

---

**Available at the bottom of every tab**

**Save As** - Allows you to save the cal kit to a new file name and type.

**Save** - Saves the cal kit to the same file name and type.

**Close** - Closes the cal kit editing session. The file is NOT saved automatically.
Allows you to assign single or multiple standards to Calibration Classes.

1. For each Cal Kit Class, select **Available Standards** from the left list, then click >> to copy the standard to the Cal Kit.

2. Use **MOVE UP** and **MOVE DOWN** to change the **ORDER** of the standards. The order is used in guided calibrations to determine which standards in that class will be used in calibrations that involve the frequency ranges over which the standards are defined. Guided cals will include standards in the order in which they appear in this class list, and in the case where standards in the class list have frequency ranges that overlap, the order also determines which standard is used for frequencies in the overlap range. For example, let's assume that you define a broadband Short from Min Freq. = 0 Hz and Max Freq. = 999 GHz, and that standard is listed first in the SB or TRL REFLECT class. If you then list a frequency-banded Short with the same connector below the broadband short in those same classes, then guided calibrations would not use the frequency-banded Shorts because the broadband Shorts would always be given priority.

**Note:** The TRL LINE/MATCH class has a slight exception to these prioritization behaviors. In
general, Line standards are given a higher priority than Match standards. So if a Line standard and a Match standard are defined to have the same frequency range and the Match standard is listed above the Line standard in the class order, a guided TRL cal will still prefer to use the Line standard rather than the Match standard.

**TRL <cal class> Label**

The cal standard category label that appears in the VNA’s user interface during unguided TRL calibrations.

**Cal Kit Classes**

- For VNA analyzers, ISOLATION calibration is not usually recommended. It could be beneficial in some situations where custom user-supplied test set hardware is being used.

**TRL THRU**

**Note:** All **THRU calibration methods** are supported in a TRL Cal **EXCEPT** Unknown Thru.

- The THRU standard can be either a zero-length or non-zero length. However, a zero-length THRU is more accurate because it has zero loss and no reflections, by definition.

- The THRU standard cannot be the same electrical length as the LINE standard.

- If the insertion phase and electrical length are well-defined, the THRU standard may be used to set the reference plane.

- The THRU standard and LINE standard have the same characteristic impedance and are perfectly matched. They define the reference impedance of the calibration.

- If a THRU standard with the correct connectors is NOT available, an adapter removal cal can be performed.

**TRL REFLECT**

- The REFLECT standard can be anything with a high reflection, as long as it is the same when connected to one or more VNA ports.

- The REFLECT standard on each port is identical.

- The actual magnitude of the reflection need not be known.

- The phase of the reflection standard must be known within 1/4 wavelength.

- If the magnitude and phase of the reflection standard are well-defined, the standard may be used to set the
The LINE and THRU standards establish the reference impedance for the measurement after the calibration is completed. TRL calibration is limited by the following restrictions of the LINE standard:

- Must be of the same impedance and propagation constant as the THRU standard.
- The electrical length need only be specified within 1/4 wavelength.
- Cannot be the same length as the THRU standard.
- A TRL cal with broad frequency coverage requires multiple LINE standards. For example, a span from 2 GHz to 26 GHz requires two line standards.
- Must be an appropriate electrical length for the frequency range: at each frequency, the phase difference between the THRU and the LINE should be greater than 20 degrees and less than 160 degrees. This means in practice that a single LINE standard is only usable over an 8:1 frequency range (Frequency Span / Start Frequency). Therefore, for broad frequency coverage, multiple lines are required.
- At low frequencies, the LINE standard can become too long for practical use. The optimal length of the LINE standard is 1/4 wavelength at the geometric mean of the frequency span (square root of \( f_1 \times f_2 \)).

**Note:** The TRL LINE standard must have a delay that is greater than 0 (zero) ps. Otherwise, calibration correction calculations will contain unpredictable results.

If the LINE standard of appropriate length or loss cannot be fabricated, a MATCH standard may be used instead of the LINE.

- The MATCH standard is a low-reflection termination connected to both Port 1 and Port 2.
- The MATCH standard may be defined as an infinite length transmission line OR as a 1-port low reflect termination, such as a load.
- When defined as an infinite length transmission line, both test ports must be terminated by a MATCH standard at the same time. When defined as a 1-port load standard, the loads are measured separately. The loads are assumed to have the same characteristics.
- The impedance of the MATCH standard becomes the reference impedance for the measurement. For best results, use the same load on both ports. The load may be defined using the data-based definition, the arbitrary impedance definition, or the fixed load definition.
**System Z0**  The system impedance is used as the reference impedance. Choose when the desired test port impedance differs from the impedance of the LINE standard. Also, choose when skin effect impedance correction is desired for coax lines.

**Line Z0**  The impedance of the line standard is used as the reference impedance, or center of the Smith Chart. Any reflection from the line standard is assumed to be part of the directivity error.

**Multiline TRL**

For information about setting up and using a calibration kit definition that will cause the VNA to do a Multiline TRL, refer to the Multiline TRL Calibration topic.

**Testport Reference Plane (TRL only)**

**Thru Standard**  The THRU standard definition is used to establish the measurement reference plane. Select if the THRU standard is zero-length or very short.

**Reflect Standard**  The REFLECT standard definition is used to establish the position of the measurement reference plane. Select if the THRU standard is not appropriate AND the delay of the REFLECT standard is well defined.

Also, select If a flush short is used for the REFLECT standard because a flush short provides a more accurate phase reference than a Thru standard.

**LRL line auto characterization**

**Note:** This setting ONLY applies if an LRL Cal Kit is being modified AND Testport Reference Plane is set to Thru Standard AND the TRL Thru class standard and the TRL Line/Match class standard both have the same values for Offset Z0 and Loss. Otherwise, this setting is ignored.

- Check the box to allow the VNA to automatically correct for line loss and dispersion characteristics.
- Clear the box if anomalies appear during a calibrated measurement which may indicate different loss and impedance values for the Line standards.

**Available at the bottom of every tab**

**Save As**  - Allows you to save the cal kit to a new file name and type.

**Save**  - Saves the cal kit to the same file name and type.

**Close**  - Closes the cal kit editing session. The file is NOT saved automatically.
Multiline TRL Calibration

This topic describes the Keysight vector network analyzers (VNA) implementation of the NIST Multiline Thru Reflect Line (TRL) calibration algorithm. The Keysight implementation also supports a Multiline Thru Short Delay (TSD) calibration algorithm. This topic describes how to set up and use a calibration kit definition that will cause the VNA to do a Multiline TRL. It also discusses the various options that can be specified and how they interact with the standards as defined in Keysight calibration kits.

In this topic:

- Multiline TRL
- TRL Kit Classes
  - TRL THRU
  - TRL LINE/MATCH
  - TRL REFLECT
- Calibration Standards
  - Relationship Between Delay and Physical Length
- Calibration Reference Z0
  - Line Z0
  - System Z0
- Testport Reference Plane
  - Legacy TRL
  - Multiline TRL
- Multiline TRL Diagnostic Traces
- Procedure to Define the Calibration Kit
- References

Multiline TRL

The Multiline TRL algorithm is based on the NIST Multiline TRL calibration algorithm called MultiCal[1][2]. It provides an over-determined solution when multiple LINE standards are used. The TRL algorithm (Legacy TRL) that existed prior to Multiline TRL being added is still available for use. If the Multiline TRL check box is selected, then the Multiline TRL algorithm will be used when the defined calibration kit is used for TRL calibration. The Legacy TRL algorithm allowed multiple LINE standards to be used, but for any given frequency only one LINE standard is used with the THRU and REFLECT standards.
The dialog shown below is the TRL tab that comes up when editing the class definitions of the calibration kit. Its purpose is to define which standards are to be used during the TRL calibration. Both Legacy TRL or Multiline TRL automatically use a Thru Reflect Match (TRM) calibration when the frequency is below the lowest start frequency of the LINE standards defined in the calibration kit if there are MATCH standards defined. The Multiline TSD algorithm differs from the Multiline TRL in how the algorithm uses the SHORT/REFLECT standard. The SHORT is assumed to be fully known and only connected to one port while the REFLECT only needs to be partially known but has to be connected to both ports; the LINE and DELAY standards are equivalent.

**TRL Kit Classes**

For TRL there are four categories of standards called Calibration Kit Classes. They are TRL THRU, TRL LINE/MATCH, TRL REFLECT and ISOLATION. The ISOLATION kit class is optional.

**TRL THRU**

---

808
The TRL THRU kit class should be set up so that only a single standard is defined for a given set of port connectors.

**Note:** Generally this means that for Multiline TRL calibrations there will only be one standard defined in the TRL THRU kit class that is intended to be used over the entire frequency range. Sometimes users define calibration kits to cover multiple different port pairs; for example, if a calibration kit is created to support an on-wafer TRL calibration the user might define different ports that correspond to the topology of the set of probes that are used to touch down on the standards. The kit may have multiple pairs of ports. Giving each port a unique name allows the user to restrict which port pairs can be automatically set up to do a TRL calibration. In this case, you would have multiple THRU standards in the TRL THRU kit class. But only one of them would be used for a given TRL calibration between a pair of testports. It should be noted that the legacy TRL algorithm still may benefit from using multiple THRU standards in order to provide the best phase margin over different frequency ranges.

**TRL LINE/MATCH**

This actually represents two kit classes that share a similar purpose during calibration. A *LINE* standard needs to have a distinct response from the *THRU* standard. Typically the *LINE* standard is chosen such that the phase of $S_{21\text{LINE}}$ is different from the phase response of $S_{21\text{THRU}}$ by at least 20 degrees. This becomes prohibitive at low frequencies so *MATCH* standards are used with a TRM calibration at frequencies where it is impractical to create *LINE* standards. Both the Legacy TRL and Multiline TRL calibrations use *MATCH* standards and TRM calibration to cover the low frequency portion as needed. In those cases, the calibration is actually a mix of TRM and TRL. The TRM portion of the calibration is not over-determined when Multiline TRL is selected. A fundamental assumption of the TRL algorithm is that the characteristic impedance of the *THRU* standard is the same as the characteristic impedance of each of the *LINE* standards. The match standard can be defined as an idealized fixed load standard or it can be defined as a databased standard which can be used to adjust for the actual reflection coefficient of the standard. Both Legacy TRL and Multiline TRL assume the databased standards have the same reflection coefficient applied to both ports.

**Note:** The NIST implementation does not permit having a calibration that mixes both TRM and TRL.

**TRL REFLECT**

The TRL REFLECT kit class contains standards to be used by both the Legacy TRL and Multiline TRL calibrations. When doing a TRL calibration there is a fundamental assumption that the *REFLECT* standard is the same on both ports. It doesn’t have to be fully known, but should be approximately known to permit the correct root selection during the calibration process. A single reflect will be used, but most of the time multiple *REFLECT* standards need to be created one for each connector gender.
The Multiline TRL algorithm has the option to do a TSD calibration instead of TRL calibration. In that case, only one reflect standard is required; but it has to be fully known. When doing a TSD calibration, one then has the option of deciding which reflect standard to be used. In general, the order of the calibration standards in a Calibration Kit Class is significant as the calibration algorithms step through the standards from top to bottom selecting standards until it accumulates enough standards to satisfy all of the frequency requirements for each port. In the case of TSD, the order of the REFLECT standards is significant because as the TSD portion of the algorithm steps through the standards listed in the TRL REFLECT kit class it will use the first REFLECT standard it finds that will mate with either of the test ports. The user specifies which algorithm to use Multiline TRL or Multiline TSD on a kit by kit basis by making the desired selection in the Multiline TRL Setup dialog as shown below.

A databased standard can be used for the reflect standard. In general there is no advantage to using a databased standard for Legacy TRL and Multiline TRL calibrations since only an approximate value for the reflect is required. There are a couple of instances where using a databased standard might provide benefit. In the Legacy TRL when the Reflect Standard is used to define the reference plane, the actual value of the reflect standard obtained from the standard model is used to set the reference planes.
In the Multiline TSD calibration, the actual value of the standard model is also used to compute the error terms.

**Calibration Standards**

Legacy TRL and Multiline TRL do not support the use of databased standards for THRU or LINE standards. Databased standards can be used for MATCH or REFLECT standards.

**Relationship Between Delay and Physical Length**

The NIST MultiCal software defines the calibration standards using physical lengths and relative permittivity. The Keysight implementation also defines the same standards but uses an offset delay term to be consistent with existing calibration standard definitions[3]. Traditionally, the offset delay model used by both *THRU* and *LINE* standards has the offset defined by a delay parameter.

Note: The delay specified for the standard offset is assumed to be equivalent to the delay a TEM mode would face for a given physical length.

The Keysight Multiline TRL algorithm assumes the physical length is constant and computes that value from the offset delay parameter defined in the standard’s model. The physical length is obtained using the real part of the relative permittivity estimate (see Multiline TRL Setup dialog above) and then held constant even while the Multiline TRL algorithm adjusts the relative permittivity during subsequent computations.

**Calibration Reference Z0**

Both Legacy TRL and Multiline TRL algorithms assume the characteristic impedance for the *THRU* and *LINE* standards is the same. The computations optimize the error terms for the characteristic impedance of the standards. There are multiple impedance terms that are used by both the Legacy TRL and Multiline TRL algorithms. There is an additional impedance term $Z_{\text{line}}$ that is not defined in the calibration kit but is computed by the TRL algorithms. It is inherently subject to the assumptions that all of the THRU and LINE standards have the same characteristic impedance. When Line Z0 is selected as the reference Z0 setting, the characteristic impedance of the measurement system after calibration is simply $Z_{\text{line}}$.

For Legacy TRL, when System Z0 is selected an impedance transformation is applied which assumes that the Offset Z0 of the LINE standard is equal to $Z_{\text{line}}$ and the math to transform from Offset Z0 of the LINE standard to Z0 of the Connector is applied. The following summarizes the effect of settings on the impedance computations:
If the user selects **System Z0** for **Calibration Reference Z0** then the following applies:

- If the user selects **Enable Zc Computation From C** then \( Z_{line} \) is computed from \( C_0 \) and \( S_{r_{computed}} \).
- If the user unchecks **Enable Zc Computation From C** then \( Z_{line} \) is set equal to Offset Z0 of the THRU standard.
- Regardless of whether **Enable Zc Computation From C** is checked or unchecked there is an additional impedance transform from \( Z_{line} \) to Connector Z0.

If the user selects **Line Z0** for **Calibration Reference Z0** then \( Z_{line} \) is set equal to Offset Z0 of the THRU standard. There is no additional impedance transform from \( Z_{line} \) to Connector Z0.

**Line Z0**

If the user selects Line Z0 in the **Calibration Reference Z0** section of the TRL tab, the computed error terms are used without any additional impedance transform. Additionally, when the user selects Line Z0, the **Define Zc** section in the Multiline TRL Setup dialog is disabled.

**System Z0**

There is a slight difference in behavior between Legacy TRL and Multiline TRL due to the additional Define Zc section in the Multiline TRL Setup dialog.

Legacy TRL uses the definition of Offset Z0 and Offset Loss specified in the **LINE** standard model. If this is not equal to the Connector Z0 specified on the Connectors tab (shown below) an impedance transform is applied to the error terms.
Multiline TRL uses the definition of Offset $Z_0$ specified in the $THRU$ standard model. If **Enable Zc computation from C** is checked in the Multiline TRL Setup dialog an additional impedance transform is done to account for the low frequency dispersion specified by C0.

**Note:** There are papers published to explain details for determining the low frequency dispersion from the characterization of the propagation constant and an estimate of C0[4][5][6]. This also uses the characterization of the relative permittivity which covers the full frequency range of the Multiline TRL algorithm. The transmission line capacitance can also be characterized by additional measurements[7]

If the Offset $Z_0$ specified in the $THRU$ standard model is different from the Connector Z0 specified on the Connectors tab the error terms are modified by an impedance transform. A default value for C0 is computed assuming a coaxial transmission structure using characteristic impedance specified by Offset $Z_0$ for the $THRU$ standard model. Inputting a negative value for C0 will cause the default value to be reinserted.

**Note:** There is also an interaction between $\varepsilon_r$ and C0. If C0 is equal to the default value changes to $\varepsilon_r$ will update C0. If a user has specified a different value for C0 no change to C0 will be made when changes to $\varepsilon_r$ are made.

**Testport Reference Plane**

The testport reference plane options differ between the Legacy TRL and Multiline TRL options. The
initial calibrations for both Legacy TRL and Multiline TRL both assume the reference plane is in the middle of the THRU standard. This becomes significant for the LRL case where the delay of the THRU standard is greater than zero.

**Legacy TRL**

Legacy TRL has the option to use either the THRU standard or the REFLECT standard to set the reference plane after calibration. If the THRU standard is selected the reference plane for each testport will be rotated towards the VNA testport by half the delay of the THRU standard. If LRL line auto characterization is checked, the propagation constant computed during the TRL calibration will be used to estimate the rotation based on the delays specified by the THRU and LINE standards; otherwise, the modeled behavior of the THRU standard will be used.

**Multiline TRL**

When Multiline TRL is selected the ability to select LRL line auto characterization is disabled because Multiline TRL only uses LRL line auto characterization. The ability to select the REFLECT standard to be used to define the testport is not an option with Multiline TRL; however, there is an additional option of defining an arbitrary reference plane. A negative number input for the reference plane shifts the testport towards the VNA testports.

The default behavior is to shift the testport reference plane to the edge of the THRU standard like with behavior similar to the Legacy TRL behavior with the testport reference plane set by the THRU standard and having LRL line auto characterization active.

**Multiline TRL Diagnostic Traces**

The diagnostic traces are available in the Cal Set viewer. The trace also shows the transition between the portion of the frequency range that did a TRM calibration and the portion that did a Multiline TRL calibration.

Multiline TRL has diagnostic traces that provide insight into the calibration as shown below.
The traces are the same as traces provided by the NIST MultiCal software. These traces can be accessed from the Cal Set Viewer under the Standards group.

**Note:** The default format for viewing the traces is log magnitude. The diagnostic traces make the most sense when viewed in real or imaginary format. The indicated units for each diagnostic trace is simply a number (U); when looking at the various diagnostic traces the screen doesn’t include units such as degrees or ohms.

The included diagnostic traces are:

- **CharImpedance** (Characteristic Impedance of line) is the estimate $Z_0$ of the transmission media. A fundamental assumption of both the Legacy TRL algorithm and the Multline TRL algorithm is that the characteristic impedance of the **THRU** and **LINE** standards are the same. If **Enable Zc computation from C** is selected for the calibration kit, CharImpedance will include the effects of low frequency dispersion and transmission line loss; otherwise it will show the Offset $Z_0$ term specified for the **THRU** standard.

- **EffPhase** (Effective Phase Delay (best pair)) provides an estimate of the effective phase delay difference between the best pair of standards taken from the collection of the **THRU** standard and all of the **LINE** standard. Note it will not always be the **THRU** standard with one of the **LINE** standards; often it is based on the combination of two **LINE** standards. Optimal phase difference is $90^\circ$.

- **RelPermittivityMean** (Effective dielectric constant (mean)) is an estimate of the complex relative permittivity ($\varepsilon_r$) given as $\varepsilon_r = \varepsilon'_r - j\varepsilon''_r$ where $\varepsilon'_r$ is the real part and $\varepsilon''_r$ is the imaginary part. Strictly speaking, this is not the mean of each of the **RelPermittivityPair(n)**; pairs of lines with the best effective phase delay have influence.

- **RelPermittivityPair(n)** (Effective dielectric constant ($n^{th}$ pair)) is an estimate of the complex relative permittivity ($\varepsilon_r$) between the **THRU** and multiple **LINE** standards paired with the **THRU** or **LINE** standard that
has the best effective phase difference with the rest of the THRU and/or LINE standards. These are the terms that are combined to give the estimate of RelPermittivityMean. The number of pairs is equal to the number of LINE standards.

- **RelPhaseConst&Atten** (Relative Phase Constant and Attenuation) provides information about the propagation constant given as \( \gamma = \alpha + j\beta \) where \( \alpha \) is the attenuation constant and \( \beta \) is the phase constant. The transmission through a line of length \( l \) in cm is given as \( e^{-\gamma l} \). The data is normalized to frequency such that the real part of RelPhaseConst&Atten represents \( \frac{\beta c}{2\pi f_{GHz}} \) and the imaginary part of RelPhaseConst&Atten represents \( \frac{\alpha c}{2\pi f_{GHz}} \) expressed in dB where \( c \) is the speed of light in a vacuum and \( f_{GHz} \) is the frequency in GHz.

- **RelStdDev** (Standard Deviation) provides a relative standard deviation for the calibration process. It is related to uncertainty of the calibration process with lower numbers indicating lower uncertainty. It is a normalized value and is not a numeric representation of the uncertainty. More details are provided in the paper by DeGroot, Jargon and Marks[2].

The diagnostic traces only have a non-zero value in the frequency range where the Multiline TRL algorithm is used. The transition between TRM and Multiline TRL is seen in the diagnostic trace above.

### Procedure to Define the Calibration Kit

The following steps are recommended when creating a new calibration kit to be used with Multiport TRL calibrations.

1. **Determine the nominal effective relative permittivity.** If using a substrate or PC board, this should account for both the effect of the substrate permittivity and air. This is the number that should be placed in the Relative Permittivity Estimate shown in the Multiline TRL Setup dialog. It will be used when estimating the Offset Delay of the calibration standards. All devices in the calibration kit are assumed to use the same relative permittivity estimate.

2. **Define the connectors.** It can be beneficial to define multiple connectors when the topology of the calibration kit is intended to have a particular standard be used only with a probe in a fixed location. In that case it could be beneficial to provide unique connector names to be associated with that probe. For example, you might end up with Probe A, Probe B, Probe C, etc.

3. **Define the characteristic impedance of the connectors.** This is the final system impedance desired after calibration. Generally, all of the connectors in the calibration kit would have the same characteristic impedance.

**Note:** This is not a fixed requirement when the calibration kit is used for multiport calibrations (more than 2 testports); what is required is that all connectors used to define the standards for a Multiline TRL calibration (or Legacy TRL calibration) must have the same characteristic impedance. For example, consider the case where TRL calibrations are intended between ports A and B and between ports C and D. Assume Probe A is intended to be connected to port A, Probe B to port B, Probe C to port C and Probe D to port D; the characteristic impedance defined for Probe A must be the same as Probe B and the characteristic impedance defined for Probe C must be the same as Probe D but the characteristic impedance defined for Probe A can be different than the characteristic impedance defined for Probe C.
4. Avoid using meander lines or bent lines for standards intended to be used as either the THRU standard or a LINE standard. It is possible to include bent lines needed during a multiport calibration but the bent lines would be included in the UNKNOWN THRU kit class under the SOLT tab in the Edit kit dialog. The SOLT tab is next to the TRL tab.

5. Measure the physical lengths of the standards in cm. This would include the standard to be used as a THRU. All of the THRU and LINE standards should be defined using the THRU standard model. Specify the parameters for each standard:
   - Specify the Label and Description.
   - Select the connectors.
   - Specify the frequency range. The THRU standard should be specified over the full frequency range of the calibration. The LINE standards should have a minimum frequency selected to ensure a minimum phase difference from the THRU standard of approximately 20 degrees.
   - The offset delay should be computed using the real part of the relative permittivity estimate ($\varepsilon''_r$) that will be defined on the Multiline TRL Setup dialog. The delay in picoseconds (psec) for the $i^{th}$ standard with length $L_i$ in cm is computed as:
     \[ D_i = \frac{L_i \sqrt{\varepsilon''_r}}{c} \]
     where
     \[ c = 2.99792458 \times 10^{-2} \text{ cm/psec} \]
   - The offset Z0 should be the nominal characteristic impedance of the THRU and LINE standards. They should all be set to the same value. In many cases, it will be the same value as the characteristic impedance defined for the connectors.
   - The offset loss can be specified. It is not critical to specify this accurately since the Multiline TRL calibration will estimate the loss of the propagation independent of this estimate.
   - If the Delay is specified as 0 psec it makes sense to check the Virtual Device box. This causes the calibration prompts to indicate connecting the testports directly to each other rather than including the device label and description.

6. Assign calibration kit class definitions by selecting the TRL tab.
   - A single THRU standard should be selected.

   **Note:** In the case where multiple TRL sets of calibration standards are defined, there should be a single THRU standard for each set. Using the previous example, there would be a THRU defined with Probe A and Probe B connectors and another THRU defined with Probe C and Probe D connectors.

   - Multiple LINE standards can be selected.
   - If the frequency range of usage is lower than the lowest frequency for the LINE standards add MATCH standards. There should be a single MATCH standard for each port connector. The model for each MATCH standard should be the same.
   - Add the REFLECT standards. There should be a REFLECT standard for each port connector. If the calibration kit is intended for Multiline TRL calibrations the REFLECT standards should all have the same model. If the calibration kit is intended for Multiline TSD the REFLECT standards do not have to have the same model. For Multiline TSD calibrations only the first standard listed that matches the connectors required during the calibration will be used.

7. Check the Multiline TRL checkbox.
8. Specify the calibration reference Z0. Usually System Z0 is selected which includes the impedance transform between the Offset Z0 defined and the Connector Z0. It also includes the transform for Zc defined in the
Multiline TRL Setup dialog.
9. Press **MLTRL Setup**... to open the Multiline TRL Setup dialog.
   - Define Zc. Check the Enable Zc Computation from C checkbox to account for low frequency
dispersion. When setting up a calibration kit for waveguide, uncheck the Check the Enable Zc
Computation from C checkbox since the cutoff frequency is used instead.
   - Specify C0. If defining a coaxial calibration kit, the default value is usually adequate. If the value has
been modified, the default value can be automatically re-entered by inputting a negative number in the
C0 field. Sometimes the value for C0 nees to be determined by an iterative process. There are papers
published that explain details for determining the low frequency dispersion from the characterization of
the propagation constant and an estimate of C0[4][5][6].
   - Make sure the values input for the relative permittivity estimate match the value used to estimate the
offset delay for the **THRU** and **LINE** standards.
   - Specify either TRL or TSD.
   - Specify the testport reference plane as desired.
10. If doing an multiport calibration, add additional THRU standards with offset delays specified. These standards
are usually loopback THRU standards or bent in order to connect ports that are not directly opposing each
other. Add these standards to the UNKNOWN THRU kit class.

References


constant measurement,” *IEEE Microwave and Guided Wave Letters*, vol. 1, no. 6, pp. 141–143,

Microwave and Guided Wave Letters*, vol. 4, no. 6, pp. 175–176, 1994.

[7] “Transmission line capacitance measurement,” *IEEE Microwave and Guided Wave Letters*, vol. 1,
Markers

Markers provide a numerical readout of measured data, a search capability for specific values, and can change stimulus settings. There are 15 regular markers and one Reference marker (used with Delta markers) available per trace. This topic discusses all aspects of markers.

Note: Marker Readout can be turned ON/OFF and customized from the Customize Display dialog box. Learn more.

- Number of General Purpose and Reference Markers
- Creating and Moving Markers
- Marker Setup
  - Coupling Method
- Searching with Markers
  - Maximum and Minimum Search
  - Peak Search
  - Multi Peak Search
  - Target Search
  - Multi Target Search
  - Bandwidth and Notch Search
  - Compression Search
  - PSAT Search
  - PNOP Search
  - Distortion Search (Option S93070xB Modulation Distortion only)
  - Spurious Search (Option S93031xB Phase Noise only)
- Search Domain
- Search Range Indicators
Marker Functions (Change Instrument Settings)

SA Analysis Markers (Spectrum Analyzer channel markers)

SA Analysis Markers (Option S93070xB Modulation Distortion only)

Marker Display

Marker Table

Note: Marker Readout can be turned ON / OFF and customized from the View/Display menu. Learn more.

Other Analyze Data topics

Creating and Moving Markers

How to Create Markers

Using Hardkey/SoftTab/Softkey

1. Press Marker > Marker 1-7 / Marker 8-15 / Reference.

2. Click left side Marker N or Reference small button.

Using a mouse

1. Move the cursor on a trace.

2. Right-click on the trace then select Add Marker....

Number of General Purpose and Reference Markers

There are 15 regular markers and one Reference marker (used with Delta markers) available per trace.

For Remote commands, there is a Preference setting to revert to the previous behavior. See the Preference.

When developing new programs, use the reference marker commands to control reference markers. Do NOT use general purpose marker commands to control reference markers.

Moving a Marker

To move a marker, make the marker active by selecting its number in any of the previous 3 methods. The active marker appears on the analyzer display as ∇. All of the other markers are inactive and are represented on the analyzer display as Δ. Then change the stimulus value using any of the following methods:
- Type a value.
- Scroll to a stimulus value using the up / down arrows. The resolution can not be changed.
- Click the stimulus box, then use the front-panel knob.
- Click and Drag Markers using a finger (touchscreen) or by left-clicking and holding a marker symbol. Then drag the marker to any point on the trace. This feature is NOT allowed in Smith Chart or Polar display formats or with a Fixed Marker type.
- Move the mouse Marker over the stimulus box, then use the mouse scroll wheel.

**Marker Setup**

**How to set the Marker Setup.**

**Using Hardkey/SoftTab/Softkey**

1. Press **Marker > Marker Setup.**
2. Set the value or select desired setting for each softkey.

**Using a mouse**

1. Move the cursor on a marker.
2. Right-click on the marker then select Marker....

**Marker dialog box help**

**Marker**  Specifies the current (active) marker number that you are defining.

**On**  Check to display the marker and corresponding data on the screen.

**Stimulus**  Specifies the X-axis value of the active marker. To change stimulus value, type a value, use the up and down arrows, click in the text box and use the front-panel knob, or drag the marker on the screen.

**Delta (and Reference) Markers**  Check to make the active marker display data that is relative to the reference (R) marker. There is only one reference marker per trace. All other markers can be regular markers or delta markers. When a delta marker is created, if not already displayed, the
A reference marker is displayed automatically. A delta marker can be activated from the Marker dialog box or the Marker Toolbar. See Also: Number of General Purpose and Reference Markers.

**Discrete Marker** Check to display values at only the discrete points where data is measured. Clear to display values that are interpolated from the data points. The interpolated marker will report y-axis data from ANY frequency value between the start and stop frequency.

**Fixed** Check to cause the marker to have a fixed X-axis and Y-axis position based on its placement on the trace when it was set to fixed. It does NOT move with trace data amplitude. It can be scrolled left and right on the X-axis by changing the marker stimulus value. Use this marker type to quickly monitor "before and after" changes to your test device. For example, you could use fixed markers to record the difference of test results before and after tuning a filter.

Clear the box to create a **Normal** marker, which has a fixed stimulus position (X-axis) and responds to changes in data amplitude (Y-axis). It can be scrolled left and right on the X-axis by changing the marker stimulus value. Use this marker type with one of the marker search types to locate the desired data.

**Marker Format** Displays the marker data in a format that you choose. The Trace Default setting has the same marker and grid formats. Choose from the following:

<table>
<thead>
<tr>
<th>Format</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log/Phase</td>
<td>Log, Real, Fahrenheit</td>
</tr>
<tr>
<td>Linear/Phase</td>
<td>Linear, Imaginary, Celsius</td>
</tr>
<tr>
<td>Real/Imag</td>
<td>Phase, Phase Unwrapped, Noise</td>
</tr>
<tr>
<td>R+jX (complex impedance)</td>
<td>SWR, Phase Positive</td>
</tr>
<tr>
<td>G+jB (complex admittance)</td>
<td>Delay, Kelvin</td>
</tr>
</tbody>
</table>

**Noise Marker Format** (IMSpectrum and SA Measurement Classes Only) - For comparison purposes, electronic noise measurements are often displayed as though the measurement was made in a 1 Hz Res BW. However, making an actual measurement at a 1 Hz Res BW is impossible, and at 10 Hz, extremely slow.

A Noise Marker mathematically calculates the noise measurement at that single data point as though it were made using a 1 Hz bandwidth.

To accurately measure noise, the Noise Marker should NOT be placed on, or too close to, a signal. The distance from a signal depends on several factors. To know if an accurate reading is being
made, move the Noise Marker until consistent measurements are displayed in adjacent data points.

**Noise Marker Format (Modulation Distortion measurement class only)** - In a Modulation Distortion channel, the noise is calculated from the current power reading divided by the tone spacing of the modulation being used: \[ P_{in}(W/Hz) = \frac{P_{in}(W)}{\text{Tone Spacing (Hz)}} \]. This provides a noise density readout with units compatible with the trace format. For example, if the trace is dBm, the noise is dBm/Hz.

**All Off**  Switches OFF all markers on the active trace.

**Coupled Markers**

The coupled markers feature causes markers on different traces to line up with the markers on the selected trace. Markers are coupled by marker number, 1 to 1, 2 to 2, 3 to 3, and so forth. If the x-axis domain is the same (such as frequency or time), coupling occurs. Trace markers in a different x-axis domain will not be coupled. If a trace marker has no marker to couple with on the selected trace, the marker remains independent.

**Coupling Method** - Determines the scope of coupling. Choose from the following:

- **All** - A marker on one trace is coupled to the same-numbered markers on all channels, all windows and all traces.

- **Channel** - A marker on one trace is coupled to the same numbered markers on traces which share the same channel number as the original trace.

**Coupled Markers Model**

This model simulates the use of coupled markers in the VNA:

1. **Click Trace A or Trace B**

2. **Click Coupled Markers**
3. Notice the following:

- Markers on the unselected trace move to the x-axis position of the selected trace.
- If a marker number on the unselected trace has no corresponding marker on the selected trace, no movement occurs for that marker.

4. Click **Reset** to run the model again. There is no Reset for coupled markers on the VNA.

**Searching with Markers**

You can use markers to search and return data for the following trace criteria:

- **Max and Min**: find the highest or lowest points on the trace
- **Peak**, then move to other peaks (left, right, next highest)
- **Multi Peak**
- **Target Value**: find a specific Y-axis value
- **Multi Target**:
- **Bandwidth** (Filters)
- **Notch** (Filters)
- **Compression Point** (Amplifiers)
- **About PSAT and PNOP Markers**
  - **Power Saturation** (Amplifiers)
  - **Power Normal Operating Point** (Amplifiers)
- **Search Domain**
- **Search Range Indicators**
# How to Search with Markers

## Using Hardkey/SoftTab/Softkey

1. Press **Search** > *Main / Peak / Target / Multi Peak & Target / Bandwidth & Notch / Compression & Saturation / Normal Op Pt. / Distortion / Spurious*

## Using a mouse

1. Move the cursor on a marker.
2. Right-click on the marker then select **Search**...to show the Marker Search Dialog box for define the search parameters.
3. From **Search Type** of Marker Search dialog box, select the desired search function.
4. Press **Execute** or check **Tracking**. Learn more.

## Marker Search dialog box help

**Marker** Specifies the marker that you are defining. Not available for search types that deploy specific markers.

**Search Range** Defines the area where the marker can move or search. For full span, the marker searches for specified values within the full measurement span. For user span, the marker searches for specified values within a measurement span that you define. Learn more about Search Range.

**Search Type**

**Note** You must either press **Target Search** or check **Tracking** to initiate all search types. If there is no valid data match for the search type, the marker will not move from its current position.

- **Target Search** Click to cause the marker to search for the specified criteria.
- **Tracking** Check to cause the marker to search for the specified criteria with each new sweep. The
searches begin with the first sweep after Tracking has been checked, based on the current search type and domain information. Therefore, make sure that the search criteria are in the desired state before using the data. You cannot manually change the stimulus setting for a marker if Tracking is selected for that marker.

**Maximum** Marker locates the maximum (highest) data value.

**Minimum** Marker locates the minimum (lowest) data value.

---

**Peak** - See below.

**Next Peak** - Marker locates the peak with the next lower amplitude value relative to its starting position.

**Peak Right** - The marker locates the next valid peak to the right of its starting position on the X-axis.

**Peak Left** - The marker locates the next valid peak to the left of its starting position on the X-axis.

**Multi Peak** - A function that search for peaks that match the multi-peak search excursion value and multi-peak polarity value. Learn more about Multi Peak Search.

- **Threshold** - Minimum amplitude (dB). To be considered valid, the peak must be above the threshold level. The valley on either side can be below the threshold level.

- **Excursion** The vertical distance (dB) between the peak and the valleys on both sides. To be considered a peak, data values must "fall off" from the peak on both sides by the excursion value.

---

**Target** - A function that searches for a target that matches the pre-defined target value and transition types (positive, negative or both (positive and negative)) and then moves the marker to that target. Learn more about Multi Target Search.

**Target Left** - A function executes the search from the current marker position to the smaller stimulus values and moves the marker to first target encountered.

**Target Right** - A function executes the search from the current marker position to the larger stimulus values and moves the marker to first target encountered.

**Multi Target** - A function that search for targets that are of the multi-target value and multi target transition value. Learn more about Multi Target Search.

- **Target** - Value in dB.
**Compression** - A function used the active marker to find the specified gain **Compression Level**. Learn more about Compression Search.

- Compression - Value in dB.

**Bandwidth** and **Notch** searches are accessed by pressing **Search > Bandwidth & Notch > BW Ref to** or **Search > Bandwidth & Notch > Notch Ref to**.

**Bandwidth** - A function for determining the bandwidth of the trace, center frequency, cut-off points (on the higher frequency and the lower frequency sides), Q and insertion loss based on the position of the active marker (if search mode set to Marker) or the peak marker (if search mode set to Peak). Learn more about Bandwidth Search.

- Bandwidth peak mode search
  - If level is negative, search is relative to the maximum peak.
  - If level is positive, search is relative to the minimum peak.
• Bandwidth level in dB.

**Notch**  -  A function is used to obtain the bandwidth, center frequency, cutoff points (high-frequency side and low-frequency side), Q and insertion loss of a trace based on the position of the active marker (if search mode set to Marker) or the peak (if search mode set to Peak). [Learn more about Notch Search.]

- Notch peak mode search
  - If level is negative, notch search is relative to the minimum peak.
  - If level is positive, notch search is relative to the maximum peak.
  - Notch level in dB.

The default behavior for searches based on the active marker or peak marker can be set using the Marker: On Preset, set BW/Notch search reference to Peak preference.

---

**Power Saturation**  -  [Learn more about PSAT Search.]

- PMax Back-off - Value in dB.

---

**Normal Operating Pt**  -  The output power where the input is offset from the back-off input power by the Pin Offset. [Learn more about PNOP Search.]

- Back-off - Value in dB.
- Pin Offset - X-axis value in dB.

---

**Distortion Search (Option S93070xB Modulation Distortion only)**

Modulation Distortion marker searches are accessed by pressing **Search** > **Distortion** or by selecting the search marker from the **Search Type** pulldown menu in the **Marker Search** dialog.

The Modulation Distortion application has marker search function types similar to a standard channel. In addition, it has marker search types specific to a Modulation Distortion channel.

**ACPR Search**  -  ACPR Search turns on the reference marker and 2 delta markers if the active trace is an ACP trace and there is an ACP measurement band defined in the setup. The reference marker covers the signal carrier range of the band, while the two delta markers cover the ACLo and ACUp ranges. The readouts of these markers are then ACPR values.

**NPR Search**  -  NPR Search turns on the reference marker and 1 delta marker if the active trace is a
NPR trace and there is a NPR measurement band defined. The reference marker covers the signal carrier range of the band, and the delta marker covers the notch range. The readout of the delta marker is then the NPR value.

---

**Spurious Search (Option S93031xB Phase Noise only)**

Phase Noise marker searches are accessed by pressing `Search > Spurious` or by selecting the search marker from the **Search Type** pulldown menu in the **Marker Search** dialog.

The Phase Noise application has marker search function types similar to a standard channel. In addition, it has marker search types specific to a Phase Noise channel. Spur analysis must be enabled to use the Spurious search functions. See [Enable Spur Analysis](#).

**Spurious Search** - Searches for spurious signals based on spurious settings.

**Spurious Right >> Search** - Searches for the next spurious signal to the right.

**<< Spurious Left Search** - Searches for the next spurious signal to the left.

**Multi Spurious Search** - Searches for multiple spurious signals.

**Spur Sensibility** - Sets the spurious sensibility number. [Learn more.](#)

---

**Maximum and Minimum Search**

**How to create Maximum and Minimum Search**

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Search &gt; Main</strong>.</td>
<td>1. Move a cursor on a marker.</td>
</tr>
<tr>
<td>2. Click <strong>Max Search</strong> or <strong>Min Search</strong>.</td>
<td>2. Right-click on the marker and then select <strong>Search</strong>...</td>
</tr>
<tr>
<td>3. Optionally click <strong>Tracking</strong> to search for the specified maximum or minimum level with each sweep. <a href="#">Learn more.</a></td>
<td>3. From <strong>Search Type</strong> of Marker Search dialog box, select <strong>Maximum / Minimum</strong>.</td>
</tr>
<tr>
<td></td>
<td>4. Press Execute or check Tracking. <a href="#">Learn more.</a></td>
</tr>
</tbody>
</table>

---

You can search for the maximum or minimum measured value on the trace and move a marker to that point.
Search for maximum (Max Search) Move active marker to point on trace where measured value is greatest.

Search for minimum (Min Search) Move active marker to point on trace where measured value is lowest.

**Note:** When the data format is in Smith chart or polar format, execute the search only for the main response value.

### Peak Search

**How to create Peak Search**

**Using Hardkey/SoftTab/Softkey**

1. Press **Search > Peak**.

2. Click **Peak Search** to show the markers on the peak.

3. Click **Peak Right >> Search, << Peak Left Search or Next Peak Search** to move the marker to the peak.

4. Click **Threshold** to enter the value of peak threshold.

5. Click **Excursion** to enter the lower limit value of peak excursion.

6. Click **Peak Polarity** to select a peak polarity.

7. Optionally click **Tracking** to search for the specified peak level with each sweep. Learn more.

**Using a mouse**

1. Move a cursor on a marker.

2. Right-click on the marker and then select **Search...**

3. From Search Type of Marker Search dialog box, select **Peak / Peak Left / Peak Right / Next Peak**.

4. Enter the value of **Threshold** and **Excursion**.

5. Press **Execute** or check **Tracking**. Learn more.
A peak is a measurement point whose value is greater or smaller than the adjoining measurement points on its right and left sides. Peaks are classified into the following two types depending on the difference in magnitude from the measurement points on either side of it.

**What Is a "Peak"?**

You define what the analyzer considers a "peak" by selecting the following two peak criteria settings:

- **Threshold** - Minimum amplitude (dB). To be considered valid, the peak must be above the threshold level. The valley on either side can be below the threshold level.

- **Excursion** - The vertical distance (dB) between the peak and the valleys on both sides. To be considered a peak, data values must "fall off" from the peak on both sides by the excursion value.

**Peak Polarity:**

<table>
<thead>
<tr>
<th>Definition</th>
<th>Detection</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Positive</strong> A peak whose measured value is greater than those of the measurement points on either side of it.</td>
<td>Detect positive peaks which are larger than Threshold.</td>
</tr>
<tr>
<td><strong>Negative</strong> A peak whose measured value is smaller than those of the measurement points on either side of it.</td>
<td>Detect negative peaks which are smaller than Threshold.</td>
</tr>
<tr>
<td><strong>Both</strong> A peak whose measured value is smaller and greater than those of the measurement points on either side of it.</td>
<td>Threshold value is not used when polarity is set to both.</td>
</tr>
</tbody>
</table>

**About Peak Excursion Value**

The peak excursion value is the smaller of the differences in measured values from the adjoining peaks of the opposite polarity.
Executing a Peak Search

The following 3 methods are available for executing the peak search:

Next Peak
Moves the marker to the maximum peak when peak polarity is Positive or Both.

Peak Left
Executes the search from current marker position to the smaller stimulus values and moves the marker to first peak encountered.

Peak Right
Executes the search from current marker position to the larger stimulus values and moves the marker to first peak encountered.
Note: Peak right, peak left and next peak may not be tracked. If these searches are selected and then tracking is turned on, the peak tracking is enabled.

When the data format is in Smith chart or polar format, execute the search for the main response value of the two marker response values.

Changing the settings of peak excursion value or peak polarity executes new search for multiple peak.

Example:

Threshold Setting: -10dB
Excursion Setting: 1dB
Scale = 1 dB / Division
Mouse over the graphic to find a valid peak.
- Peak A = Valid Peak (Above Threshold and Excursion Settings)
- Peak B = Invalid Peak (Below Excursion Setting)
- Peak C = Invalid Peak (Below Threshold Setting)

Multi Peak Search

How to create Multi Peak Search

Using Hardkey/SoftTab/Softkey

1. Press Search > Multi Peak & Target.
2. Click Multi Peak Search to show the markers on the multi peaks.
3. Click Peak Threshold to enter the value of peak threshold.
4. Click Peak Excursion to enter the lower limit value of peak excursion.
5. Click Peak Polarity to select a peak polarity.
6. Optionally click Tracking to search for the specified multi peak level with each sweep. Learn more.

Using a mouse

1. Move a cursor on a marker.
2. Right-click on the marker and then select Search....
3. From Search Type of Marker Search dialog box, select Multi Peak.
4. Enter the value of Threshold and Excursion.
5. Press Execute or check Tracking. Learn more.

The multi peak search function enables you to display markers on multiple peaks on traces. Depending on the number of detected peaks, markers 1 through 15 are displayed from the start frequency. The reference marker is not affected.
Multiple peak search has threshold, excursion and polarity as user defined values. This search may have tracking enabled.

When the multiple peak search is executed, previous markers search and tracking are disabled and the settings for the multiple peak search are used.

**Note:** Do not use individual marker settings or marker domain.

Put markers on each valid peak, using up to 15 markers.

**Target Search**

**How to create Target Search**

**Using Hardkey/SoftTab/Softkey**

1. Press **Search > Target**.
2. Click **Target Search** to enable the target search.
3. Click **Target Right >> Search** or **<< Target Left Search** to move the marker to the target.
4. Click **Target Value** to input the value of target search.
5. Click **Transition** to select a transition type.
6. Optionally click **Tracking** to search for the specified target level with each sweep. Learn more.

**Using a mouse**

1. Move a cursor on a marker.
2. Right-click on the marker and then select **Search**....
3. From **Search Type** of Marker Search dialog box, select Target/Target Left/Target Right.
4. Enter the value of the **Target**.
5. Press **Execute** or check **Tracking**. Learn more.

The target search is a function that searches for a target that matches the pre-defined target value and transition types (positive, negative or both positive and negative) and then moves the marker to that target.
Target Transition Types

A target is a point that has a specific measured value on the trace. Targets can be divided into the 3 groups shown below depending on their transition type.

<table>
<thead>
<tr>
<th>Transition Type</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>The target value is larger than the measured value immediately preceding it.</td>
</tr>
<tr>
<td>Negative</td>
<td>The target value is smaller than the measured value immediately preceding it.</td>
</tr>
<tr>
<td>Both</td>
<td>The conditions for either Positive or Negative transition are satisfied.</td>
</tr>
</tbody>
</table>

Executing a Target Search

The following 3 methods are available for executing the target search:

- **Target Left**: Executes the search from the current marker position to the smaller stimulus values and moves the marker to first target encountered.
- **Target Right**: Executes the search from the current marker position to the larger stimulus values and moves the marker to first target encountered.
- **Multi Target**: Executes the search for targets that are of the multi-target value and multi target transition value. See Multi Target Search.
**Note:** Target right and target left cannot have tracking enabled. If target left or target right is the selected search and then tracking is enabled, target tracking is enabled.

When the data format is in Smith chart or polar format, execute the search for the main response value of the 2 marker response values.

Changing the settings of target value or transition type executes new search for multiple target.

The marker moves to the first occurrence of the Target value to the right of its current position. Subsequent presses of the **Target Search** softkey cause the marker to move to the next value to the right that meets the Target value. When the marker reaches the upper end of the stimulus range, it will "wrap around" and continue the search from the lower end of the stimulus range (left side of the window).
If **Discrete Marker** is OFF, the marker locates the interpolated data point that equals the target value.

If **Discrete Marker** is ON and there are two data points on either side of the target value, the marker locates the data point closest to the Target value.

### Multi Target Search

#### How to create Peak Search

**Using Hardkey/SoftTab/Softkey**

1. Press **Search > Multi Peak & Target**.
2. Click **Multi Target Search** to show the markers on the multi target.
3. Click **Target Value** to enter the value of target.
4. Click **Transition** to select a transition type.
5. Optionally click **Tracking** to search for the specified multi target level with each sweep. Learn more.

**Using a mouse**

1. Move a cursor on a marker.
2. Right-click on the marker and then select **Search**....
3. From **Search Type** of Marker Search dialog box, select **Multi Target**.
4. Enter the value of the **Target**.
5. Press **Execute** or check **Tracking**. Learn more.

The multi target search is a function that searches for targets that match to pre-defined target value and transition types (positive, negative or both of positive and negative) and displays markers on the targets being searched.

Depending on the number of detected targets, markers 1 through 15 are displayed from the start frequency. The reference marker is not affected.

When the multi target search is executed, search and tracking settings for markers 1 through 15 are ignored and the settings for the multi target search are used.

**Note:** Put markers on each found target value, using up to fifteen markers. Reference marker is not affected. Do not use individual marker settings or marker domain. Search range is applied.

Multiple target search has **target** and **transition types** as user defined values. This search may have tracking enabled. When this search is executing, previous marker searches are disabled.

### Bandwidth and Notch Search
Bandwidth and notch search behavior depends on whether the preference called **Marker: Use single marker for marker search** is set or not. When set, only one marker is used for a marker search. Sub markers are displayed and used for Bandwidth and Notch searches. When cleared, multiple markers are used for a marker search. The default is cleared.

### How to create Bandwidth and Notch Search

**Using Hardkey/SoftTab/Softkey**

1. Press **Search > Bandwidth & Notch**.
2. Click left side **Bandwidth Search** or **Notch Search** small button to turn it ON/OFF.
3. For Bandwidth search, click **BW Ref To > Marker** or **Peak**.
4. For Notch search, click **Notch Ref To > Marker** or **Peak**.
5. Specify the **BW Level** or **Notch Level** in dB from the peak or valley where bandwidth / notch is measured.
6. Optionally click **Tracking** to search for the specified bandwidth or notch level with each sweep. Learn more.

**Using a mouse**

1. Move a cursor on a marker.
2. Right-click on the marker and then select **Search**....
3. From **Search Type** of Marker Search dialog, select **Bandwidth** or **Notch**.
4. Specify the **Level** in dB from the peak or valley where bandwidth / notch is measured.
5. Press **Execute** or check **Tracking**. Learn more.

### Bandwidth Search

The bandwidth search is a function for determining the bandwidth of the trace, center frequency, cut-off points (on the higher frequency and the lower frequency sides), Q and insertion loss based on the position of the active marker or peak marker. The definitions of the parameters determined through the bandwidth search are shown in below.
Notch Search

The notch search function is used to obtain the bandwidth, center frequency, cutoff points (high-frequency side and low-frequency side), Q and insertion loss of a trace based on the active marker or peak marker position. The notch search function starts from the left side of the active marker position and ends when points that meet the conditions are found.

The figure and the table below shows the definition of parameters obtained by notch search function. The notch value in figure below must be specified by the user.
The following values are displayed for Bandwidth and Notch Search:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bandwidth (BW)</td>
<td>The difference in frequency between the higher frequency cut-off and lower frequency cut-off points (High - Low).</td>
</tr>
<tr>
<td>Center frequency (cent)</td>
<td>Frequency at the middle point between the lower frequency cut-off and higher frequency cut-off points. (High + Low)/2.</td>
</tr>
<tr>
<td>Lower frequency cut-off point (Low)</td>
<td>Lower frequency of 2 measurement points, both separated by the defined bandwidth / notch value from the active marker position.</td>
</tr>
<tr>
<td>Higher frequency cut-off point (High)</td>
<td>Higher frequency of 2 measurement points, both separated by the defined bandwidth / notch value from the active marker position.</td>
</tr>
<tr>
<td>Q</td>
<td>Ratio of Center Frequency to Bandwidth (Center Frequency / Bandwidth).</td>
</tr>
<tr>
<td>Insertion loss (loss)</td>
<td>The measured value of the position of the center frequency at the time the bandwidth/notch search is executed.</td>
</tr>
</tbody>
</table>

- Bandwidth / Notch Search can be used ONLY with Log Mag display format.
To use Bandwidth Search on a peak or valley other than the maximum or minimum values, change the Search Domain.

Compression Search

Uses the active marker to find the specified gain Compression Level.

Note: Valid ONLY for S21 (Gain) measurements with a Power Sweep.

How to create Compression Search

Using Hardkey/SoftTab/Softkey

1. Press Search > Comp & Sat.
2. Click Compression Search to turn ON/OFF.
3. Specify the Comp Level in dB.
4. Optionally click Tracking to search for the specified compression level with each sweep. Learn more.

Using a mouse

1. Move a cursor on a marker.
2. Right-click on the maker and then select Search....
3. From Search Type of Marker Search dialog box, select Compression.
4. Enter the Y-axis (Power OUT) difference between the first point and the compression marker.
5. Press Execute or check Tracking. Learn more.

Linear gain is defined as the Y-axis value (gain) of the first data point of the Search Domain (Full Span by default).

Marker > N - X-axis value and Y-axis value.

Comp Pin - Input power (marker X-axis value).
Comp Pout - Output power (Pin + gain).

Comp Level - Compression level found.

- When Discrete is OFF (default setting), the marker finds the exact specified compression, interpolated between the two closest data points and calculates the Comp Pin and Comp Pout value for that point.

- The marker can move from one actual measurement point to another. Because it is interpolated, it can also move in the space between measurement points.

- When Discrete is ON (not interpolated), the marker resides on the closest data point to the requested compression level.

- A marker moves only between actual measurement points. When a specific marker stimulus value is specified as a numerical value, the marker is placed at the measurement point closest to the specified value. A marker that is placed between interpolated points with the discrete mode off automatically moves to the nearest measurement point when the discrete mode is turned on.

Comp. Not Found - Displayed when the requested compression level is not found.

About PSAT and PNOP Search

Compression measurements based on the Pout vs Pin curves are common in the satellite test industry. In the case of Travelling Wave Tube (TWT) amplifiers, PSAT markers identify the normal operating point near saturation, and the amplifiers are operated with the power slightly backed-off approximately 0.03 to 0.1 dB. For TWT amplifiers, the saturation curve always "folds over" and produces a maximum power out.

For Solid State Power Amplifiers (SSPA), the saturation is not as well defined. A common reference is the Normal Operating Point, which is a power backed-off by 8 to 10 dB from the maximum power. In this case, the normal operating point marker replaces the Psat with the PNOP values. Also, because the backoff is important, the backoff output and input powers are displayed (PBO Out), (PBO in) as well as gain at back off (PBO Gain).

Power Saturation (PSAT) Search

If the Marker: Use single marker for marker search preference is cleared, this search uses Markers 1, 2, and 3 to quickly identify output power saturation parameters of an amplifier. If the Marker: Use single marker for marker search preference is set, then only one marker is used for the search and 2 notational markers are displayed. The notational markers may not be moved. These markers are for
display purposes only.

Back-off is a point at which the output power is sufficiently lower than the saturated output power so that the device under test behaves in a more linear fashion.

**Note:** Valid ONLY for Power IN vs Power OUT measurements.

### How to make Power IN (X-axis) vs Power OUT (Y-axis) measurement

#### Using Hardkey/SoftTab/Softkey

1. Press **Preset**.
2. Press **Sweep > Main > Sweep Type**.
3. Select **Power Sweep**.
4. Press **Trace > Trace Setup > Measure...** and set **Trace Meas** to "B" Receiver.
5. Connect DUT input to port 1.
6. Connect DUT output to port 2.

### How to create PSAT Search

#### Using Hardkey/SoftTab/Softkey

1. Press **Search > Comp & Sat**.
2. Click left side **Saturation Search** small button to turn ON/OFF.
3. For **PMax Back-Off**, enter the Y-axis (Power OUT) difference between the Max Power marker (3) and the Back-off marker (2).
4. Optionally click **Tracking** to search for the specified power saturation level with each sweep. Learn more.

#### Using a mouse

1. Move a cursor on a marker.
2. Right-click on the marker and then select **Search > Search**....
3. From **Search Type** of Marker Search dialog box, select **Power Saturation**.
4. For **PMax Back-Off**, enter the Y-axis (Power OUT) difference between the Max Power marker (3) and the Back-off marker (2).
5. Press **Execute** or check **Tracking**. Learn more.
This setting uses three markers to calculate and display 10 values.

The three markers:

- Marker 1: Linear gain; the first data point in the sweep.
- Marker 2: Specified output power Back-off from max power.
- Marker 3: Max Power output; usually the last data point.

The 9 displayed values:

<table>
<thead>
<tr>
<th>Param</th>
<th>Description</th>
<th>Calculated from...</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSat Out</td>
<td>Output power at the saturation point.</td>
<td>Marker 2 Y-axis value</td>
</tr>
<tr>
<td>PSat In</td>
<td>Input power at the saturation point.</td>
<td>Marker 2 X-axis value</td>
</tr>
<tr>
<td>Gain Sat</td>
<td>Gain at the saturation point.</td>
<td>Psat Out - Psat In</td>
</tr>
<tr>
<td>Comp Sat</td>
<td>Compression at the saturation point.</td>
<td>Gain Sat - Gain Linear</td>
</tr>
<tr>
<td>PMax Out</td>
<td>Maximum output power.</td>
<td>Marker 3 Y-axis value</td>
</tr>
<tr>
<td>PMax In</td>
<td>Input power at the maximum output power.</td>
<td>Marker 3 X-axis value</td>
</tr>
<tr>
<td>Gain Max</td>
<td>Gain at the maximum output power.</td>
<td>PMax Out - PMax In</td>
</tr>
<tr>
<td>Comp Max</td>
<td>Compression at the maximum output power.</td>
<td>Gain Max - Gain Linear</td>
</tr>
<tr>
<td>Gain Linear</td>
<td>Linear gain at the first data point.</td>
<td>Marker 1 - Y-axis value MINUS X-axis value</td>
</tr>
</tbody>
</table>

- **Comp. Not Found** is displayed when the requested Back-off point is not found.
When Discrete marker is NOT selected (the default setting), the three markers find an interpolated value between the two closest data points.

When Discrete marker is selected (NOT interpolated), the three markers reside on the closest data points.

Power Normal Operating Point (PNOP) Search

If the Marker: Use single marker for marker search preference is cleared, this search uses Markers 1, 2, 3, and 4 to quickly identify Normal Operating Point parameters of an amplifier. If the Marker: Use single marker for marker search preference is set, then only one marker is used for the search and 2 notational markers are displayed. The notational markers may not be moved. These markers are for display purposes only.

Back-off is a point at which the output power is sufficiently lower than the saturated output power so that the device under test behaves in a more linear fashion.

The power normal operating point is the output power where the input is offset from the back-off input power by the Pin Offset.

Note: Valid ONLY for Power IN vs Power OUT measurements.

See Power Saturation to learn how to make a Power IN (X-axis) vs Power OUT (Y-axis) measurement.

How to create PNOP Search

**Using Hardkey/SoftTab/Softkey**

2. Click left side Normal OP Search small button to turn ON/OFF.
3. For Back-Off, enter the Y-axis (Power OUT) difference between the Max Power marker (3) and the Back-off marker (2).
4. For Pin Offset, enter the X-axis (Power IN) difference between Back-off marker (2) and PNOP marker (4).
5. Optionally click Tracking to search for the specified power normal operating point level with each sweep. Learn more.

**Using a mouse**

1. Move a cursor on a marker.
2. Right-click on the marker and then select Search > Search....
3. From Search Type of Marker Search dialog box, select Normal Operating Pt.
4. For Back-Off, enter the Y-axis (Power OUT) difference between the Max Power marker (3) and the Back-off marker (2).
5. For Pin Offset, enter the X-axis (Power IN) difference between Back-off marker (2) and PNOP marker (4).
6. Press Execute or check Tracking. Learn more.
This setting uses **four** markers to calculate and display 12 values.

The **four** markers:

- Marker 1: Linear gain; the first data point in the sweep.
- Marker 2: Max Output Power MINUS the specified Output (Y-axis) **Back-off** value in dB.
- Marker 3: Max Output Power; usually the last data point in the sweep.
- Marker 4: X-axis value of Back-off (Marker 2) plus the **Pin Offset** (X-axis) value in dB.

The 11 displayed values:

<table>
<thead>
<tr>
<th>Param</th>
<th>Description</th>
<th>Calculated from...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pnop Out</td>
<td>Output power at the power normal operating point.</td>
<td>Marker 4 Y-axis value</td>
</tr>
<tr>
<td>Pnop In</td>
<td>Input power at the power normal operating point.</td>
<td>Marker 4 X-axis value</td>
</tr>
<tr>
<td>Pnop Gain</td>
<td>Gain at the power normal operating point.</td>
<td>Pnop Out - Pnop In</td>
</tr>
<tr>
<td>Pnop Comp</td>
<td>Compression at the power normal operating point.</td>
<td>Pnop Gain - Linear Gain*</td>
</tr>
<tr>
<td>PMax Out</td>
<td>Maximum output power.</td>
<td>Marker 3 Y-axis value</td>
</tr>
<tr>
<td>PMax In</td>
<td>Input power at the maximum output power.</td>
<td>Marker 3 X-axis value</td>
</tr>
<tr>
<td>Gain Max</td>
<td>Gain at the maximum output power.</td>
<td>PMax Out - PMax In</td>
</tr>
<tr>
<td>Comp Max</td>
<td>Compression at the maximum output power.</td>
<td>Gain Max - Linear Gain*</td>
</tr>
<tr>
<td>PBO Out</td>
<td>Output power at the back-off point.</td>
<td>Marker 2 Y-axis</td>
</tr>
<tr>
<td>PBO In</td>
<td>Input power at the back-off point.</td>
<td>Marker 2 X-axis</td>
</tr>
<tr>
<td>PBO Gain</td>
<td>Gain at the back-off point.</td>
<td>PBO Out - PBO In</td>
</tr>
</tbody>
</table>
Linear Gain (not shown): Marker 1 - Y-axis value MINUS X-axis value

- PNOP Not Found is displayed when the requested back-off level is not found.
- When Discrete marker is NOT selected (the default setting), the four markers each find an interpolated value between the two closest data points.
- When Discrete marker is selected (NOT interpolated), the four markers each reside on the closest data point.

Search Range

Search domain settings restrict the stimulus values (X-axis for rectangular format) to a specified span. Set the User Start and User Stop stimulus settings of these User spans. If User Start is greater than User Stop, the marker will not move. Learn how to set Search Range.

- The default domain of each new marker is "full span".
- There are 16 user-defined domains for every channel.
- The user-defined domains can overlap.
- More than one marker can use a defined domain.
- Search Domain settings are shared with Trace Statistics User Ranges.

The graphic below shows examples of search domains.
A search range will be indicated with a pair of small, outlined triangles sitting on the X-axis. Although there can be multiple search ranges in use on various markers, only the current-selected search range for the active marker is displayed. This rule prevents the possibility of the X-axis being cluttered with many search range triangles. This rule applies even when there are multiple traces in a window.

Only one search range will be displayed on a grid at any time. The displayed search range will correspond to the active trace and active marker. The color of the range indicators will match that of the active trace.

Range indicators will appear automatically when appropriate and cannot be disabled. The mouse or touchscreen can't be used to “click-and-drag” the position of the range indicators which will alter the search range definition.

Search Within

The zoomed frequency range becomes the **User 16 Search Domain span**.

A marker is created if not already present on the trace. If markers are already present on the trace, the lowest marker is moved to the found value.

1. Left-click the mouse or use a finger, then drag across a portion of a trace.
2. Release the mouse or lift the finger.
3. Select **Search Within**.
4. Then choose from the following:

   - **Max** - A marker moves to the HIGHEST value within the zoomed range.
   - **Min** - A marker moves to the LOWEST value within the zoomed range.
- **Target** - A marker moves to the first value within the zoomed range that is currently set in the Marker Search 'Target' setting. The same Discrete Marker rules apply as those for the standard Target Marker Search.

**Marker Functions - Change Instrument Settings**

The following settings change the relevant VNA settings to the position of the active maker.

### How to change Instrument settings using markers

**Using Hardkey/SoftTab/Softkey**

1. Press **Marker -> Functions**.

**Using a mouse**

1. Move a cursor on a marker.
2. Right-click on the marker and then select **Functions**.
3. Select the desired search function.
4. Click **Function...** to show the Marker Function Dialog box.

### Marker Function dialog box help

![Marker Function Dialog Box]

**Note:** Marker Functions do not work with channels that are in **CW** or Segment Sweep mode.

- **Marker => Start**  Sets the start sweep setting to the value of the active marker.
- **Marker => Stop**   Sets the stop sweep setting to the value of the active marker.
- **Marker => Center** Sets the center of the sweep to the value of the active marker.
- **Marker => Ref Level** Sets the screen reference level to the value of the active marker.
- **Marker => Delay** The phase slope at the active marker stimulus position is used to adjust the line length to the receiver input. This effectively flattens the phase trace around the active marker. Additional Electrical Delay adjustments are required on devices without constant group delay over
the measured frequency span. You can use this to measure the electrical length or deviation from linear phase.

This feature adds phase delay to a variation in phase versus frequency; therefore, it is only applicable for ratioed measurements. See Measurement Parameters.

**Marker =>Span** Sets the sweep span to the span that is defined by the delta marker and the marker that it references. Unavailable if there is no delta marker.

**Marker =>CW Freq** Sets the CW frequency to the frequency of the active marker. NOT available when the channel is in CW or Power Sweep. Use this function to first set the CW Frequency to a value that is known to be within the current calibrated range, THEN set Sweep Type to Power or CW.

**Marker =>IM Spectrum** This feature is supported in Swept IMD/IMDX measurement classes (channels) ONLY. The generated IMD tones and 2 input tones are measured. This feature provides a plot of the fixed input tones and IMD tones across the receiver frequency.

**Marker =>SA** This feature is supported in Standard, SMC or Swept IMD measurement classes (channels) ONLY. In this section, these are called NA channels.

With a marker residing on a trace in an NA channel, Marker=>SA creates a new SA channel in full frequency span. A marker is created on the trace at the same frequency as the NA channel trace. This is a quick way to see the frequency spectrum of the NA channel at a specific frequency. Learn more.

**Note:** Some Marker Functions do not work with channels that are in certain Sweep Types.

<table>
<thead>
<tr>
<th>Marker Function</th>
<th>Lin/Log Freq.</th>
<th>Sweep Type</th>
<th>Segment</th>
<th>Power</th>
<th>CW Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start, Stop, Center</td>
<td>F</td>
<td>S</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Span</td>
<td>S</td>
<td>S</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ref Level</td>
<td>F</td>
<td>S</td>
<td>S</td>
<td></td>
<td>S</td>
</tr>
<tr>
<td>Delay</td>
<td>F</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>CW Freq.</td>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td>S</td>
</tr>
</tbody>
</table>

F: Available in both Standard and SMC classes

S: Available in only Standard Class

---

**SA Analysis Markers**
The following markers are specific to Spectrum Analyzer measurement channels.

<table>
<thead>
<tr>
<th>Spectrum Analysis markers</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Using Hardkey/SoftTab/Softkey</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press <strong>Marker &gt; SA Analysis</strong></td>
<td>1. Click <strong>Response</strong>.</td>
</tr>
<tr>
<td><strong>Using a mouse</strong></td>
<td>2. Select <strong>Marker</strong>.</td>
</tr>
<tr>
<td>1. Click <strong>Response</strong>.</td>
<td>3. Select the desired marker.</td>
</tr>
</tbody>
</table>

**Programming Commands**
**Band Power** – Provides a readout of the total power obtained by integrating over the specified frequency range defined by Band Span, centered at the marker position.

The *Average* detector type makes the display more consistent with Band Power markers. With this detector type the marker readouts and the estimated power values or delta from the display have a good match. Many legacy spectrum analyzers need to set the detector type to average to do band power measurements. PNA-SA does not need to select the average detector to compute the right band power measurements, it’s only
useful to check the values between marker readouts and display estimates.

**Band Density** – Selections are **Off**, **Noise**, **Power**, **Tone**, **NPR**, or **ACPR**

- **Noise** – Provides a readout of the noise-power density (dBm/Hz) at the marker position, obtained by integrating over the specified frequency range defined by Band Span, and normalizing to the same frequency range.

- **Power** – Provides a readout of the power density (dBm/Hz) at the marker position, obtained by integrating over the specified frequency range defined by Band Span, and normalizing to the bandwidth defined by Equivalent Span.

- **Tone** – Provides a readout of the power density (dBm/tone) at the marker position, obtained by integrating over the specified frequency range defined by Band Span, and normalizing to the bandwidth defined by Density BW and Equivalent Span. When Density BW is set to the tone spacing of the multitone signal, and Equivalent Span is the same as Band Span, Tone Density gives dBm/tone. This feature is useful for measuring the power density of the signals inside and outside the notch for Noise Power Ratio (NPR) measurements. For example, for measuring a 10 MHz notch with -50 dBm band power and 100 kHz tone spacing (number of tones = 10 MHz/100 kHz = 100), setting the Band Span and Equivalent Span to 10 MHz and the Density BW to 100 kHz, the result shows that each tone would be -70 dBm (-50 dBm - 10\*Log10[100]). Tone band density can also be used to measure Adjacent-Channel-Power-Ratio (ACPR).

- **NPR** – Uses the same calculation method as **Tone**, but provides a simpler readout consisting only of marker frequency and NPR value.

- **ACPR** - Uses the same calculation method as **Tone**, but provides a simpler readout consisting only of marker frequency and ACPR value.

**Note:** The band-density markers can be set to Delta ON mode which is very useful for NPR and ACPR, which are defined as distortion power in a specified band divided by in-band power. For these ratioed measurements, the reference marker can be set to Tone Band Density, with a Band Span that covers the in-channel bandwidth (for NPR, see Equivalent Span below to account for the reduction in channel power due to the notch). The ACPR and NPR delta markers will have a Band Span defined by the adjacent-channel bandwidth or the notch bandwidth respectively.

**Band Span** – Sets the frequency range over which Band Power and Band Density measurements are calculated.

**Equivalent Span** – Sets the frequency range over which Power and Tone Band Density measurements are normalized. This feature is useful for calculating the power density of a wideband signal with a notch (or multiple notches), where the power in the notch(es) is significant and cannot be ignored. Adjusting the Equivalent Span is essential for calculating the correct tone powers of a noise-power-ratio (NPR) signal. For example, for a 100 MHz signal with a 10 MHz notch, the Equivalent Span would be 90 MHz. For measuring the power in the notch, the Equivalent Span and Band Span are the same (10 MHz in this example).
Density BW – Sets the frequency range over which Tone Band Density measurements are calculated.

Occupied BW – Enables/disables the frequency range that contains a defined percentage of the overall band power as specified by OBW Percent.

OBW Percent – Sets the percentage used to calculate Occupied BW.

NPR Measurement Example

Noise-Power-Ratio (NPR) measurements are displayed using a fixed reference marker (set to Tone Band Density) that calculates the power density of the main signal, and a delta marker (set to NPR Band Density) that calculates the power density of the distortion signals within the notch. The Equivalent Span of the main signal is set to the Band Span minus the notch bandwidth – for example, for a 100 MHz wide NPR signal with a 10 MHz notch, the Equivalent Span would be 100 MHz - 10 MHz = 90 MHz. For the notch in this same example, the Equivalent Span would be set to 10 MHz.

Calculations for Band Power and Density Markers

All the band-power and band-density markers are computed from Discrete Fourier Transform (DFT) data, before the data is processed into the display points shown on the screen as traces. As a result, these markers are not affected by the number of display points, or by the detection algorithm used (average, sample, peak, etc.). There are two ways to access the DFT data if a user wishes to do their own calculations of power, or to perform other mathematical processing:

1. Set the Detector Type to Bypass, which works if the number of DFT points is below 100,003 (the trace-size limit).

2. Use the SA Data feature (accessed via the Advanced tab of the SA Setup dialog). In general, the display points do not provide an accurate representation of band-power or band-density values.

There are many more DFT data points than display points, and the data reduction performed by the detection algorithm causes band-power accuracy to degrade at signal discontinuities, where there is a wide change in power values. In addition, when using peak detection, the algorithm takes the largest power value of the DFT data-set to produce a single display point, so the displayed value does not represent the average power of the band covered for that point. For these reasons, it is impossible to compute an accurate band-power or band-density value from display points using the peak detector. This is why the SA band-power and band-density markers are computed from the underlying DFT data.

SA Analysis Markers (Option S93070xB9x070B Modulation Distortion only)
The following markers are specific to Modulation Distortion measurement channels.

**Modulation Distortion markers**

*Using Hardkey/SoftTab/Softkey*

1. Press **Marker > SA Analysis**.
**Band Function** - Selections are **Off, Marker Noise, Band Power,** or **Band Density**.

**Marker Noise** - Noise integrated over the frequency span defined by **Band Span**.

**Band Power** - Power integrated over the frequency span defined by **Band Span**. This is the total power of the signal in dBm or Watts.

**Band Density** - Band power expressed in Watts divided by the integrating bandwidth specified by **Band Span**. The density value is constant as span is decreased.

**Band Span** - The Band Span is used to determine total signal power within a specified frequency span. For example, to calculate the total power of a signal composed of 100 tones spaced 1 MHz apart over a 100 MHz span, the signal power would be integrated over a span of 100 MHz.

**Occupied BW** – Enables/disables the frequency range that contains a defined percentage of the overall band power as specified by OBW Percent.

**OBW Percent** – Sets the percentage used to calculate Occupied BW.

---

**Marker Display**

The marker display dialog allows you to change how markers and the associated readout is displayed on the VNA screen. Several marker display features also apply to **Statistics** display.

---

### How to change Marker Display settings

#### Using Hardkey/SoftTab/Softkey

1. Press **Marker > Marker Setup > Marker Display**....

OR

1. Press **Display > Display Settings > Customize Display**....

2. Select **Markers** tab.

#### Using a mouse

1. Move a cursor to the marker data area on top right corner of grid box.

2. Right click on the marker data display area.

3. Select **Marker Display**....

OR

1. Right click on any window area.

2. Click **Customize Display**....

3. Select **Markers** tab.

---

857
The following settings apply to readouts of ALL currently-displayed marker, bandwidth, and trace statistics.

These settings revert to their defaults on Preset but ARE stored with Instrument State and User Preset.

**Marker Readout**

- Checked - Shows readout information.
- Cleared - Shows NO readout information.

**Large Readout**

- Checked - Shows the marker readout in large font size for easy reading. However, all readout lines may not be visible.
- Cleared - Shows the marker readout in normal font size.

**Active Trace Only**

- Checked - Shows the marker readout for the active trace only.
- Cleared - Shows all marker readouts.

**Readouts Per Trace**
Choose the quantity of marker readouts to show in the window for each trace. Choose to display up to 16 readouts per trace, up to 20 readouts per window. When more markers are present than the specified quantity of readouts, the marker numbers for which readouts are displayed can change depending on the marker number that is active. Readouts Per Trace can be set independently for each window.

**Symbol**

Choose from the following marker symbols.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Triangle" /></td>
<td>Triangle</td>
</tr>
<tr>
<td><img src="image" alt="Flag" /></td>
<td>Flag</td>
</tr>
<tr>
<td><img src="image" alt="Line" /></td>
<td>Line</td>
</tr>
</tbody>
</table>

Line symbols are NOT used on Smith or Polar display formats.

Symbols can be set independently for each window.

**Symbols Above Trace**

- Cleared - ONLY the active marker is displayed above the trace. Inactive markers are displayed below the trace.
- Checked - ALL marker symbols are displayed above the trace. The active marker is always filled solid.

**Decimal Places**

Choose the marker readout resolution to display. These values also apply to the readouts that are displayed in the marker table. Decimal Places can be set independently for each window.

- **Stimulus** (X-axis) - Choose from 2 to 6 places after the decimal point. Default is 3.
- **Response** (Y-axis) - choose from 1 to 4 places after the decimal point. Default is 2.

**Readout Position**

Choose where to place the marker readouts. Marker readouts are right-justified on the specified X-axis and Y-axis position. The default position (10.0, 10.0) is the upper-right corner of the grid. Position (1.0,1.0) is the lower-left corner. Readout position can also be set independently for each window.

**Note:** Readout Position can also be changed using a mouse by left-clicking on the top readout and...
Marker Table

You can display a table that provides a summary of marker data for the active trace. The marker data is displayed in the specified format for each marker.

**How to view the Marker Table**

Using **Hardkey/SoftTab/Softkey**

1. Press **Marker > Marker Setup > Marker Table.**
Using Math / Memory Operations

You can perform four types of math on the active trace versus a memory trace. In addition three statistics (Mean, Standard Deviation and Peak to Peak) can be calculated and displayed for the active data trace.

- Trace Math
- Trace Statistics
- AM Distortion
- Trace Deviation

**Note:** Trace Math (described here) allows you to quickly apply one of four math operations using memory traces. *Equation Editor* allows you to build custom equations using several types of traces from the same, or different channels.

**Other Analyze Data topics**

**Trace Math**

To perform any of the math operations, you must first store a trace to memory. You can display the memory trace using the *View* options.

Trace math is performed on the complex data before it is formatted for display. See the *VNA data processing map*.

Markers can be used while viewing a memory trace.

**How to select Trace Math**

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Math &gt; Memory</strong>.</td>
<td>1. Right click on any trace status area above the grid box.</td>
</tr>
<tr>
<td>2. <strong>Normalize</strong>, available only from the Memory menu, (not on the Math / Memory dialog), performs the same function as <strong>Data=&gt;Memory</strong>, then <strong>Data / Memory</strong>.</td>
<td>2. Select on <strong>Memory</strong>....</td>
</tr>
</tbody>
</table>
Normalize, available only from the Memory menu, (not on the Math / Memory dialog), performs the same function as Data=>Memory, then Data / Memory.

Data=>Memory Puts the active data trace into memory. You can store one memory trace for every displayed trace.

Note: Many VNA features are NOT allowed on Memory traces. For example, Memory traces can NOT be saved to any file type (PRN, SNP, CTI, CSV, MDF). However, you can restore a memory trace to a data trace using the Memory-to-Data utility at the http://na.support.keysight.com/pna/apps/applications.htm website.

Data Math

All math operations are performed on linear (real and imaginary) data before being formatted. See the VNA Data flow.

Data (or OFF) Does no mathematical operation.

Data / Memory - Current measurement data is divided by the data in memory. Use for ratio comparison of two traces, such as measurements of gain or attenuation. Learn more.

Data – Memory - Data in memory is subtracted from the current measurement data. For example, you can use this feature for storing a measured vector error, then subtracting this error from the DUT measurement. Learn more.
**Data + Memory** - Current measurement data is added to the data in memory. Learn more.

**Data * Memory** - Current measurement data is multiplied by the data in memory. Learn more.

<table>
<thead>
<tr>
<th>Trace View Options</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Trace</strong></td>
</tr>
<tr>
<td><strong>Memory Trace</strong></td>
</tr>
<tr>
<td><strong>Data and Memory Trace</strong></td>
</tr>
</tbody>
</table>

**Interpolate**

**Note:** The E5080A and M9485A do not support this function.

After performing a Data->Memory operation, memory interpolation controls whether the memory data is interpolated or not if the start frequency, stop frequency, or Number of Points is subsequently changed. Using the GUI control, interpolate applies to the currently active measurement. When using the remote interfaces (SCPI or COM), the commands apply to the specified measurement.

**Note:** Interpolate does not support the 8510 Mode.

The PNA will return to a default interpolation state after a Preset, creating a new trace, or closing the PNA application. The default interpolation state is set in the Preferences dialog by checking or unchecking the **Memory: Interpolate ON is the default condition** preference. The factory default is unchecked. The default can also be set using the remote interfaces (SCPI or COM).

- When unchecked, after a Data->Memory operation the memory trace's x-y positions will not change if the start or stop frequency is subsequently changed. In addition, if the Number of Points in the sweep is changed after a Data->Memory operation, the memory trace will be invalidated and disappear. If the Number of Points is changed while using Data Math, the Memory trace will be invalidated and Data Math will be forced to the "Off" condition.

- When checked, after a Data->Memory operation the memory trace's x-y positions will be interpolated if the start or stop frequency is subsequently changed. In addition, if the Number of Points in the sweep is changed after a Data->Memory operation, the memory trace will be interpolated.
Note: The PNA will not extrapolate to stimulus values beyond the range that was present at the time of the Data->Memory operation. Instead, the Memory data will be invalidated if the stimulus values exceed the original range.

Note: If Interpolate is checked (ON) and stimulus conditions are different than they were at the time of Data->Memory operation, unchecking (OFF) Interpolate will cause the Memory trace to be either updated (using both original and current stimulus settings) or invalidated (if Number of Points changed since Data->Memory operation). The Memory trace will remain disabled until either Interpolate is checked (ON) or the stimulus settings corresponding to the Data->Memory operation are restored.

(Data / Memory) and (Data - Memory)

(Data / Memory) and (Data - Memory) math operations are performed on linear data before it is formatted. Because data is often viewed in log format, it is not always clear which of the two math operations should be used. Remember: dividing linear data is the same as subtracting logarithmic data. The following illustrates, in general, when to use each operation.

Use **Data / Memory** for normalization purposes, such as when comparing S21 traces "before" and "after" a change is made or measurement of trace noise. In the following table, the Data/Mem values intuitively show the differences between traces. It is not obvious what Data-Mem is displaying.

<table>
<thead>
<tr>
<th>S21 values to compare</th>
<th>Data/Mem</th>
<th>Data-Mem</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5 dB and 0.6 dB</td>
<td>0.1 dB</td>
<td>-39 dB</td>
</tr>
<tr>
<td>0.5 dB and 0.7 dB</td>
<td>0.2 dB</td>
<td>-33 dB</td>
</tr>
</tbody>
</table>

Use **Data - Memory** to show the relative differences between two signals. Use for comparison of very small signals, such as the S11 match of two connectors.

In the following table, Data/Mem shows both pairs of connectors to have the same 2 dB difference. However, the second pair of connectors have much better S11 performance (-50 and -52) and the relative significance is shown in the Data-Mem values.

<table>
<thead>
<tr>
<th>S11 values to compare</th>
<th>Data/Mem</th>
<th>Data-Mem</th>
</tr>
</thead>
<tbody>
<tr>
<td>-10 dB and -12 dB</td>
<td>2 dB</td>
<td>-24 dB</td>
</tr>
<tr>
<td>-50 dB and -52 dB</td>
<td>2 dB</td>
<td>-64 dB</td>
</tr>
</tbody>
</table>

**Data * Memory and Data + Memory**
Use **Data * Memory and Data + Memory** to perform math on an active data trace using data from your own formulas or algorithms rather than data from a measurement. For example, if you want to simulate the gain of a theoretical amplifier placed in series before the DUT, you could do the following:

1. Create an algorithm that would characterize the frequency response of the theoretical amplifier.

2. Enter complex data pairs that correspond to the number of data points for your data trace.

3. Load the data pairs into memory with SCPI or COM commands. The analyzer maps the complex pairs to correspond to the stimulus values at the actual measurement points.

4. Use the **data + memory** or **data * memory** function to add or multiply the frequency response data to the measured data from the active data trace.

**Note:** The data trace must be configured before you attempt to load the memory.

### Trace Statistics

You can calculate and display statistics for the active data trace. These statistics are:

- Mean
- Standard deviation
- Peak-to-peak values

You can calculate statistics for the full stimulus span or for part of it by using User Ranges.

You can define up to 16 user ranges per channel. These user ranges are the same as the **Search Domain** specified for a marker search in that same channel. They use the same memory registers and thus share the same stimulus spans.

The user ranges for a channel can overlap each other.

A convenient use for trace statistics is to find the peak-to-peak value of passband ripple without searching separately for the minimum and maximum values.

The trace statistics are calculated based on the format used to display the data.

- **Rectangular data formats** are calculated from the scalar data represented in the display
- **Polar or Smith Chart** formats are calculated from the data as it would be displayed in Log Mag format
See how to make Trace Statistics display settings.

### How to activate Trace Statistics

Using Hardkey/SoftTab/Softkey

1. Press **Math > Analysis > Statistics...**

#### Trace Statistics dialog box help

![Trace Statistics dialog box](image)

See how to make Trace Statistics display settings.

**Statistics** Check to display mean, standard deviation, and peak to peak values for the active trace.

**Span** Specifies the span of the active trace where data is collected for a math operation. You can select Full Span, or define up to 16 user spans per channel with Start and Stop. You can also define the user spans from the Search Domain selector on the Marker Search dialog box.

**Start** Defines the start of a user span.

**Stop** Defines the stop of a user span.

Learn more about Trace Statistics (scroll up)

### AM Distortion

Calculate amplitude modulation or phase modulation per trace. This feature is supported on all standard channels and all application channels. This feature can be enabled on any sweep type, but should be applied to a power sweep for useful results. It operates on the underlying data of the trace and will ignore the current format of the trace. It will not perform additional sweeps or collect additional data. AM distortion is only applied to rectilinear formats (for example, Polar, Smith).
How to select AM Distortion

**Using Hardkey/SoftTab/Softkey**

1. Press **Math > Analysis > AM Distortion.**

**Using a mouse**

1. Click on **Response.**
2. Select **Math then AM Distortion.**
3. Select on **Setup....**

---

**AM Distortion** dialog box help

**AM Distortion**

**Off** - Turns off AM Distortion.

**AM-PM** - Displays the phase distortion in either degrees or deg/dB. If the **Y-axis = Slope Calculated Over Aperture** mode is enabled, then the unit is deg/dB. Otherwise, it is in degrees.

**AM-AM** - Displays the amplitude distortion in either dB or dB/dB. If the **Y-axis = Slope Calculated Over Aperture** mode is enabled, then the unit is dB/dB.

**Y-axis = Slope Calculated Over Aperture** Check to calculate the slope over the specified aperture. For AM-PM, the phase slope is shown in deg/dB. For AM-AM, the gain slope is shown in dB/dB. If this mode is not enabled, then the phase or power is normalized using the median linear gain of the first 30% of the trace.

**X-axis = Backoff For Compression Level** Check to calculate compression using the median gain from the first 30% of the trace as the linear gain. The compression point is determined based off this median linear gain value. The x-axis will be shifted but retain the same power range. The back-off from compression is 5 dB. The text field next to this setting specifies the compression level, not the back-off level. By default, the back-off level is 5 dB and cannot be changed.
If the compression point cannot be found, then a warning is displayed in the status bar. An approximate power level of the compression point is calculated with this formula:

\[
\text{Target compression power} = \text{input stop power} + 10\log\left(\frac{\text{compression level}}{\text{compression level at stop power}}\right).
\]

**Trace Deviation**

You can calculate deviation from a least-best fit line. The choices are:

- Off
- Linear - Fit to 1st order curve minimizing RSS deviation.
- Parabolic - Fit to 2nd order curve minimizing RSS deviation.
- Cubic - Fit to 3rd order curve minimizing RSS deviation.

**How to activate Trace Deviation**

**Using** Hardkey/SoftTab/Softkey

1. Press **Math > Analysis > Trace Deviation**
On the Trace Math dialog, check 8510 Mode to simulate the Keysight 8510 data processing chain as it pertains to Trace Math and Memory. This setting applies to all channels. When the box is checked or cleared, the analyzer performs an Instrument Preset and retains its setting through subsequent Instrument Presets.

This setting can be saved as part of an instrument state. However, when recalled, this setting is assumed only temporarily. When a subsequent analyzer Preset is performed, the analyzer reverts to the setting that was in effect before the state was recalled.

You can set a preference to always use 8510 mode.

This image represents the relevant portion of the data flow.

See the entire analyzer data processing chain.

A settings change in any of the operations that occur after the Memory operation on the above analyzer Data Flow diagram changes both the Data trace and the Memory trace. For example, after storing a data trace to memory, when you change the format for the Data Trace, the format for the Memory Trace is also changed to the same setting.

How to turn ON/OFF 8510 mode

Using Hardkey/SoftTab/Softkey

1. Press Math > Memory > 8510 Mode.

No programming are available for this feature
Equation Editor

Equation Editor allows you to enter an algebraic equation that can mathematically manipulate measured data. The results are displayed as a data trace. Data that is used in the equation can be from the same or different channels.

- Overview
- How to start Equation Editor
- Using Equation Editor
- Data that is used in Equation Editor
- Trace Settings, Error Correction, and an Example
- Functions and Constants
- Operators used in Equation Editor
- Example Equations
- Saving Equation Editor Data

See Also

Using Noise Power Traces in Equation Editor
Equation Editor and MATLAB
Equation Editor Import Functions
External DC Meter Data Conversion

- BestFit.dll
- EqnErrorTerms.dll
- Expansion.dll

Other 'Analyze Data' topics

Overview
Equation Editor allows you to enter an algebraic equation of standard mathematical operators and functions, referencing data that is available in the analyzer. Once a valid equation is entered and enabled, the display of the active trace is replaced with the results of the equation, and updated in real-time as new data is acquired. For equations that can be expressed with Equation Editor's supported functions, operators, and data, there is no need for off-line processing in a separate program.

For example, enter the equation \( \frac{S_{21}}{1 - S_{11}} \). The resulting trace is computed as each \( S_{21} \) data point divided by one minus the corresponding \( S_{11} \) data point. For a 201 point sweep setup, the computation is repeated 201 times, once for each point.

As another example, suppose you want the analyzer to make a directivity measurement of your 3-port DUT. This is not a native measurement, but can be achieved using the Equation Editor. The desired result is the sum and difference of LogMag formatted traces, expressed as: \( S_{12} + S_{23} - S_{13} \).

Because Equation Editor operates on \textbf{unformatted complex data}, the required equation is:

\[
\text{DIR} = \frac{S_{12} \times S_{23}}{S_{13}}
\]

DIR becomes a display label to help you identify the computed data trace.

On the equation trace, set the format to LogMag.

---

**How to start Equation Editor**

**Using Hardkey/SoftTab/Softkey**

1. Press \textbf{Math > Analysis > Equation Editor}.
Notes

- **Double-click**, or type, the Functions, Operators, and Data to build an Equation.
- Scroll down to learn more about Using Equation Editor

**Equation**: The field in which equations are built. Click the down arrow to the right to use or modify equations that have been previously saved. This is where equations are saved when you press 'Store Equation'.

**Enabled** Check this box to enable the equation that is currently in the Equation field. If the Enabled box is not available, then the equation is not valid. If a data trace is used that is from a different channel than the Equation trace, the channels MUST have the same number of data points to be valid.

**< Backspace** Moves the cursor to the left while erasing characters.

**<-** Moves the cursor to the left without erasing characters.

**->** Moves the cursor to the right without erasing characters.

**Store Equation** Press to save the current equation. To later recall the equation, click the down arrow to the right of the equation.
Delete Equation  Removes the current equation from the drop-down list.

Functions/Constants:  See descriptions of Functions.

Select the "library" of functions to view. The "built-in" library appears by default which includes the standard functions of equation editor. Other functions that can appear here are functions that you have written and imported. Learn more.

Operators:  See descriptions of Operators.

Trace Data: Select from ALL of the currently valid displayed traces on ALL channels. Valid displayed traces are defined as: any trace with the same number of points as the equation trace and the trace cannot reference itself.

Parameter: Select from undisplayed data that is available ONLY from the active channel (same channel as the equation trace). See Data that is used in Equations.

Note: With an external test set enabled, only parameters involving ports 1 through 4 are listed. However, all available parameters can be typed directly into the Equation field.

Use Short Names  Some functions have shortened names that are entered automatically when checked. Both long and short names can be used interchangeably.

Fast Processing  When checked, updates from trace references or marker references will occur once per sweep. The underlying trace will still update normally. Also, if there are no trace references or marker references in the equation, then there is no change in behavior.

A trace reference is when the equation uses another trace (for example, “eq = tr1 + tr2”). A marker reference is when the built-in marker functions mrkx() or mrky() are used.

Enable Matlab  Available when a full MATLAB version is installed by you on the analyzer. Learn more.

Import... Click to launch the Import Functions Dialog box.

Keyboard...: Provided to allow navigation of the entire dialog with a mouse.

Using Equation Editor

1. Pick a trace in which to enter the equation

   • Equation Editor works on the active trace.
Either create a new trace, or click the Trace Status button on an existing trace to make the trace active.

2. Enter an equation

Start Equation Editor See how.

- The equation text can be in the form of an expression \((S21)/(1-S11)\) or an equation \((\text{DIR} = S12 * S23 / S13)\). This topic refers to both types as equations.
- Either type, or double-click the Functions, Operators, and Data to build an equation.
- Functions and Constants ARE case-sensitive; Data names are NOT case sensitive.
- Learn more about referring to data traces.

3. Check for a valid equation

When a valid equation is entered, the Enabled checkbox becomes available for checking. When the Enabled box is checked:

- The Equation Trace becomes computed data.
- The equation is visible on the Trace Status (up to about 10 characters).
- The equation is visible in the trace Title area (up to about 45 characters) when the Equation trace is active.
- The equation is visible in the Status Bar at the bottom of the display. This is updated only after the equation is entered and the Trace Status button is clicked.
- If an equation is NOT valid, and a trace from a different channel is used, make sure the number of data points is the same for both channels.

Learn more about the Functions, Operators, and Data that are used in Equation Editor.

Data that is used in Equation Editor

Definitions

- **Equation trace** A trace in which an equation resides.
- **Referred trace** A trace that is used as data in an equation.

Example: \(eq=\text{Tr}2+\text{S}11\) is entered into \(\text{Tr}1\).

\(\text{Tr}1\) becomes an equation trace.
**Notes**

- Referred traces are processed one data point at a time. For example, the expression S11/S21 means that for each data point in S11 and S21, divide point N of S11 by point N of S21.

- Once an equation is enabled, the trace is no longer identified by its original measurement parameter. It becomes an equation trace.

- An equation trace can NOT refer to itself. For example, an equation in Tr1 cannot refer to trace Tr1.

- Referred traces can be selected from S-Parameters, Receiver data, Memory traces, and application channels. When equation editor is opened in an application channel, the corresponding application parameters can be selected.

  - See note regarding External Test Sets.

  - See Using Noise Power Traces in Equation Editor

**There are three ways to refer to traces:**

The following distinction is important when discussing the three ways to refer to traces/data.

- **Trace** - a sequential collection of data points that are displayed on the screen.

- **Data** - analyzer measurements that are acquired but not displayed. When an equation trace refers to data that is not displayed, the analyzer will automatically acquire the data.

1. Using **TrX** Trace notation (for example, Tr2).

When a trace is created, check "Show Tr Annotation" to see the Tr number of that trace.

  - **Simple** - ALWAYS refers to displayed traces.

  - Must be used for referring to traces in a different channel as the equation trace.

  - All trace settings are preserved in the equation trace. If you do NOT want a trace setting to be used in the equation trace, you must disable it in the referred trace.

  - If the referred trace is error corrected, then that data is corrected in the equation trace.

  - Used to refer to a memory trace (it must already be stored in memory). Append .MEM to the TrX trace identifier. For example, Tr2.mem refers to the memory trace that is stored for Tr2.

2. Using **S-parameter** notation (for example, S11/S21)
- **Convenient** - ALWAYS refers to data that is NOT displayed.

- Refers to data that resides in the same channel as the equation.

- NOT the same as referring to a displayed S11 trace using TrX notation. See Example.
  
  - The referred data includes NO trace settings.
  
  - If correction is applied to the channel, equation editor traces in that channel will attempt to use corrected parameter data regardless if correction is on/off for the measurement. If there is no corrected data available, then raw data will be used. TrX notation always ignores the correction state.

3. Using **Receiver** notation (for example AB_2); NOT case sensitive.

At least one receiver is required, followed by an underscore and a number.

- The letters before the underscore refer to the receivers.
  
  - Letters alone refer to physical receivers.
  
  - Letters immediately followed by numbers refer to logical receivers. Learn more.
  
  - If two receivers are referenced, they are ratioed.

- The number after the underscore refers to the source port for the measurement.

Examples

- AR1_2 = physical receiver A / physical receiver R1 with 2 as the source port.

- a3b4_1 = reference receiver for port 3 / test port receiver for port 4 with 1 as the source port.

Learn more about ratioed and unratioed receiver measurements.

Receiver notation is like S-parameter notation in that:

- Refers to data that is NOT displayed and resides in the same channel as the equation.

- The referred data includes NO trace settings.

- If correction is applied to the channel, equation editor traces in that channel will attempt to use corrected parameter data regardless if correction is on/off for the measurement. If there is no corrected data available, then raw data will be used. TrX notation always ignores the correction state.
When the equation trace refers to a trace on a different channel:

- The trace must already be displayed.
- Must refer to the trace using TrX notation.
- The Equation trace and the referred trace MUST have the same number of data points or the Enable checkbox will not be available.
- The Equation trace is updated when the last referred data in the same channel is acquired. Therefore, to prevent 'stale' data from being used, the Equation trace must be on a higher numbered channel than the referred trace. This is because the analyzer acquires data in ascending channel number order - first channel 1, then channel 2, and so forth. If the Equation trace is on channel 1, and it refers to a trace on channel 2, the Equation trace will update after channel 1 is finished sweeping, using 'old' data for the channel 2 trace.

**Port Extensions and Equation Editor**

When using port extension with an equation, turn Fixturing ON to ensure that the underlying parameters have port extension properly applied. Learn more.

**Trace Settings, Error Correction, and an Example**

This discussion highlights the differences between using S-parameter / Receiver notation and TrX notation when referring to traces. The key to understanding the differences is realizing that S-parameter / Receiver notation ALWAYS refers to data that is NOT displayed.

- **Trace Settings** Normalization, Trace Math, Gating, Phase and Mag Offset, Electrical Delay, Time Domain.
- **Equation Editor** processing occurs on the equation trace immediately after error correction.
- **Referred Data/Trace** (used in the equation) is taken from the following locations:
  - When using TrX notation, data is taken immediately before formatting. These traces are always displayed
and include **Trace Settings**.

- When using **S-parameter / Receiver** notation, data is taken immediately after error correction. This data is NOT displayed and includes **NO** trace settings *(see example)*.

See Equation Editor Notes at [GetData Method](#) or [GetDataByString Method](#).

### Error-correction and Equation Editor

**Using **TrX** notation:**

- The Trace Settings and Error-correction on the referred trace are used in the Equation trace.

- If error correction is NOT ON, then the raw, uncorrected data is used in the equation trace.

- To see if error correction is ON, make the trace active, then see the **Correction level in the status bar**.

- Turning error correction ON/OFF on the equation trace has no meaning. The referred data that is used in the equation is ALWAYS what determines its level of correction.

**Using **S-parameter** and **Receiver** notation:**

- Because the data is not displayed, **NO** trace settings are used in the Equation trace.

- If correction is applied to the channel, equation editor traces in that channel will attempt to use corrected parameter data regardless if correction is on/off for the measurement. If there is no corrected data available, then raw data will be used. **TrX** notation always ignores the correction state.

- When using S-parameter and Receiver notation to refer to a trace on a channel that has been calibrated with a **Response Cal** or **Receiver Cal**, correction can NOT be turned ON, even though the Status Bar indicates otherwise. For example: Tr1 is an S11 measurement with a Response Cal. Tr2 is an equation trace that refers to S11. The Tr2 equation trace is NOT corrected, even though the Status Bar may indicate that it is corrected. However, if Tr2 refers to Tr1 (not S11), the Tr2 equation trace is corrected.

### Example

This example illustrates the differences when referring to a trace using **S-parameter** notation and **TrX** notation:
- **Tr1** is an S11 measurement with no equation, 2-port correction ON, and Time Domain transform ON.

- **Tr2** is an equation trace that refers to **Tr1**. Tr2 is corrected because Tr1 is corrected. Tr2 is transformed because Tr1 is transformed. If transform is turned ON for Tr2, the data will be transformed AGAIN, which results in "unusual" data.

- **Tr3** is an equation trace that refers to **S11**. This is NOT the same as referring to Tr1. The S11 trace that is referred to is a different instance of S11 that is NOT displayed, and has NO trace settings. Notice that Tr3 data is NOT transformed, although Tr1 is transformed. Correction for **Tr3** can be turned ON and OFF because a calibration was performed on the channel in which the S11 trace resides.

- **Note**: X-axis annotation of the Equation trace is completely independent of the data that is presented. ONLY the data values from a referred trace are used. For example, notice that the Equation trace **Tr2** has Frequency on the X-axis although the referred trace **Tr1** is presented in Time.

**Functions and Constants used in Equation Editor**
ALL trace data that is used in Equation Editor is unformatted, complex data.

When using a mouse with the analyzer, hover over a function in the dialog to learn how it is used.

In the following table,

- Function(scalar x) means that an automatic conversion from a complex number to its scalar magnitude is performed before passing the value to the function.
- Function(complex x) means that the entire complex value is used.
- \( a, b, c, d \) are arguments that are used in the function.

<table>
<thead>
<tr>
<th>Function/Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acos(scalar a)</td>
<td>returns the arc cosine of ( a ) in radians</td>
</tr>
<tr>
<td>asin(scalar a)</td>
<td>returns the arc sine of ( a ) in radians</td>
</tr>
<tr>
<td>atan(scalar a)</td>
<td>returns the arc tangent of ( a ) in radians</td>
</tr>
<tr>
<td>atan2</td>
<td>returns the phase of complex ( a = (\text{re},\text{im}) ) in radians</td>
</tr>
<tr>
<td></td>
<td>has the following two argument sets:</td>
</tr>
<tr>
<td></td>
<td>- atan2(complex a) - returns the phase in radians</td>
</tr>
<tr>
<td></td>
<td>- atan2(scalar a, scalar b)</td>
</tr>
<tr>
<td>conj(complex a)</td>
<td>takes ( a ) and returns the complex conjugate</td>
</tr>
<tr>
<td>cos(complex a)</td>
<td>takes ( a ) in radians and returns the cosine</td>
</tr>
<tr>
<td>cpx(scalar a, scalar b)</td>
<td>returns a complex value ((a+ib)) from two scalar values</td>
</tr>
<tr>
<td>e</td>
<td>returns the constant ( \approx 2.71828... )</td>
</tr>
<tr>
<td>exp(complex a)</td>
<td>returns the exponential of ( a )</td>
</tr>
<tr>
<td>getNumPoints()</td>
<td>returns the number of points for the current sweep</td>
</tr>
<tr>
<td>im(complex a)</td>
<td>returns the imag part of ( a ) as the scalar part of the result (zeroes the imag part)</td>
</tr>
<tr>
<td>kfac(complex a, complex b, complex c, complex d)</td>
<td>k-factor:</td>
</tr>
<tr>
<td></td>
<td>( k = (1 -</td>
</tr>
<tr>
<td></td>
<td>when entered in EE:</td>
</tr>
<tr>
<td></td>
<td>kfac(S11,S21,S12,S22) returns a scalar result - the imaginary part of the complex result is always 0</td>
</tr>
<tr>
<td>ln(complex a)</td>
<td>returns the natural logarithm of ( a )</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>log10(complex a)</td>
<td>returns the base 10 logarithm of <code>a</code></td>
</tr>
<tr>
<td>mag(complex a)</td>
<td>returns <code>sqrt(a.re*a.re+a.im*a.im)</code></td>
</tr>
<tr>
<td>max(complex a, complex b,...)</td>
<td>returns the complex value that has the largest magnitude of a list of values.</td>
</tr>
<tr>
<td>max_hold(complex a)</td>
<td>holds the current maximums of the sweep. Disable the equation to reset. See example</td>
</tr>
<tr>
<td>median(complex a, complex b,...)</td>
<td>returns the median of a list of complex values</td>
</tr>
<tr>
<td></td>
<td>- The median is determined by sorting the values by magnitude, and returning the middle one.</td>
</tr>
<tr>
<td></td>
<td>- If an even number of values is passed, then the smaller of the two middle values is returned.</td>
</tr>
<tr>
<td>min(complex a, complex b,...)</td>
<td>returns the complex value that has the smallest magnitude of a list of values.</td>
</tr>
<tr>
<td>min_hold(complex a)</td>
<td>holds the current minimums of the sweep. Disable the equation to reset. See example</td>
</tr>
<tr>
<td>mrkx(a,b)</td>
<td>returns the x-axis value of marker number b on trace number a.</td>
</tr>
<tr>
<td>mrky(a,b)</td>
<td>returns the y-axis value of marker number b on trace number a.</td>
</tr>
<tr>
<td>mu1(complex a, complex b, complex c, complex d)</td>
<td>when entered in EE: <code>mu1(S11,S21,S12,S22)</code></td>
</tr>
<tr>
<td></td>
<td>`mu1 = (1 -</td>
</tr>
<tr>
<td>mu2( complex a, complex b, complex c, complex d)</td>
<td>when entered in EE: <code>mu1(S11,S21,S12,S22)</code></td>
</tr>
<tr>
<td></td>
<td>`mu2 = (1 -</td>
</tr>
<tr>
<td></td>
<td>for both <code>mu1</code> and <code>mu2</code> (Usually written with the Greek character µ)</td>
</tr>
<tr>
<td></td>
<td>- <code>conj</code> is the complex conjugate. For scalars <code>a</code> and <code>b</code>, <code>conj(a+ib) = (a-ib)</code></td>
</tr>
<tr>
<td></td>
<td>- returns a scalar result - the imaginary part of the complex result is always 0</td>
</tr>
<tr>
<td>phase(complex a)</td>
<td>returns <code>atan2(a)</code> in degrees</td>
</tr>
<tr>
<td>PI</td>
<td>returns the numeric constant <code>pi (3.141592)</code>, which is the ratio of the circumference of a circle to its diameter</td>
</tr>
<tr>
<td>pow(complex a,complex b)</td>
<td>returns <code>a</code> to the power <code>b</code></td>
</tr>
<tr>
<td>re(complex a)</td>
<td>returns the scalar part of <code>a</code> (zeroes the imag part)</td>
</tr>
<tr>
<td>sin(complex a)</td>
<td>takes <code>a</code> in radians and returns the sine</td>
</tr>
</tbody>
</table>
\( \sqrt{\text{complex } a} \) returns the square root of \( a \), with phase angle in the half-open interval \((-\pi/2, \pi/2]\).

\( \tan(\text{complex } a) \) takes \( a \) in radians and returns the tangent.

\( \text{traceDataArray}(\text{complex } a) \) returns the entire set of points from a sweep. Function is intended to be used as an argument in a custom function to allow access for data array processing.

\( \text{xAxisArray}() \) or \( \text{xAxisArray}(\text{integer } a) \) returns the current value of the x-axis for this channel or from a specified channel.

\( \text{xAxisIndex}() \) returns the current index in the sweep.

\( \text{xAxisValue}() \) or \( \text{xAxisValue}(\text{integer } a) \) returns the current value of the x-axis index for this channel or from a specified channel.

### Operators used in Equation Editor

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Addition</td>
</tr>
<tr>
<td>-</td>
<td>Subtraction</td>
</tr>
<tr>
<td>*</td>
<td>Multiplication</td>
</tr>
<tr>
<td>/</td>
<td>Division</td>
</tr>
<tr>
<td>(</td>
<td>Open parenthesis</td>
</tr>
<tr>
<td>)</td>
<td>Close parenthesis</td>
</tr>
<tr>
<td>,</td>
<td>Comma - separator for arguments (as in S11, S22)</td>
</tr>
<tr>
<td>=</td>
<td>Equal (optional)</td>
</tr>
<tr>
<td>E</td>
<td>Exponent (as in 23.45E6)</td>
</tr>
</tbody>
</table>

### Example Equations

The following examples may help you get started with Equation Editor.

**Offset each data point in Tr2 from Tr1 by 2dB**

Use the function: \( \text{pow}(\text{complex } a, \text{complex } b) \) -- returns \( a \) to the power \( b \).

\[
20\log(a) + 2 = 20\log(x) \\
\log(a) + 2/20 = \log(x) \quad // \text{divide all by 20.} \\
x = 10^{\log(a) + 2/20} \quad // \text{swap sides and take 10 to the power of both sides} \\
x = 10^{\log(a)} * 10^{(2/20)} \\
x = a * 10^{(2/20)}
\]
The equation is entered into Tr2 as:

\[ \text{Offset} = \text{Tr}1 \times \text{pow}(10, \frac{2}{20}) \]

To offset by 5 dB

\[ \text{Offset} = \text{Tr}1 \times \text{pow}(10, \frac{5}{20}). \]

**Balanced Match using a 2-port analyzer**

\[ \text{SDD11} = \frac{(\text{S}11 - \text{S}21 - \text{S}12 + \text{S}22)}{2} \]

**Conversion loss**

\[ \frac{\text{B}_1}{\text{pow}(10, \frac{-15}{20})} \]

- \( \text{B}_1 \) is a receiver measurement;
- -15 is the input power in dBm

**Third-order intercept point (IP3 or TOI)**

\[ \text{TR}1 \times \sqrt{\frac{\text{Tr}1}{\text{Tr}3}} \]

- Tr1 = input signal power
- Tr3 = intermodulation power (both traces measured with single receivers)

**Harmonics in dBC**

\[ \frac{\text{B}_1}{\text{Tr}2} \]

- \( \text{B}_1 \) is tuned to a harmonic frequency
- Tr2 = power at fundamental frequency, measured with \( \text{B}_1 \) receiver

**PAE (Power Added Efficiency)**

\[ \frac{\text{Pout} - \text{Pin}}{\text{Pdc}} \]

Type the following equation into a new trace with an unratioed measurement, such as AI1. The data format is REAL:

\[ \text{PAE} = 100 \times \left( \frac{0.001 \times \text{pow} (\text{mag(Tr}1),2)}{\text{Tr}2} - \frac{0.001 \times \text{pow} (\text{mag(Tr}1),2)}{\text{pow} (\text{mag(Tr}2),2)} \right) / (\text{Tr}3 \times \text{Tr}4) \]

Where:
- Tr1 - a trace that measures unratioed B receiver.
- Tr2 - a corrected S21 trace (amplifier gain)
- Tr3 - a trace that measures ADC voltage (AI1) across a sensing resistor.
- Tr4 = an equation trace containing Isupp = (Tr3 / value of sensing resistor).

Data is displayed in Real format with units actually being watts.

1-port Insertion Loss

When it is not possible to connect both ends of a cable to the analyzer, a 1-port insertion loss measurement can be made. However, the measured loss must be divided by 2 because the result includes the loss going down and coming back through the cable. This assumes that the device is terminated with a short or open to reflect all of the power. The 'divide by 2' operation (for dB) is performed as follows using Equation Editor:

- Tr1 - an S11 trace in log mag format.
- Tr2 - an equation trace containing sqrt(Tr1)

Max and Min Hold

These two functions allow you to capture and display either the Maximum or Minimum values for each data point over multiple sweeps.

Maxhold(S21) - displays the maximum value for each data point until reset. Reset by disabling, then enabling the equation. This example refers to an S21 trace that is not displayed.

Saving Equation Editor Data

Equation data can be saved to the analyzer hard drive in the following formats:

- Citifile (.cti) - Equation data is saved and recalled. The file header indicates the "underlying" s-parameter trace type.
- PRN - read by Spreadsheet software. Can NOT be recalled by the analyzer.
- CSV - read by Spreadsheet software. Can NOT be recalled by the analyzer.
- MDIF - compatible with Keysight ADS (Advanced Design System). Can NOT be recalled by the analyzer.
- Print to File (bmp, jpg, png) - saves an image of the screen.
Equation data can NOT be saved in .SnP file format. When attempting to save an Equation trace in .SnP format, the "underlying" S-parameter data is saved; not Equation data.
Import Functions

Several additional functions are provided with the VNA. In addition, you can create custom functions which are compiled into a DLL. Import these functions for use in the Equation Editor.

- How to Import Functions
- Supplied User Functions
  - BestFit.dll
  - EqnErrorTerms.dll
  - Expansion.dll

See Also

Custom Equations In PNA.pdf Detailed directions. (This link requires an internet connection.)

**Note:** When building custom equations, ensure that 64-bit DLLs are built when targeting one of the 64-bit VNA versions.

Create custom functions for Equation Editor Template. (This link requires an internet connection.)

Equation Editor Main topic.

How to Import Functions

From the main Equation Editor dialog, click **Import Functions**

**Import Functions** dialog box help

Imports and removes libraries that are used with Equation Editor. A library is a *.DLL file that contains one or more functions.

Although not all functions are applicable to all channels or data sets, they will still appear in the "Function/Constants" list.

Once imported, each library is automatically loaded when the VNA application starts. If a function is not found or if an error occurs while loading, the VNA will not attempt to reload the library when starting.
Import Library tab

Browse  Click to navigate to the .DLL file on the VNA. The recommended location for the custom equation DLLs is the “C:\Program Files\Keysight\Network Analyzer\UserFunctions” directory on the VNA.

Library Preview  Lists the functions that are contained in the library.

Click OK or Apply to load the library.

Remove Library tab

Left pane  Lists the imported libraries. These also appear in the Equation Editor main dialog and remain until removed from the VNA.

Arrows  Click the relevant arrows to move some (>) or all (>>>) libraries from the VNA.

Right Pane  Lists the libraries to remove.

Click OK or Apply to remove the library.

Supplied User Functions

The VNA firmware includes the following custom equation DLLs to supplement the list of built-in
functions. They are available on the VNA at: 'C:\Program Files\Keysight\Network Analyzer\UserFunctions'.

- BestFit.dll
- EqnErrorTerms.dll
- Expansion.dll

**BestFit.dll**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>d_best_fit_dB()</code></td>
<td>Draws the best fit linear regression line to data specified by PARAM. The best fit line minimizes the RSS (root-sum-square) of the trace data in log-magnitude format. The phase of the regression line is identically zero.</td>
</tr>
<tr>
<td><code>d_channelPower()</code></td>
<td>Computes the channel power for the specified measurement on the given frequencies.</td>
</tr>
</tbody>
</table>

- **d_best_fit_dB**(getNumPoints(),xAxisIndex(),xAxisArray(),traceDataArray(PARAM))
- **d_channelPower**(FA,FB,CHANNELNUM,xAxisIndex(),TRACETYPE,traceDataArray(PARAM))

Computes the channel power for the specified measurement on the given frequencies.

- **FA** and **FB** specify the frequency start/stop values in Hz.
- **CHANNELNUM** is the 1-based channel number to use.
- **PARAM** indicates the measurement to compute channel power for.
- **TRACETYPE** indicates how to display the computed result.
  - If TRACETYPE= 0, the display is a flat line with value equal to the channel power;
  - if TRACETYPE= 1 (default), the display is set to the trace minimum for frequencies outside the range from FA to FB. See examples below.
- If **CHANNELNUM** is hosting an IM Spectrum measurement, the channel power is computed from the formula:
  \[
  \text{channelPower} = 10\log_{10}\left( \frac{\text{area under PARAM trace between FA and FB}}{\text{resolutionBandwidth}} \right)
  \]

For all other measurement types, the formula is:

\[
\text{channelPower} = 10\log_{10}\left( \frac{\text{area under PARAM trace between FA and FB}}{\text{IFBandwidth}} \right)
\]

**TraceType Example 1**: FA=7.75 GHz, FB=8.5 GHz, CHANNELNUM power is displayed over the entire frequency range.
TraceType Example 2: As above, but TRACETYPE is 1 (the default). Displayed result is set to trace minimum outside of specified range.

\[
d_{DFLP}( \text{FreqStart, FreqStop, xAxisIndex()}, \text{getNumPoints()}, \text{xAxisArr})
\]

Computes the deviation from linear phase data of the specified trace, PAI displayed in Phase format, the displayed data will be residual phase response after the group delay slope has been removed. The FreqStart and FreqStop arguments allow the span for this computation. To cover the channel’s entire span, set the frequency number such as 1E100.
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>d_flatness_dB()</code></td>
<td>Computes the magnitude flatness of the trace data in PARAM, by first re-normalizing the results to 0 dB.</td>
</tr>
<tr>
<td><code>d_mean()</code></td>
<td>Computes the mean of the specified trace and creates a resulting trace whose mean is the average of the real part at each stimulus point.</td>
</tr>
<tr>
<td><code>d_min_max_dev()</code></td>
<td>Computes the slope and intercept of the line which minimizes that maximum deviation and the data specified by PARAM.</td>
</tr>
<tr>
<td><code>d_min_max_dev_d2()</code></td>
<td>Computes the parameters of the parabola which minimizes that maximum deviation and the data specified by PARAM.</td>
</tr>
<tr>
<td><code>d_min_max_dev_range()</code></td>
<td>Identical to the function <code>d_min_max_dev()</code>, but only operates on data in the frequency range from FSTART to FSTOP.</td>
</tr>
<tr>
<td><code>d_min_sum_dev()</code></td>
<td>Computes the slope and intercept of the line which minimizes that sum of the absolute deviations and the data specified by PARAM.</td>
</tr>
<tr>
<td><code>d_tilt_dB()</code></td>
<td>The result of the tilt() function when displayed in LogMag format is the total deltaY of the best fit line for the specified trace or parameter.</td>
</tr>
</tbody>
</table>
d_unwrap()  \hspace{1cm} d\_unwrap(getNumPoints(), xAxisIndex(), traceDataArray(PARAM))

The result of the unwrap() function when displayed in Real format is the trace or Parameter in degrees.

---

**EqnErrorTerms.dll**

**d_Corr1P()**  \hspace{1cm} d\_Corr1P(chan, xAxisIndex(), rcvr, src, RAWDATA)

Computes and displays 1 port corrected data for the trace data supplied in the RAWDATA placeholder.

*chan* - the channel of interest.

*xAxisIndex()* - the bucket (data point) number.

*rcvr* - the port number of the receiver used to acquire the data.

*src* - the port being driven. The rcvr and src arguments are needed to select the appropriate error terms used in the correction process.

*RAWDATA* - Select the data to be corrected by substituting in a trace number or parameter name.

---

**d_DIR()**  \hspace{1cm} d\_DIR(chanNum, xAxisIndex(), rcvr, src)

Displays the directivity term from the cal set used by the channel <chanNum>.

Set chanNum to the desired channel.

Set rcvr and src to the port number for the desired directivity term.

---

**d_EnhResp()**  \hspace{1cm} d\_EnhResp(chan, xAxisIndex(), rcvr, src, RAWMATCH, RAWGAIN)

Computes the corrected gain using enhanced response correction techniques. This technique is useful in cases where you want to ignore the output match of a device or the output match cannot be accurately measured. In this the raw input match and gain are supplied to equation (via RAWMATCH and RAWGAIN placeholders). To use this equation select a trace (TR n) or parameter to use in place of the raw match and gain terms.

*chan* - the channel number

*xAxisIndex()* - the bucket number (do not modify)

*rcvr* - The port where the data is acquired.
**Sr** - The port being driven. The src and rcvr ports are required so that the appropriate error terms are used to calculate the result.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
</table>
| d\_LDM()   | Displays the loadmatch term from the calset used by the channel `<chanNum>`.
|            | Set chanNum to the desired channel.                                          |
|            | Set rcvr to the load port, and src to the source port for the desired load match term. |
|            | LDM(ch, xAxisIndex(), 2, 1) gives you the match presented by port 2 while driving port 1. |
| d\_RFT()   | Displays the reflection tracking term from the calset used by the channel `<chanNum>`. |
|            | Set chanNum to the desired channel.                                          |
|            | Set rcvr and src to the port number for the desired reflection tracking term. |
| d\_SRM()   | Displays the sourcematch term from the calset used by the channel `<chanNum>`. |
|            | Set chanNum to the desired channel.                                          |
|            | Set rcvr and src to the port number for the desired source match term.       |
| d\_TRT()   | Displays the transmission tracking term from the calset used by the channel `<chanNum>`. |
|            | Set chanNum to the desired channel.                                          |
|            | Set rcvr to the receive port and src to the source port such that TRT( ch, xAxisIndex(), 2, 1) gives you the transmission tracking term for the port 2 input receiver driven by port 1, or in other words, the raw S21 tracking term. |
| d\_XTLK()  | Displays the isolation term from the calset used by the channel `<chanNum>`. |
|            | Set chanNum to the desired channel.                                          |
Set rcvr to the receive port and src to the source port such that XTLK(ch, xAxiIndex(), 2, 1) gives you the isolation term for the port 2 input receiver while port 1 is on.

### Expansion.dll

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>admittance(x)</td>
<td>Admittance(x) = 1/x</td>
</tr>
<tr>
<td>Calculate the admittance</td>
<td></td>
</tr>
<tr>
<td>max_hold(x)</td>
<td>(KEY, getNumPoints(), xAxisIndex(), PARAM)</td>
</tr>
<tr>
<td>Shows maximum value of each point</td>
<td></td>
</tr>
<tr>
<td>min_hold(x)</td>
<td>(KEY, getNumPoints(), xAxisIndex(), PARAM)</td>
</tr>
<tr>
<td>Shows minimum value of each point</td>
<td></td>
</tr>
<tr>
<td>PAE(B,S21,Al1,Al2,R,SCALE)</td>
<td>[0.001 \times (B - (B/S21)) / ( SCALE^2 \times Al1(Al1 - Al2)/R ) ]</td>
</tr>
<tr>
<td>Power Added Efficiency.</td>
<td></td>
</tr>
<tr>
<td>- B - power out</td>
<td></td>
</tr>
<tr>
<td>- S21 - corrected amplifier gain</td>
<td></td>
</tr>
<tr>
<td>- Al1 - DC power supply</td>
<td></td>
</tr>
<tr>
<td>- Al2 - DC power amp</td>
<td></td>
</tr>
<tr>
<td>- R - resistance</td>
<td></td>
</tr>
<tr>
<td>- SCALE - scale</td>
<td></td>
</tr>
<tr>
<td>reset(x)</td>
<td>x - a number.</td>
</tr>
<tr>
<td>Resets the max_hold() or min_hold() function.</td>
<td></td>
</tr>
<tr>
<td>To reset a given max_hold() or min_hold() function, call reset with the same key.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDD11(S11,S21,S12,S22)</td>
<td>[(S11 - S21 - S12 + S22)/2]</td>
</tr>
<tr>
<td>Differential mode reflection</td>
<td></td>
</tr>
<tr>
<td>SDC11(S11,S21,S12,S22)</td>
<td>[(S11 - S21 + S12 - S22)/2]</td>
</tr>
<tr>
<td>C to D mode conversion reflection</td>
<td></td>
</tr>
<tr>
<td>SCD11(S11,S21,S12,S22)</td>
<td>[(S11 + S21 - S12 - S22)/2]</td>
</tr>
<tr>
<td>D to C mode conversion reflection</td>
<td></td>
</tr>
</tbody>
</table>
SCC11(S11,S21,S12,S22) \quad (S21 + S12 + S11 + S22)/2

Common mode reflection

Use the following two equations to display impedance versus frequency.

Replace 'LOAD' with the value for Z0 (usually 50).

**Note:** You can read out impedance versus time (not using this function) by creating a marker on a Time Domain trace, then changing the marker format to R+jX. Learn how.

<table>
<thead>
<tr>
<th>Function</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>zReflect(S11,LOAD)</td>
<td>LOAD*(1 + S11)/(1-S11)</td>
</tr>
<tr>
<td>zTransfer(S21,LOAD)</td>
<td>2 * LOAD * (1 - S21)/S21</td>
</tr>
</tbody>
</table>
External DC Meter Data Conversion

When creating equations using values from an external DC meter, it is important to understand how these values are stored in the VNA's data buffers and the conversion that occurs when used in an equation. For example, when a voltage is read from an external DC meter, the value is displayed on the VNA as you would expect. That is, if you are reading a voltage level of 2 V from the DC meter in a trace, the VNA will display a level of 2 V. However, the value stored in the VNA data buffers is not a voltage but is a unit-less value. Voltage, Amperes, dBm, and Watts values from an external DC meter are converted so that the format matches that of the data in the VNA internal receivers. In this way, all of the formats within the VNA are the same. This information is important when performing analysis using the Equation Editor because the trace data is the converted value.

See Also

Equation Editor
Configure a DC Device

The following table shows the formats (which are selected from the Type setting on the External DC Meter Properties dialog) and corresponding equations that convert between external DC meter readings and the VNA representation when using the trace data in an equation.

**Note:** Z0 is the characteristic impedance (typically 50 Ohms), dcMeter is the value from the external DC meter, and pnaVal is the value stored in the VNA data buffers. All data types are REAL.

<table>
<thead>
<tr>
<th>Formats</th>
<th>DC Meter to VNA Data Conversion</th>
<th>VNA to DC Meter Data Conversion</th>
</tr>
</thead>
<tbody>
<tr>
<td>V (volts - default)</td>
<td>+/-sqrt((dcMeter*dcMeter/Z0)*1000)</td>
<td>+/-sqrt((pnaVal*pnaVal/1000)*Z0)</td>
</tr>
<tr>
<td>A (amperes)</td>
<td>+/-sqrt((dcMeter<em>dcMeter</em>Z0)*1000)</td>
<td>+/-sqrt((pnaVal*pnaVal/Z0)/1000)</td>
</tr>
<tr>
<td>dBm</td>
<td>pow(10,dcMeter/20)</td>
<td>20*log(pnaVal)</td>
</tr>
<tr>
<td>W (watts)</td>
<td>sqrt(dcMeter*1000)</td>
<td>pnaVal*pnaVal/1000</td>
</tr>
<tr>
<td>K (kelvin)</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>F (degrees)</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>C (degrees)</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**External DC Meter Voltage Example**

The following example shows how trace data is converted when used in an equation. In this example, a
level of 2 V is read from an external DC meter.

1. Configure the external DC meter as described in Configure a DC Device.
2. In the **External DC Meter** dialog, ensure that **Type** is set to **V**.
3. Press **Trace** > **Trace Setup** > **Add Trace** > **New Trace...**.
4. In the **New Trace** dialog, select the **Receivers** tab, check **Activate**, click on the corresponding down arrow in the **Numerator** column, select the external DC meter from the drop down list, then click **OK**.

![New Trace dialog]

**Note:** If the external DC meter is not displayed in the list, ensure that **Active - Show in UI** is checked in the **External Device Configuration** dialog.

5. Trace 1 and Trace 2 should now be displayed on the VNA. Add markers to both traces. The Trace 2 marker should read 2.00 V from the external DC meter.
6. Select Trace 1, then select **Response**, **Format**, **Lin Mag**.
7. Select **Response**, **Math**, then **Equation Editor...**.
8. Enter the following Trace 1 equation to add a value of 2 to the Trace 2 data.
9. Check **Enable Equation**, then click **OK**.
10. Note that instead of a voltage level of 4.00 V, the Trace 1 marker reads 10.94 U (unit-less value).

As shown in the table above, a voltage from an external DC meter is converted using \( \sqrt{\text{dcMeter}^2 \times \frac{\text{dcMeter}}{Z0} \times 1000} \). Therefore, substituting 2 for dcMeter in the equation and using 50 as Z0 results in a value of 8.94. Adding a value of 2 to the Trace 2 data, as defined in the Trace 1 equation, results in the displayed marker value of 10.94.

10. To ensure that the displayed value is 4 instead of 10.94, which is not useful, use the equation from the **VNA to DC Meter Data Conversion** column of the table above as follows:
11. The Trace 1 marker now displays a value of 4.00 U.
Performing Parameter Conversion of Measurement Results

- **Overview**

- **Selecting Conversion Target Parameter**

### Other 'Analyze Data' topics

### Overview

You can use the parameter conversion function to convert the measurement results of the S-parameter ($S_{ab}$) to the following parameters.

- **Equivalent impedance ($Z_r$) and equivalent admittance ($Y_r$) in reflection measurement**

\[
Z_r = Z_{0a} \times \frac{1 + S_{aa}}{1 - S_{aa}}, \quad Y_r = \frac{1}{Z_r}
\]

- **Equivalent impedance ($Z_t$) and equivalent admittance ($Y_t$) in transmission measurement**

\[
Z_t = \frac{2 \times \sqrt{Z_{0a} \times Z_{0b}}}{S_{ab}} - (Z_{0a} + Z_{0b}), \quad Y_t = \frac{1}{Z_t}
\]

- **Inverse S-parameter (1/$S_{ab}$)**

where:

- $Z_{0a}$: Characteristic impedance of port a
- $Z_{0b}$: Characteristic impedance of port b

- **Z/Y Transmission Shunt**
Conjugation converts the measurement value to complex conjugate number.

When the fixture simulator function is ON and the port impedance function is ON, the value set in the port impedance conversion is used. In other cases, the system $Z_0$ (preset value: 50 &) is used.

**Selecting Conversion Target Parameter**

1. Press **Meas > Meas Setup > Conversions**

2. Select function.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>Z-Reflection</td>
<td>Impedance ($Z_t$) in reflection measurement</td>
</tr>
<tr>
<td>Z-Transmit</td>
<td>Impedance ($Z_t$) in transmission measurement</td>
</tr>
<tr>
<td>Z-Trans-Shunt</td>
<td>Impedance ($Z_t$) Transmission Shunt</td>
</tr>
<tr>
<td>Y-Reflection</td>
<td>Admittance ($Y_t$) in reflection measurement</td>
</tr>
<tr>
<td><strong>Y-Transmit</strong></td>
<td>Admittance ($Y_t$) in transmission measurement</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td><strong>Y-Trans-Shunt</strong></td>
<td>Admittance ($\bar{Y}_t$) Transmission Shunt</td>
</tr>
<tr>
<td><strong>1/S</strong></td>
<td>Inverse S-parameter</td>
</tr>
<tr>
<td><strong>Conjugation</strong></td>
<td>Complex conjugate number</td>
</tr>
</tbody>
</table>
Using Limit Lines

Limit lines allow you to compare measurement data to performance constraints that you define.

- Overview
- Create and Edit Limit Lines
- Display and Test with Limit Lines
- Limit Test Setup
- Point Limit Test
- Saving/Recalling Limit Table
- Displaying Judgement Result of Limit Test
- Testing with Sufficient Data Points

Other Analyze Data topics

Overview

Limit lines are visual representations on the VNA screen of the specified limits for a measurement. You can use limit lines to do the following:

- Give the operator visual guides when tuning devices.
- Provide standard criteria for meeting device specification.
- Show the comparison of data versus specifications.

Limit testing compares the measured data with defined limits, and provides optional Pass or Fail information for each measured data point.

You can have up to 100 discrete lines for each measurement trace allowing you to test all aspects of your DUT response.

Limit lines and limit testing are NOT available with Smith Chart or Polar display format. If limit lines are ON and you change to Smith Chart or Polar format, the analyzer will automatically disable the limit lines and limit testing.
By default, limit lines are drawn in the same color as the trace on which they are created. However, all limit lines can be drawn in Red by setting a preference. **Learn more.**

### Create and Edit Limit Lines

You can create limit lines for all measurement traces. The limit lines are the same color as the measurement trace.

Limit lines are made up of discrete lines with four coordinates:

- BEGIN and END stimulus - X-axis values.
- BEGIN and END response - Y-axis values.

#### Limit Table

**How to turn Limit Table ON/OFF**

**Using Hardkey/SoftTab/Softkey**

1. Press **Math > Analysis**.
2. Click **Limit Table > Limit** to turn ON/OFF the Limit Table.

**Programming Commands**

![](image.png)

**Note:** To ADD a limit line to the table, change the last limit line to either MAX or MIN

1. In the **Type** area of the Limit Table, select **MIN** or **MAX** for Limit Line 1.
   - The MIN value will fail measurements BELOW this limit.
   - The MAX value will fail measurements ABOVE this limit.
2. Click **BEGIN STIMULUS** for Limit Segment 1. Enter the desired value.
3. Click **END STIMULUS** for Limit Segment 1. Enter the desired value.
4. Click **BEGIN RESPONSE** for Limit Segment 1. Enter the desired value.
5. Click **END RESPONSE** for Limit Segment 1. Enter the desired value.

6. Repeat Steps 1-5 for each desired limit line.

**Displaying and Testing with Limit Lines**

After creating limit lines, you can then choose to **display** or **hide** them for each trace. The specified limits remain valid even if limit lines are not displayed.

Limit testing cannot be performed on memory traces.

You can choose to provide a visual and / or audible PASS / FAIL indication.

With limit testing turned ON:

- Any portion of the measurement trace that **fails** is **displayed in red**.
- Any portion of the measurement trace that does **NOT fail** remains unchanged and silent.

**Display failed trace points or trace segments**

You can display the data points that fail limit line testing as red dots or as a red trace segment. The default behavior (red trace) can be changed with a Preference setting. Learn how.

![Red dots and Red trace segment](image)

**PASS is the default mode of Pass / Fail testing.**

A data point will **FAIL** only if a measured point falls outside of the limits.

- If the limit line is set to OFF, the entire trace will PASS.
- If there is no measured data point at a limit line stimulus setting, that point will PASS.

**Limit Test Setup**
How to set Limit Test Setup

Using Hardkey/SoftTab/Softkey

1. Press **Math > Analysis**.

2. Click **Limits...** and then select **Limit** tab on the dialog box.

**Limit Test** dialog box help

**Limit Test ON** Check the box to compare the data trace to the limits and display PASS or FAIL.

**Limit Line ON** Check the box to make the limits visible on the screen. (Testing still occurs if the limits are not visible.)

**Sound ON Fail** Check the box to make the VNA beep when a point on the data trace fails the limit test.

**Pass/Fail Position**

Sets the position of the Limit Line Pass/Fail status indicator on the VNA screen.

**X** - X-axis position. 0 is far left; 10 is far right.

**Y** - X-axis position. 0 is bottom; 10 is top.
Show Table  Shows the table that allows you to create and edit limits.

Load Table - Recall the saved limit table. Learn more.

Save Table - Save the limit table. Learn more.

Note: To ADD a limit line to the table, change the last limit line to either MAX or MIN.

Global Pass/Fail

The Pass/Fail indicator provides an easy way to monitor the status of ALL measurements.

Global Pass/Fail ON  Check to display the Global Pass/Fail status.

Policy:  Choose which of the following must occur for the Global Pass/Fail status to display PASS:

- **All Tests** (with **Limit Test ON**) Must Pass - This setting reads the results from the Limit Tests. If all tests (with **Limit Test ON**) PASS, then the Global Pass/Fail status will PASS.

- **All Measurements Must Pass** - This more critical setting shows FAIL unless all measured data points fall within established test limits and Limit Test is ON. Note: In this mode, if one measurement does NOT have **Limit Test ON**, Global Pass/Fail will show FAIL.

Note: In this mode, if one measurement does NOT have **Limit Test ON**, Global Pass/Fail will show FAIL.

Learn more about displaying and testing with Limits (scroll up)

Saving/Recalling Limit Test Table

The limit test table can be saved in a file and recalled later for use on the screen. The file is saved in the csv format (with the extension *.csv), and values are saved as a character string with the unit. The csv formatted file can also be reused in spreadsheet software made for PCs.
How to turn Save or Load Limit Test Table

Using **Hardkey/SoftTab/Softkey**

1. Press **Math > Analysis.**

2. Click **Limits...** and then select **Limit** tab on the dialog box.

3. Click **Load Table** to recall the saved Limit Table.

4. Click **Save Table** to save the Limit Table.

| No Programming are available for this feature |

**Load Table**

1. To recall the saved limit table, click **Load Table** from the Limit Test Setup dialog and a Recall dialog box is open. At this time, CSV Files (with the extension *.csv) is selected as the file type.

2. Specify the folder that contains the file and then select the file. Click Recall to recall the saved limit table on the screen.

**Note:** You can recall a limit table from a trace on any channel independently of the channel and trace that were active when the limit table was saved to the file.

**Save Table**

1. To save the limit table, click **Save Table** from the Limit Test Setup dialog and a Save As dialog box is open. At this time, CSV Files (with the extension *.csv) is selected as the file type.

2. Specify any folder in which you want to save the file and enter the file name. Click **Save** to save the limit table displayed on the screen to a file.

**The limit table is saved in the following format:**

- First line indicates the type of limit test of the instrument.

- Second line indicates the revision of the limit test.

- Third line indicates a header for the segment items that are output from the fourth line onward.
From the fourth line onward, the segment data are output.

**Sample Limit table saved format:**

"# E5080 Limit Test"

"# Revision: 1.00"

<table>
<thead>
<tr>
<th>TYPE, BEGIN STIMULUS, END STIMULUS, BEGIN RESPONSE, END RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIN, 5.600000 GHz, 7.500000 GHz, -30.000000dB, -30.000000dB</td>
</tr>
<tr>
<td>MAX, 4.700000 GHz, 5.800000 GHz, -10.000000dB, -10.000000dB</td>
</tr>
<tr>
<td>MAX, 6.200000 GHz, 8.000000 GHz, -10.000000dB, -10.000000dB</td>
</tr>
<tr>
<td>OFF, 0.000000 Hz, 0.000000 Hz, 0.00dB</td>
</tr>
</tbody>
</table>

**Displaying Judgement Result of Limit Test**

**Judgment result of measurement points and trace**

Measurement points that fail are displayed in red on the screen. The judgment result of the trace is indicated by Pass or Fail displayed at the right bottom of screen by default and its position can be edited.

**Judgment Result of Channels**

If a channel has a judgment result of fail, the result is displayed at Global Pass/Fail dialog box when the
Global Pass/Fail ON is checked (ON). It will be judged as failed if one or more unsatisfactory trace exists in any of the limit test within the channel.

How to turn ON/OFF Global Pass/Fail

Using Hardkey/SoftTab/Softkey

1. Press **Math > Analysis**.
2. Click **Limits...** and then select **Limit** tab on the dialog box.
3. Check the box to turn ON the Global Pass/Fail.
4. Clear the box to turn OFF the Global Pass/Fail.

Testing with Sufficient Data Points

When **System > System Setup > Preferences**, **Limit: Test the nearest measurement point** is NOT checked, limits are checked only at the actual measured data points. Therefore, it is possible for a device to be out of specification without a limit test failure indication if the data point density is insufficient.

The following image is a data trace of an actual filter using 11 data points (approximately one every vertical graticule). The filter is being tested with a minimum limit line (any data point under the limit line fails).

Although the data trace is clearly below the limit line on both sides of the filter skirts, there is a PASS indication because there is no data point being measured at these frequencies.
The following image shows the exact same conditions, except the number of data points is increased to 1601. The filter now fails the minimum limit test indicated by the red data trace.

When System > System Setup > Preferences, Limit: Test the nearest measurement point is checked, the limit is compared with the nearest measurement point.
Limit Test at certain point

The limit test at a certain frequency point is available. This function is similar to one in the E5071C. When (Begin Stimulus = End Stimulus) and (Begin Response = End Response) in the limit test table, the point is defined as point limit test and v (for max) or ^ (for min) symbol is displayed.

When you use the point limit test, confirm if System > System Setup > Preferences, Limit: Test the nearest measurement point is checked. In this setting, even if the test point (= Begin Stimulus = End Stimulus) is not located at measurement point, the result is determined using the nearest measurement point.
Use Ripple Limit Test

- Overview
- Concept of Ripple Limit Test
- Create and Edit Ripple Limit Lines
- Displaying Ripple Limit Test Results
- Ripple Limit Test Setup
- Saving/Recalling Ripple Limit Table

Other 'Analyze Data' topics

Overview

The ripple limit function can be executed independently of limit test function. Independently of the limit test, you can evaluate the measurement results on a PASS/FAIL basis by setting a limit for the ripple. This function is called the Ripple Limit Test.

In this picture, A is greater than B. Therefore, A is considered as the ripple of the specified stimulus range.

The ripple limit function evaluates the measurement point values only. Interpolated values are not used.

Concept of Ripple Limit Test

The ripple limit test is a function for evaluating the results on a Pass/Fail basis based on the ripple
limit, which is set using the ripple limit table. Ripple is defined as the difference between the largest and smallest value within a specified stimulus range. You can specify up to 12 frequency bands, which permits a test for each frequency band.

The ripple limit test judges the measurement as "Pass" when the ripple value specified with the ripple limit is not exceeded by any of the measurement points on the trace; Otherwise, it judges the measurement as "Fail." For the measurement points in a stimulus range without a specified ripple limit, the test judges the measurement as "Pass."

**Note:** The measurement point alone is the target of evaluation for pass/fail. The interpolated part between measurement points is not evaluated.

The ripple limit is defined with the start point stimulus value, end point stimulus value, ripple limit value and type (on/off). For detailed information, see Ripple Limit Table.

While the ripple limit test function is turned on, the measurement points corresponding to a "FAIL" judgment will be indicated in red on the screen and the trace's test results based on the results of each measurement point will be displayed (judged as "Fail" if one or more red measurement point exist on the trace). For information on how to display the results, see Ripple Limit Setup. You can also confirm the channel test results on the screen (judged as "Fail" if one or more failed traces appear in the limit test, ripple limit test or bandwidth limit test within the channel).

### Create and Edit Ripple Limit Lines

You can create ripple limit lines for all measurement traces. The ripple limit lines are the same color as the measurement trace.

Ripple limit lines are made up of discrete lines with three coordinates:

- Begin Stimulus and End Stimulus - X-axis values.
- Max Ripple - Y-axis values.

### Ripple Limit Table

You must configure the ripple limit before you can use the ripple limit test function. You can specify a ripple limit table for each trace, where up to 12 ripple limit bands (frequency bands) can be configured.
How to turn ON/OFF Ripple Limit Table

Using Hardkey/SoftTab/Softkey

1. Press **Math > Analysis.**

2. Click **Limit Table > Ripple** to turn ON/OFF the Ripple Table.

Note: To ADD a frequency band to the ripple limit table, change the last ripple limit line to either ON or OFF.

Note: No frequency band is provided in the ripple limit table by default.

1. In the **Type** area of the Ripple Limit Table, select **ON or OFF** for Ripple Limit Line 1.
   - ON - Band used for the ripple limit test.
   - OFF - Band not used for the ripple limit test.

2. Click **Begin Stimulus** for Ripple Limit Segment 1. Enter the desired value.

3. Click **End Stimulus** for Ripple Limit Segment 1. Enter the desired value.

4. Click **Max Ripple** for Ripple Limit Segment 1. Enter the desired value.

5. Repeat Steps 1-4 for each desired ripple limit line.

Example of ripple limit configuration

- The individual frequency bands for the ripple limit test can overlap each other; in this case, the ripple limit test is performed for each frequency band.

Note: Acceptable range for the stimulus value: -500G to +500G. If any out ranging value is specified, it will be reset to fall within the range.

Note: Even if the VNA's sweep range is changed after the stimulus value has been set, the stimulus value is not susceptible.
- Even if the VNA’s span value is set to zero, you must enter a parameter for both Begin Stimulus and End Stimulus.

- If the data format is Smith chart or polar, the limit test is performed for the main response value among the two marker response values.

**Displaying Ripple Limit Test Results**

**Test result for trace**

The test result for the trace will be indicated as Pass or Fail in the upper-left area of the graph. You can also display the ripple value at the selected frequency band. If a trace is unsatisfactory, test results and ripple lines are displayed by red color.

The result will be displayed as Ripln: Pass (or Fail) for each trace. n denotes the trace number. Bn will be followed by the ripple value (if the ripple display is turned off, only Bn will be displayed without the ripple value).

![Graph showing Ripple Limit Test Result](image)

**Global Pass/Fail**

The Global Pass/Fail setting in the limit tab is applied to ripple limit test. It will be judged as failed if one or more unsatisfactory trace exists in any of the ripple limit test within the channel.

**Ripple Limit Test Setup**
How to set Ripple Limit Test Setup

Using Hardkey/SoftTab/Softkey

1. Press Channel or Trace to select the trace on which you want to apply the ripple limit test function.
2. Press Math > Analysis.
3. Click Limits... and then select Ripple tab on the dialog box.

**Ripple Test**

**Ripple Test ON** - Check the box to set the ripple test ON or OFF and also display PASS or FAIL.

**Ripple Line ON** - Check the box to make the ripple limit line visible on the screen (Test still runs even though the ripple line does not turns on).

**Ripple Result**

**Type** - Sets how the ripple values are displayed. Available settings are **Off**, **Absolute** value (difference between maximum and minimum values within the band) display and **Margin** (difference between absolute value of ripple and ripple limit) display.
**Segment** - Enable to specify a ripple limit table up to 12 stimulus segment for each trace.

**Ripple Table**

**Show Table** - Check the box to show the table that allows you to create and edit limits.

**Load Table** - Recall the saved ripple limit table. Learn more.

**Save Table** - Save the ripple limit table. Learn more.

**Saving/Recalling Ripple Limit Table**

The ripple limit table can be saved in a file and recalled later for use on the screen. The file is saved in the csv format (with the extension *.csv), and values are saved as a character string with the unit. The csv formatted file can also be reused in spreadsheet software made for PCs.

**How to set Ripple Limit Setup**

**Using Hardkey/SoftTab/Softkey**

1. Press **Math > Analysis.**

2. Click **Limits...** and then select **Ripple** tab on the dialog box.

3. Click **Load Table** to recall the saved Ripple Limit Table.

4. Click **Save Table** to save the Ripple Limit Table.

| No Programming are available for this feature |

**Load Table**

1. To recall the saved ripple limit table, click **Load Table** from the Ripple Limit Test Setup dialog and a Recall dialog box is open. At this time, CSV Files (with the extension *.csv) is selected as the file type.

2. Specify the folder that contains the file and then select the file. Click Recall to recall the saved ripple limit table on the screen.

**Note:** You can recall a ripple limit table from a trace on any channel independently of the channel and trace that were active when the ripple limit table was saved to the file.
Save Table

1. To save the ripple limit table, click Save Table from the Ripple Limit Test Setup dialog and a Save As dialog box is open. At this time, CSV Files (with the extension *.csv) is selected as the file type.

2. Specify any folder in which you want to save the file and enter the file name. Click Save to save the ripple limit table displayed on the screen to a file.

The ripple limit table is saved in the following format:

- First line indicates the type of limit test of the instrument.
- Second line indicates the revision of the limit test.
- Third line indicates a header for the segment items that are output from the fourth line onward.
- From the fourth line onward, the segment data are output.

Sample Ripple Limit table saved format:

"# E5080 Ripple Limit Test"

"# Revision: 1.00"

TYPE, BEGIN STIMULUS, END STIMULUS, MAX RIPPLE
ON, 933.0000000 MHz, 964.0000000 MHz, 1.5 dB
ON, 938.0000000 MHz, 953.0000000 MHz, 500 mDB
Use Bandwidth Limit Test

- Overview
- Displaying Bandwidth Limit Test Results
- Bandwidth Limit Test Setup

Other 'Analyze Data' topics

Overview

The bandwidth limit test function can be used for testing bandwidth for the band-pass filters.

The bandwidth test finds the peak of a signal in the passband and locates a point on each side of the passband at an amplitude below the peak specified in test setup. The frequency between these two points is the bandwidth of the filter. Then, the obtained bandwidth is compared to minimum and maximum allowable bandwidth that you specify beforehand.

Displaying Bandwidth Limit Test Results

Test Result for Trace

The test result for the trace will be indicated as Pass, Wide or Narrow in the upper-left area of the graph by following BWn. "n" denotes the trace number. You can also display the bandwidth value. If a trace is unsatisfactory, test results and bandwidth markers are displayed by red color.
Global Pass/Fail

The Global Pass/Fail setting in the limit tab is applied to bandwidth limit test. It will be judged as failed if one or more unsatisfactory trace exists in any of the bandwidth test within the channel.

Bandwidth Limit Test Setup

You must set up the bandwidth threshold and the upper and lower limits before you can use the bandwidth limit test function. You can specify the threshold, upper limit and lower limit for each trace.

How to set Bandwidth Limit Setup

Using *Hardkey/SoftTab/Softkey*

1. Press Channel or Trace to select the trace on which you want to apply the bandwidth limit test function.

2. Press **Math > Analysis.**

3. Click **Limits...** and then select **Bandwidth** tab on the dialog box.
Bandwidth Test Setup Dialog Box Help

Bandwidth Test

**Bandwidth Test ON** - Check the box to set the bandwidth limit test ON or OFF.

**Bandwidth Marker ON** - Check the box to make the bandwidth marker visible on the screen (Test still runs even though the bandwidth marker does not turn on).

**Sound ON Fail** - Check the box to turn ON when the bandwidth limit test is FAIL.

**Target Bandwidth**

**N dB Points** - Specify the bandwidth threshold in dB unit.

**Min Bandwidth** - Enter the lower limit for the bandwidth in Hz unit.

**Max Bandwidth** - Enter the upper limit for the bandwidth in Hz unit.

**Note:** If the data format is Smith chart or polar, the test is skipped.
Save and Recall a File

You can save and recall files to and from an internal or external storage device in a variety of file formats.

- How to Save Instrument State
- How to Save Measurement Data
- How to Recall a File
- About Instrument State and Calibration Data (.csa, .cst, .sta, .cal)
- About Measurement Data Files (.prn, .snp, .cti, .csv, .mdf)
- Define Data Saves
- Managing Files without a Mouse

Other Data Outputting topics

How to Save Instrument State and Calibration Files

Using Hardkey/SoftTab/Softkey

1. Press Save Recall > Save State or Save Other.
Save State

Learn all about VNA Instrument State files.

Save State - Immediately saves the VNA state, possibly calibration data and link to the selected filename by depends on the Save Type. The selected filename is automatically generate in the storage when you performed a save.

Auto Save - Saves state, calibration data and link to the storage. Saves state and calibration data to the internal storage in the D: folder. A filename is generated automatically using the syntax "atxxx"; where xxx is a number that is increment by one when a new file is Auto Saved. The filename is depends on the Save Type to save it in ".sta", ".csa" or ".cst".

Save State As... - Starts the Save As dialog box. (Not available on M948xA and E5080A.)

Save Register - Immediately saves the specified register (Register 1 to 8) to the selected filename by depends on the Save Type. The selected filename is automatically generate in the storage when you performed a save on selected register.

Save Type

State - Save VNA state in .sta filename.

State + Cal Data - Save VNA state and calibration data in .csa filename.

State + Cal Link - Save VNA state and calibration link in .cst filename.

Save Other

Save Calset... & Save Screen... - Starts the Save As dialog box.

Save Data... - Starts the Save Data As dialog box.

Save User Preset... - Start the User Preset dialog box.
**Save As dialog box help**

**Save** Allows you to navigate to the directory where you want to save the file.

**File name** Displays the filename that you either typed in or clicked on in the directory contents box.

**Note:** Filenames (not including the path name) MUST be limited to 64 characters.

**Save as type**

The following file types save **Instrument states and Calibration data**. You can save, and later recall, instrument settings and calibration data for **all channels** currently in use on the analyzer. These file types are only recognized by Keysight VNA analyzers.

*Learn more about these file types.*

- ***.csa** - save Instrument state and actual Cal Set data (cal/state archive) **Default selection.**
- ***.cst** - save Instrument state and a link to the Cal Set data.
- ***.sta** - save Instrument state ONLY (no calibration data)
- ***.cal** - save actual Calibration data ONLY (no Instrument state)

**Note:** To save the screen as .bmp, .jpg, or .png graphics file types, click **File / Print to File.** Learn more.

**Save** Saves the file to the specified file name and directory.

---

**Save VNA Measurement Data**
How to Save Measurement Data

Using **Hardkey/SoftTab/Softkey**

1. Press **Save Recall > Save Other > Save Data...**

**Save Data As** Saves the current trace(s) to the specified type of file.

**Note:** This dialog now contains the settings previously selected from the old **Define Data Save** dialog.

### Save Data As dialog box help

**Note:** Before saving measurement data, always **trigger a single** measurement, and then allow the channel to go into Hold. This ensures that the entire measurement trace is saved.

**Note:** Memory traces can NOT be saved to any file type (PRN, SNP, CTI, CSV, MDF).

**Save in** Allows you to navigate to the directory where you want to save the file.

**File name** Displays the filename that you either typed in or clicked on in the directory contents box.

**Note:** Filenames (not including the path name) MUST be limited to 64 characters.

**Save as type** Choose from: (click each to learn more about each file type): *.prn, *.SNP, *.SNPX, *.cti (citifile), *.csv, *.mdf.

- FCA, GCA, Swept IMD, and Swept IMDx data can be saved to a special csv format. Learn how (FCA, GCA - Swept IMD)

- **Trace and Noise Parameters (*.snp)** - Save the noise figure parameters and S-parameters. Learn more.

- To save the screen as .bmp, .jpg, or .png graphics file types, click **File / Print / Print to File.** Learn more.

- Save Uncertainty (Opt S93015A/B) data (*.u*p, *.dsd, *.sdatcv). Learn more.

### Data Scope


- **Auto**
- When correction is OFF, saves the specified trace.
- When correction is ON, saves all corrected parameters associated with the calibrated ports in the Cal Set.
- For GCA and Swept IMD channels, saves the active trace only.
  - **Single Trace** - Saves the active trace.
  - **Displayed Traces** - Saves all displayed traces for all channels.
  - **Channel Traces** - Saves all displayed traces for active channels.

**Format**

Determines the format of the data. Available with (CTI Formatted, CSV, SNP, MDIF)

  - **Auto** - Data is saved in LogMag or LinMag if one of these is the currently selected display format. If format is other than these, then data is saved in Real/Imag.
  - **LogMag/Angle (dB/deg), LinMag/Angle (unit/deg), Real/Imaginary** - Select output format.
  - The imaginary portion for all LogMag and LinMag data is saved in degrees (dB/deg).
  - **Real/Imaginary data is never smoothed.**
  - **Displayed Format (CSV and MDIF only)** - Data is saved in the format of the displayed trace.

**Note:** .prn files can only save the active trace in the displayed format.

**Save** Saves the file to the specified file name and directory.

**Cancel** - Closes the dialog.

**Help** - Displays **Save Data As** dialog box help.
### How to Recall (open) a file

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Save Recall &gt; Recall.</strong></td>
<td>1. Click on <strong>File</strong></td>
</tr>
<tr>
<td></td>
<td>2. Select <strong>Recall Data...</strong></td>
</tr>
</tbody>
</table>

#### Save Recall > Recall Softtab Help

**Recall State** - Recall the specified filename.

**Recall State** - Select from a list of files shown on softkeys. The list can be sorted by 'most recently used' or alphabetically depending on a preference. The preference setting appears at the bottom of the second page of softkeys listing files to be recalled or on the **Preference dialog**.

**Recall State...** - Starts the **Recall** dialog box.

**Recall Register** - Recall the register (Register 1 to 8) which is saved in the D:\ drive (Only the saved register will enable to recall).

**Recall Calset...** - Starts the **Recall** dialog box.

**Recall Data...** - Starts the **Recall** dialog box.

**Recall Order** - A list of files for recall can arrange according to NAME or RECENT files.

### Recall dialog box help

**Look in** Allows you to select the directory that contains the file that you want to recall.

**Filename** Displays the filename that you either typed in or clicked on in the directory contents box.

**Files of type** Allows you view and select files that are listed in categories of a file type. The following types of files can be recalled into the analyzer: All **State files**, Citi files, SNP files.

#### Recalling instrument state files

When an Instrument State file is recalled, the current state of the instrument is overwritten with the recalled state. A *.cal file does not contain an instrument state, but only calibration data. Learn more about Instrument States.

See also **Power ON and OFF during Save / Recall, User Preset, and Preset.**

#### Recalling Data files

Citi files and SNP files can be recalled and viewed in the analyzer.
1. Click **File** then **Recall**.

2. Select **Citifile Data** or **Snp**.

3. Select the file to recall

4. Click **Recall**.

**Note:** Citi files that were saved in **CW Time sweep** can NOT be recalled into the VNA.

**Note:** Filenames (not including the path name) that are longer than 64 characters will NOT be recalled.

Recalled data is ALWAYS displayed using **LogMag format**, regardless of how the file was stored.

The channel is placed in Trigger Hold. If triggering is resumed, the data will be overwritten.

**SNP files** are recalled as traces into a single window and channel, beginning at the **highest available channel number allowed on the analyzer**. For multi-port SNP files (greater than 4 ports), if the number of S parameters in the file is beyond the **maximum number of traces in a window**, then new windows will be created.

**Citi files** are recalled into the same window and channel configuration as when they were saved. However, the new recalled channel numbers begin with the **highest channel number allowed on the analyzer** and decrement for each additional channel.

For example, when a citi file is saved, two traces are in window 1, channel 1 and two additional traces are in window 2, channel 2. When recalled into a factory preset condition (1 trace in window 1, channel 1), the first two recalled traces appear in window 2, highest channel number, and the second two traces appear in window 3, (highest channel number -1). See also **Traces, Channels, and Windows**.

**Recall** Recalls the file displayed in the file name box.

**Instrument State / Calibration Files**

You can save, and later recall, instrument settings and calibration data **for all channels** currently in use on the analyzer.

An **Instrument State** contains almost every analyzer setting. The following settings are NOT saved and recalled with Instrument State:

- **GPIB address**
The following file types are used to save and recall instrument states and Cal Set information:

<table>
<thead>
<tr>
<th>File Types</th>
<th>Information that is stored for each channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>.sta</td>
<td>Instrument State Information</td>
</tr>
<tr>
<td>.cst</td>
<td>Frequency range</td>
</tr>
<tr>
<td>.csa</td>
<td>Number of points</td>
</tr>
<tr>
<td>.cal</td>
<td>IF bandwidth</td>
</tr>
<tr>
<td></td>
<td>Sweep type</td>
</tr>
<tr>
<td></td>
<td>Sweep mode</td>
</tr>
</tbody>
</table>

**Cal Set Information**

- GUID (Globally Unique Identifier)
  - provides link to Cal Set
- Name, Description, Modify date
- Stimulus Information:
  - Frequency range  | Alternate sweep |
  - Number of points | Port powers |
  - IF bandwidth    | Source attenuators |
  - Sweep type      | Receiver attenuators |
  - Sweep mode      | Test Set port map |
- Error Terms: Directivity, Crosstalk, Source match, Load match, Reflection tracking, Transmission tracking

**File Type Descriptions and Recall**

The following describes each file type, and what occurs when the file type is recalled.

Instrument states can have the following suffixes: .sta, .cst, and .csa.

Common to all of the instrument state files is the state of the instrument including the quantity and content of channels, traces, windows, markers, limit lines, etc.
What is different about the instrument state files is the way they handle the calset - a calset is the set of data that results from having executed a calibration on a channel.

**Compatibility of Files**

- There is no compatibility among the VNA family products. For example, M937xA/P937xA cannot recall the state file which is saved by M980xA/P50xxA.
- The analyzers cannot recall file saved by that with the smaller number of port. For example, 2 port VNAs cannot recall the files saved by 4 port VNAs.
- When the installed option is different, recalling the file may fail.

**.sta files**

- Contain ONLY instrument state information - NOT Cal data.
- When recalled, they always replace the current instrument state immediately.

This instrument state file is saved without any calset data. You might choose to use this type of save file if you are concerned about disk space or specifically do not want to store calibration data that tends to have a shelf life. That is, calibrations become less accurate as cables move and temperature changes. Perhaps you want to force a new calibration when the instrument state is used.

**.cst files**

- Contain BOTH instrument state and a LINK to the Cal Sets. [Learn more about Cal Sets.](#)
- The **quickest and most flexible** method of saving and recalling a calibrated instrument state.
- Channels need not have cal data to save as .cst file.
- When recalled, the state information is loaded first. Then the analyzer attempts to apply a Cal Set as you would do manually. If the stimulus settings are different between the instrument state and the linked Cal Set, the usual choice is presented (see Cal Sets). If the linked Cal Set has been deleted, a message is displayed, but the state information remains in place.
- Because only a link to the Cal Set is saved, the Cal Set can be shared with other measurements.
- If you perform a calibration and save the result to a calset called "MyCalSet", then save a .cst file (for example, MyState.cst), then that file will have a reference to the name of the calset ("MyCalSet").
  - If you redo the calibration and store the data again in MyCalSet, then the next time you recall
MyState.cst, your instrument state will use the new calibration data.

- If you subsequently delete MyCalset, and then recall the MyState.cst, the resulting instrument state will not be calibrated as the calset no longer exists.

**Note:** Before saving a .cst file, be sure that a User Cal Set (NOT a Cal Register) is being used for the calibration. Cal Registers are overwritten with new data whenever a calibration is performed, and may not be accurate cal data when the .cst file is recalled. Learn more about Cal Sets.

* .cal files

- Contain ONLY Cal Set information.
- When recalled, the Cal Set is NOT automatically applied. Apply the calibration data to a channel as you would apply any Cal Set.

* .csa files

- Contain ALL instrument state and the actual Cal Set; not a link to the Cal Set.
- The **safest** method of saving and recalling a calibrated instrument state. However, the file size is larger than a .cst file, and the save and recall times are longer.
- Channels need not be calibrated to save as .cst file.
- The Cal Set that is saved could be a **Cal Register or a User Cal Set**.

A .csa file is an instrument state and a collection of calset data. Every channel that has an active calibration (that is, has a calset selected and applied to the channel) stores its calibration data into the .csa file. For example, let's say you have a calset called “mycalset” and you are using that calset on channel 1. You save a .csa file called MyState. If you then perform another calibration and save that calibration to “mycalset”, the data in “mycalset” will not be the same as the calibration data in the MyState file. Consequently, when you recall MyState the old calset called “mycalset” will overwrite the new calset “mycalset”. When this is about to occur, you are given a warning. If this is a problem for you, you might prefer to use a .cst file.

**Note:** *.pcs files are the internal file format used for storing cal sets. These files should never be accessed or copied by the user.

**Measurement Data Files**

932
Measurement data is saved as ASCII file types for use in a spreadsheet or CAE programs.

**Note:** Before saving measurement data, always **trigger a single** measurement, and then allow the channel to go into Hold. This ensures that the entire measurement trace is saved.

**Note:** **Memory traces** can NOT be saved to any file type (PRN, SNP, CTI, CSV, MDF).

The following file types can be saved by the analyzer:

- *.prn files
- *.SNP (Touchstone)
- *.cti (Citifile)
- *.csv
- *.mdf (MDIF)

***.prn Files**

Prn files have the following attributes:

- Comma-separated data which can be read into rows and columns by spreadsheet software, such as Microsoft Excel. To avoid the "delimiting" dialog boxes, change the filename extension from .prn to .csv. Then open directly into Microsoft Excel.
- Contain formatted and corrected stimulus and response data for the current active trace ONLY.
- Are Output only - they cannot be read by the analyzer.
- Applications and **Cal Set Viewer** data can be saved to *.prn files

**How to Save PRN Trace Data (*.prn)**

**Using Hardkey/SoftTab/Softkey**

1. Press **Save Recall > Save Other > Save Data....**
2. Under **Save as type**, select **PRN Trace Data (*.prn)**.
Example:

"S11 Log Mag"

"Frequency (Hz)", "dB"

3.000000e+005  , -3.528682e+001 ,
4.529850e+007  , -2.817913e+001 ,
9.029700e+007  , -3.216808e+001 ,
1.352955e+008  , -3.101017e+001 ,

**SNP Format (\*.s1p, \*.s2p, \*.s3p, \*.s4p, and so forth)**

- \*.SNP file format, also known as Touchstone format, is specified by IBIS. See the Touchstone specification.
- \*.SNP file format is used by CAE programs such as Keysight's Microwave Design System (MDS) and Advanced Design System (ADS).
- \*.SNP data is saved using the **File, Save Data As** dialog.

Before saving measurement data, always **trigger a single** measurement, and then allow the channel to go into Hold. This ensures that the entire measurement trace is saved.

**\*.SNP files and other analyzer settings**

- \*.SNP data can be **recalled** and viewed on the analyzer, or read by the **embed/de-embed** functions.
- To save SNP data with an **external test set** enabled, at the File, **Save As** dialog, select **SNP File(\*.s*p)**, then complete the "Choose Ports " dialog.
- When **Fixturing** is enabled, all of the enabled data transforms (De-embedding, Port Z Conversion, and so forth) are applied to saved SNP files.
- When **Smoothing** is applied to a trace, the smoothing is NOT saved when the format is Real, Imaginary (RI). Select a different format to save the smoothed data.
- Segmented FCA data is saved to **\*.S2PX files**. Scroll down or **click here** to learn more.
- Learn about **FCA parameters that are saved to an S2P file**.
- Noise Figure parameters and S-parameters can be saved to an \*.s2p file. Learn more.
- Balanced parameters can be saved to \*.SNP files. See the "Choose Ports " dialog.
- **IMPORTANT** - ALL valid data is saved using the same format and settings (trace math, offset, delay, and so forth) as the active measurement. This can cause the data that is saved for the non-active measurements to
be dramatically different from the data that is displayed. For example, when saving an S2P file, if the active S11 measurement is set to Data/Mem (data divided by memory), then ALL 4 S-parameters are saved using Data/Mem. The memory trace that is used in the Data/Mem operation is the same as that used in the active (S11) measurement.

What is Saved

*.SNP data is generally used to gather all S-parameters for a fully corrected measurement.

The analyzer saves the data that is available on the channel of the active measurement.

<table>
<thead>
<tr>
<th>File Type</th>
<th># of Ports</th>
<th># of S-parameters saved</th>
</tr>
</thead>
<tbody>
<tr>
<td>*.s1p</td>
<td>1</td>
<td>1 S-parameter</td>
</tr>
<tr>
<td>*.s2p</td>
<td>2</td>
<td>4 S-parameters</td>
</tr>
<tr>
<td>*.s3p</td>
<td>3</td>
<td>9 S-parameters</td>
</tr>
<tr>
<td>*.s4p</td>
<td>4</td>
<td>16 S-parameters</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>*.SNP</td>
<td>N</td>
<td>N^2 S-parameters</td>
</tr>
</tbody>
</table>

- If correction for a **Full N-port cal** is applied, then valid data is returned for all corrected s-parameters. Response cals will save uncorrected data.

- If requesting **less** data than is available, the Choose ports for SNP data dialog appears.

- If correction is NOT applied, the analyzer returns as much applicable raw data as possible using S-parameter measurements on the selected channel. Data that is not available is zero-filled. For example, if correction is NOT applied and the active measurement is S11, and an S21 measurement also exists on the channel, then data is returned for the S11 and S21 measurements. Data for S12 and S22 is not available and therefore returned as zeros in Real/Imaginary format. In Log Mag/Phase format, this appears as -200 dB and 45 degrees.

**How to Save .SNP Format (*.s1p, *.s2p, *.s3p, *.s4p)**

Using **Hardkey/SoftTab/Softkey**

1. Press **Save Recall > Save Other > Save Data...**
2. Under **Save as type**, select **Trace (*.s1p, *.s2p, *.s3p or *.s4p).**

**.SNP Data Output**

*.SNP files contain header information, stimulus data, a response data pair for EACH S-parameter
measurement. The only difference between .s1p, s2p, and so forth, is the number of S-parameters that are saved.

The following is a sample of **Header information:**

!Keysight Technologies,E8362B,US42340026,Q.03.54
!Keysight E8362B: Q.03.54
!Date: Friday, April 25, 2003 13:46:41
!Correction: S11(Full 2 Port SOLT,1,2) S21(Full 2 Port SOLT,1,2) S12(Full 2 Port SOLT,1,2) S22(Full 2 Port SOLT,1,2)
!S2P File: Measurements:S11,S21,S12,S22:
# Hz  S RI R 50

**Note:** Although the following shows Real / Imag pairs, the format could also be LogMag / Phase or LinMag / Phase

### *.s1p Files

Each record contains 1 stimulus value and 1 S-parameter (total of 3 values)

Stim  Real (Sxx)  Imag(Sxx)

Example:

!Keysight Technologies,E5080A,MY5100056,A.10.99.02
!Date: Tuesday, November 25, 2014 13:15:03
!Correction: S11(Off)
!S1P File: Measurement: S11:
# Hz S dB R 50
100000 -0.10494874 -0.30662519
4509500 -0.03906466 -0.64403939
9009900 -0.03812474 -1.0683264
13509800 -0.0094802867 -1.579396
180098000 0.014229189 -2.3191988
225097500 -0.02068497 -2.8618499
270097000 -0.014636636 -3.4809942

### *.s2p Files

Each record contains 1 stimulus value and 4 S-parameters (total of 9 values)

Stim  Real(S11)  Imag(S11)  Real(S21)  Imag(S21)  Real(S12)  Imag(S12)  Real(S22)  Imag(S22)

Example:
**.s3p Files**

Each record contains 1 stimulus value and 9 S-parameters (total of 19 values)

<table>
<thead>
<tr>
<th>Stim</th>
<th>Real(S11)</th>
<th>Imag(S11)</th>
<th>Real(S12)</th>
<th>Imag(S12)</th>
<th>Real(S13)</th>
<th>Imag(S13)</th>
<th>Real(S21)</th>
<th>Imag(S21)</th>
<th>Real(S22)</th>
<th>Imag(S22)</th>
<th>Real(S23)</th>
<th>Imag(S23)</th>
<th>Real(S31)</th>
<th>Imag(S31)</th>
<th>Real(S32)</th>
<th>Imag(S32)</th>
<th>Real(S33)</th>
<th>Imag(S33)</th>
</tr>
</thead>
</table>

Example:

```plaintext
!Keysight Technologies,E5080A,MY55100056.A.1.19.99.02
!Date: Tuesday, November 25, 2014 13:23:10
!Correction: S11(Off)
S21(Off)
S12(Off)
S22(Off)
S2P File: Measurements: S11, S21, S12, S22:
# Hz S db R 50
100000 -200 45 -93.193119 44.821617 -200 45 -200 45
45099500 -200 45 -85.316757 83.057785 -200 45 -200 45
90099000 -200 45 -86.266129 117.26331 -200 45 -200 45
135098500 -200 45 -97.657417 -75.884865 -200 45 -200 45
180098000 -200 45 -83.678968 -38.655216 -200 45 -200 45
225097500 -200 45 -100.30298 110.7329 -200 45 -200 45
270097000 -200 45 -94.416489 -95.377288 -200 45 -200 45
```

**.s4p Files (and so forth...)**

Each record contains 1 stimulus value and 16 S-parameters (total of 33 values)

<table>
<thead>
<tr>
<th>Stim</th>
<th>Real(S11)</th>
<th>Imag(S11)</th>
<th>Real(S12)</th>
<th>Imag(S12)</th>
<th>Real(S13)</th>
<th>Imag(S13)</th>
<th>Real(S14)</th>
<th>Imag(S14)</th>
<th>Real(S21)</th>
<th>Imag(S21)</th>
<th>Real(S22)</th>
<th>Imag(S22)</th>
<th>Real(S23)</th>
<th>Imag(S23)</th>
<th>Real(S24)</th>
<th>Imag(S24)</th>
<th>Real(S31)</th>
<th>Imag(S31)</th>
<th>Real(S32)</th>
<th>Imag(S32)</th>
<th>Real(S33)</th>
<th>Imag(S33)</th>
<th>Real(S34)</th>
<th>Imag(S34)</th>
<th>Real(S41)</th>
<th>Imag(S41)</th>
<th>Real(S42)</th>
<th>Imag(S42)</th>
<th>Real(S43)</th>
<th>Imag(S43)</th>
<th>Real(S44)</th>
<th>Imag(S44)</th>
</tr>
</thead>
</table>

Example:
**.S2PX Data Output**

*.S2PX files are used for Segmented Mixer Data. [Learn more.](#)

The following ADDITIONAL columns precede parameter data:

SegIndex, InputFreq, OutputFreq, LO1Freq, InputPower, LO1Power, <parameter data>

---

**Choose ports for SNP File** dialog box help

This dialog appears when any of the following conditions exist while attempting to save data to an *snp file:
- you request less data than is available

- you want data for more than 4 ports

- a balanced measurement is active

**Number of ports**  Select the number of ports for which data will be saved.

The following buttons appear ONLY when a **Balanced measurement** is displayed.

**Normal**  Click to save normal (single-ended) port data.

**Mixed Mode**  Click to save balanced (logical) port data. Choices are based on the **topology selection** for current active parameter:

- **SE-Bal**: Choose from S1, D2, C2 (Single-ended port 1, Differential port 2, Common port 2)

- **SE, SE, Bal**: Choose from S1, S2, D3, C3 (Single-ended port 1, Single-ended port 2, Differential port 3, Common port 3)

- **Bal-Bal**: Choose from D1, C1, D2, C2 (Differential port 1, Common port 1, Differential port 2, Common port 2)

For example, with SE-Bal topology, choose 2 ports, S1 for first, and D2 for second. The following 4 parameters are saved: Sss11, Ssd12, Sds21, Sdd22.

**Arrow buttons**  Click to Add or Remove ports from or to the following columns:

**Available Ports**  All test set ports are listed. There may NOT be valid data available for all of these ports. Learn more.

**Chosen Ports**  When **OK** is clicked, SNP data is saved for these ports.

**OK**  Becomes available when the number of **Chosen ports** = the **Number of ports** to save. Click to save to SNP file.

With **Number of ports** = 2, .s2p data is saved; with **Number of ports** = 3, .s3p data is saved, and so forth. Learn more about SNP files

.cti Citifiles

Citifile format is compatible with the Keysight 8510 Network Analyzer and Keysight's Microwave Design System (MDS).

You can do the following using citifiles:
• save the active trace, or all traces.

• save formatted or unformatted citifile data

---

How to Save Citifile Formatted Data (*.cti)

Using Hardkey/SoftTab/Softkey

1. Press **Save Recall** > **Save Other** > **Save Data**.

2. Under **Save as type**, select **Citifile Formatted Data (*.cti)**.

---

*.cti files contain:

• Header information

• Stimulus data

• Data pairs for EACH S-parameter measurement

---

The above image is a Citifile opened in Notepad. There are two traces in separate channels - one is an FCA trace. Each trace has 3 data points. The save settings = **Displayed Traces** Content, and **Auto** Format.

Format is identified by DBANGLE (log mag), MAGANGLE (Lin Mag), or RI (real, imaginary - NOT shown)

On the **data access map**, Formatted data is taken from location 2 or 4.
How to Save Citifile Unformatted Data (*.cti)

Using Hardkey/SoftTab/Softkey

1. Press Save Recall > Save Other > Save Data....

2. Under Save as type, select Citifile Data Data (Real,imag) (*.cti).

On the data access map, Unformatted data is taken from the block just before Format.

Citifiles can be recalled and viewed in the analyzer. Learn more.

*.csv Files

Note: 2D Gain Compression data is saved as *.csv files using a different format than shown here. Learn more.

CSV files are read by spreadsheet programs such as Microsoft Excel.

How to Save CSV Formatted Data (*.csv)

Using Hardkey/SoftTab/Softkey

1. Press Save Recall > Save Other > Save Data....

2. Under Save as type, select CSV Formatted Data (.csv).

*.csv files contain: header information and the following Comma-Separated Values.

- Stimulus data
- Data pairs for EACH S-parameter
*.mdf Files

MDIF files are compatible with Keysight ADS (Advanced Design System). Learn more at the Keysight website.

How to Save MDIF Data (*.mdif)

Using **Hardkey/SoftTab/Softkey**

1. Press **Save Recall > Save Other > Save Data...**
2. Under **Save as type**, select **MDIF Data (*.mdif)**.

*.mdf files contain: header information and space-separated data:

- **Stimulus data**
- **Real and Imaginary data pair for EACH S-parameter measurement**

Define Data Saves

**Note:** Although these settings are still supported, they are no longer necessary to save data files. The **Save Data As** dialog box contains these settings.
How to select Define Data Saves

Using Hardkey/SoftTab/Softkey

1. Press System > System Setup > Preferences....

2. Click the Data Saves... button on the Preferences dialog box.

Define Data Saves dialog box help

Note: Although these settings are still supported, they are no longer necessary to save data files. The Save Data As dialog box contains these settings.

The following settings survive an Instrument Preset and Shutdown.

CitiFile, CSV, and MDIF Contents

Determines what is saved to a .cti file.

Auto - Saves the active trace. Additional traces are saved if correction is ON. For Full 2-port calibration, 4 traces are saved; for Full 3-port calibration, 9 traces are saved, and so forth.

Single Trace - Saves the active trace.

Displayed Traces - Saves all displayed traces for all channels.

CitiFile and CSV Format

Auto - Data is saved in LogMag or LinMag if one of these is the currently selected display format. If format is other than these, then data is saved in Real/Imag.

LogMag, LinMag, Real/Imag - Select output format.

- The imaginary portion for all LogMag and LinMag data is saved in degrees.
- Real/Imag data is never smoothed.

SnP Format (.s1p, .s2p, .s3p)

Learn more about SnP files.

Auto - Data is saved in LogMag or LinMag if one of these is the currently selected format. If format is other than these, then data is saved in Real/Imag.

LogMag, LinMag, Real/Imag - Select output format. The imaginary portion for all LogMag and
LinMag data is output is in degrees.

Manage Files without a Mouse

How to Manage Files without a Mouse

Using Hardkey/SoftTab/Softkey

1. Press Save Recall > Save Other.
2. Click Manage Files... and then D: \ drive folder dialog box appears.

OR

1. Press System > Main.
2. Click Manage Files... and then D: \ drive folder dialog box appears.

Manage Files dialog box help

The Manage Files dialog box is designed to be used from the front panel. It performs the same function as Windows Explorer, but can be used without the use of a mouse or keyboard.
Drive Mapping

Drive mapping allows you to share disk drives between the VNA and an external computer.

To prepare for Drive Mapping:

1. Both the PC and VNA must be connected to a shared computer network.
2. You must know the full computer name of the Analyzer you are mapping TO. Tell me how.

Note: This procedure requires a mouse and keyboard.

Map to a drive on the VNA from an External PC

1. On the analyzer desktop, click Windows Explorer. Right-click on the drive you want to share. Click Sharing...
2. In the dialog box, select Shared this folder. In the Share Name box, type in a share name for the drive. For example: C$. Click OK.
3. On the external PC desktop, click Windows Explorer. From the Tools menu, click Map Network Drive.
4. If the current logon on your PC is different from the current logon on the analyzer, click Connect using a different Logon to connect to using the current analyzer logon. This logon must be registered on the external PC. To see the current logon on either the PC or analyzer, hold Ctrl - Alt, and press Delete.
   a. In the Connect as box, type the logon currently being used by the analyzer.
   b. In the Password box, type the logon password that you use on the external computer. Click OK.
5. In the Folder box, type //computername (prep1)/share name (from step 2). (For example: //SLT1234/C$)
6. Click Finish.
Print a Displayed Measurement

The analyzer allows you to print a displayed measurement to a printer or to a file. The printer can be either networked or local.

- Connecting a Printer
- Printing

Other Outputting Data topics

Connecting a Printer

You can connect a printer to one of the VNA USB ports or to the LAN connector.

To Add a Printer

**Note:** If you try to print from the VNA application and the Add Printer Wizard appears, click Cancel and add the printer using the following procedure.

1. From the VNA application, press System > Main > Minimize Application.
2. On the Windows taskbar, click Devices and Printers.
3. Double-click Add Printer.
4. Follow the instructions in the Add Printer Wizard.

For more information, refer to Microsoft Windows Help or your printer documentation.

Printing

- Print a Hardcopy
- Page Setup
- Print to File

The measurement information on the screen can be printed to any local or networked printer that is connected to the VNA. The graphic below shows an example of how a screen-capture image appears...
when printed. The Page Setup settings allows you to customize the printed form of the measurement information.

How to Print a Hardcopy

Using Hardkey/SoftTab/Softkey

1. Press System > Print.
2. Click Print....

No programming commands are available for this feature.
Note: For information on the choices in the Print dialog box, see Windows Help.

Page Setup

The Page Setup dialog allows flexibility in the appearance that measurement data is printed. After setting up the page, click File, then Print... to obtain a hard-copy.

How to select Page Setup

Using Hardkey/SoftTab/Softkey

1. Press System > Print > Page Setup....

OR

1. Press System > System Setup > Preferences.....

2. Click the Page Setup... button on Preferences dialog box.

Page Setup dialog box help
Paper, Orientation, and Margins

These settings do NOT survive a VNA shutdown.

See Windows Help for information on these settings.

Windows

The following VNA-specific settings DO survive a VNA shutdown:

Minimum vertical size  Adjust to change the amount of a page that the measurement window fills. The adjustment range is from 40 to 100%.

One window per page  Check to print one window per page. Clear to print all selected windows without a forced page break.

Only active window  Check to print only the active window. Clear to print all windows.

Instrument logo  Check to print the Keysight logo to the header.

Data and Time  Check to add the current date and time to the header.

Global Pass/Fail  Check to add the Global Pass/Fail status to the header.

Page Numbers  Check to add page numbers (1 of n) to the header.

Channel Settings Table

Print  Check to print the channel settings table.
Segment data can no longer be printed.

**Trace Attributes Table**

**Print** Check to print the Trace Attributes Table. The Trace Attributes are measurement type, correction factors ON or OFF, smoothing, options, and marker details. The Trace Attributes are listed by Trace ID# for each window.

Each Trace ID# can have multiple entries depending on the number of markers associated with the trace. The marker details are marker number, position and response. If there are multiple markers on a trace, the trace attributes are only shown for the first marker. However, the trace attributes for the first marker apply to all other markers on that trace.

The options column can have one or more options. D for Delay, M for Marker, G for Gating. Multiple options selected would appear as follows: DMG.

**Print marker data** Check to print all marker data. The amount of data depends on how many markers are created.

---

**Print to a File**

The analyzer can save a screen-capture image in any of the following formats:

- .png (preferred format)
- .bmp (bitmap)
- .jpg

The analyzer automatically saves the file to the current path. If not previously defined, the analyzer automatically selects the default path D:\.

A .bmp file, like a .prn file, can be imported into software applications such as Microsoft Excel, Word, or Paint to display a screen-capture image.

See Save and Recall files for more information.

---

**How to Print to a File**

**Using Hardkey/SoftTab/Softkey**

1. Press **System > Print > Print to File**....
Finding Programming Commands

Three ways to find programming commands:

1. **From simulated User Interface:** **Hardkeys**, **SoftTabs**, and **Softkeys**.

   Click on **Command Finder** to display the screen below. Clicking on one of the **Hardkeys** shown in the online help screen opens the corresponding **SoftTab** and **Softkey** menus and corresponding SCPI/COM commands. This is a "mapping" of GUI functions to their corresponding SCPI/COM commands.
2. From a simulated User Interface of the drop-down menus:

File  |  Instrument  |  Response  |  Stimulus  |  Utility  |  Cal  |  Apps  |  Remote ONLY

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3.</td>
<td>Command Tree</td>
<td>COM Object Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>See Also</td>
<td>Example Programs</td>
<td>Example Programs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Learning about GPIB</td>
<td>Learning about COM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

See Also

- New Programming Commands
• Remotely Specifying a Source Port
• Shut Down or Restart the PNA Remotely
• Your Programs on Windows
• LXI and VXI-11.3 Compliance
• VEE Examples with runtime installed.
• Using Macros
• Code Translator App.
• Superseded / Replacement Commands
• Data Access Map
• See more programming information and examples at:http://na.support.keysight.com/pna/programming/
ActiveParametersApp Object

Description

Controls the Active (Hot) Parameter Application settings.

Accessing the ActiveParametersApp object

```vba
Dim app
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset
app.CreateCustomMeasurementEx 2, "Active Parameters", "IPwr"
Set ActParam = app.ActiveChannel. CustomChannelConfiguration
```

See Also:

- About Active Match Application
- PNA Automation Interfaces
- The PNA Object Model
- Programming Example

### Method

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>See History</td>
</tr>
</tbody>
</table>

### Property

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AbsoluteExtractionToneLevel</td>
<td>IActiveChannelSettings</td>
<td>Set and read the absolute tone power level.</td>
</tr>
<tr>
<td>DisplayDomain</td>
<td>IActiveChannelSettings</td>
<td>Set and read the X-axis domain type.</td>
</tr>
<tr>
<td>DisplayInputPower</td>
<td>IActiveChannelSettings</td>
<td>Set and read a fixed input power.</td>
</tr>
<tr>
<td>DisplayInterpolationState</td>
<td>IActiveChannelSettings</td>
<td>Set and read whether or not interpolation is on for display.</td>
</tr>
<tr>
<td>ExtractionToneMode</td>
<td>IActiveChannelSettings</td>
<td>Set and read the tuning tone mode.</td>
</tr>
<tr>
<td>PhaseSweepPoints</td>
<td>IActiveChannelSettings</td>
<td>Set and read the number of phase points.</td>
</tr>
</tbody>
</table>
PowerStepsIn3DSweep  IActiveChannelSettings  Set and read the number of power steps for a 3D sweep.
RelativeExtractionToneLevel IActiveChannelSettings  Set and read the relative tone power level.
StartPowerIn3DSweep IActiveChannelSettings  Set and read the start power level for a 3D sweep.
StopPowerIn3DSweep IActiveChannelSettings  Set and read the stop power level for a 3D sweep.
SweepType IActiveChannelSettings  Set and read the sweep type.

IActiveChannelSettings History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IActiveChannelSettings</td>
<td>12.90</td>
</tr>
</tbody>
</table>

957
<table>
<thead>
<tr>
<th><strong>Channels</strong></th>
<th><strong>Measurements</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel</td>
<td>Measurement</td>
</tr>
<tr>
<td>AuxiliaryTrigger</td>
<td>IArrayTransfer</td>
</tr>
<tr>
<td>BalancedTopology</td>
<td>IMixer</td>
</tr>
<tr>
<td>Calibrator</td>
<td>EmbeddedLO</td>
</tr>
<tr>
<td>ICalData</td>
<td>ELODiag</td>
</tr>
<tr>
<td>Converter</td>
<td>BalancedMeasurement</td>
</tr>
<tr>
<td>ConvEmbedLO</td>
<td>BalancedStimulus</td>
</tr>
<tr>
<td>CorrectionMethods</td>
<td>Equation</td>
</tr>
<tr>
<td>DCStimulus</td>
<td>GainCompressionMeas</td>
</tr>
<tr>
<td>Fixturing</td>
<td>Gating</td>
</tr>
<tr>
<td>FOM</td>
<td>GroupDelayAperture</td>
</tr>
<tr>
<td>FOMRange</td>
<td>Marker</td>
</tr>
<tr>
<td>IFConfiguration</td>
<td>LimitTest</td>
</tr>
<tr>
<td>PathConfiguration</td>
<td>LimitSegment</td>
</tr>
<tr>
<td>PathElement</td>
<td>PNOP</td>
</tr>
<tr>
<td>PhaseControl</td>
<td>PSaturation</td>
</tr>
<tr>
<td>PulseGenerator</td>
<td>Transform</td>
</tr>
<tr>
<td>PulseMeasControl</td>
<td>Capabilities</td>
</tr>
<tr>
<td>RxLeveling</td>
<td>MeasurementClassProperties</td>
</tr>
<tr>
<td>Segments</td>
<td>PowerRange</td>
</tr>
<tr>
<td>Segment</td>
<td>E5091ATestSets</td>
</tr>
<tr>
<td>SignalProcessing</td>
<td>E5091ATestSet</td>
</tr>
<tr>
<td>DIQ</td>
<td>ENRFile</td>
</tr>
<tr>
<td>GainCompression</td>
<td>ExternalDevices</td>
</tr>
<tr>
<td>IMSpectrum</td>
<td>ExternalDevice</td>
</tr>
<tr>
<td>NoiseFigure</td>
<td></td>
</tr>
<tr>
<td>SweptIMD</td>
<td></td>
</tr>
<tr>
<td>SpectrumAnalyzer</td>
<td>ExternalDCDevice</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>SpectrumAnalyzerDFT</td>
<td>ExternalPulseGenerator</td>
</tr>
<tr>
<td>SpectrumAnalyzerCoherence</td>
<td>ExternalSMUDevice</td>
</tr>
<tr>
<td>NaWindows</td>
<td>PowerSensorAsReceiver</td>
</tr>
<tr>
<td>NaWindow</td>
<td>PowerLossSegmentsPMAR</td>
</tr>
<tr>
<td>Traces</td>
<td>PowerLossSegmentPMAR</td>
</tr>
<tr>
<td>Trace</td>
<td>CalFactorSegsPMAR</td>
</tr>
<tr>
<td>CalKit</td>
<td>PowerSensorCal</td>
</tr>
<tr>
<td>CalStandard</td>
<td>FactorSegmentPMAR</td>
</tr>
<tr>
<td>CalManager</td>
<td>ExternalTestSets</td>
</tr>
<tr>
<td>CalibrateAllChannels</td>
<td>TestSetControl</td>
</tr>
<tr>
<td>CalSets</td>
<td>FIFO</td>
</tr>
<tr>
<td>CalSet</td>
<td>GlobalPowerLimit</td>
</tr>
<tr>
<td>ICalData</td>
<td>HW AUXIO</td>
</tr>
<tr>
<td>GuidedCalibration</td>
<td>HWE ExternalTestSet</td>
</tr>
<tr>
<td>GuidedCalibration</td>
<td>HW Material Handler IO</td>
</tr>
<tr>
<td>PowerSensors</td>
<td>InterfaceControl</td>
</tr>
<tr>
<td>GuidedCalibration</td>
<td>IO Configuration</td>
</tr>
<tr>
<td>PowerSensor</td>
<td>PathConfigurationMgr</td>
</tr>
<tr>
<td>GuidedCalibration</td>
<td>Port Extension</td>
</tr>
<tr>
<td>PowerSensor</td>
<td>Preferences</td>
</tr>
<tr>
<td>ECalUserCharacterizer</td>
<td>Display/PrintColors</td>
</tr>
<tr>
<td>ECalModules</td>
<td>Trace (1-8)</td>
</tr>
<tr>
<td>ECalModule</td>
<td>SCPI String Parser</td>
</tr>
<tr>
<td>PhaseReferenceCal</td>
<td>Trigger Setup</td>
</tr>
<tr>
<td>SMCTYPE</td>
<td>Uncertainty Manager</td>
</tr>
<tr>
<td>VMCTYPE</td>
<td>Cables</td>
</tr>
<tr>
<td>NoiseCal</td>
<td></td>
</tr>
<tr>
<td>GainCompressionCal</td>
<td></td>
</tr>
<tr>
<td>SweptIMDCal</td>
<td></td>
</tr>
<tr>
<td>SourcePowerCalibrator</td>
<td></td>
</tr>
<tr>
<td>PowerLossSegments</td>
<td>Cable</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>PowerLossSegment</td>
<td>Ports</td>
</tr>
<tr>
<td>PowerMeterInterfaces</td>
<td>Port</td>
</tr>
<tr>
<td>PowerMeterInterface</td>
<td>Characterizer</td>
</tr>
<tr>
<td>PowerSensors</td>
<td>GuidedCalibration</td>
</tr>
<tr>
<td>PowerSensor</td>
<td>VISAPassthrough</td>
</tr>
</tbody>
</table>

Legend:
- Object
- Collection
- Interface

---

960
**Application Object**

**Description**

The Application object is the highest object in the VNA **object model**. This object presents methods and properties that affect the entire analyzer, rather than a specific channel or measurement. For example, the application object provides the GetIDString method. There's only one ID string for the instrument, unrelated to the channel or parameter being measured. Likewise, the TriggerSignal Property is global to the instrument. You can elect to use an internally generated (free run) trigger or a manual trigger. Either way, that type of trigger generation will be used on all measurements, on all channels. Therefore, it is under the Application object.

**Accessing the Application object**

This object is unique in that you must **create** this object rather than just get a handle to it.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
```

Replace `<analyzerName>` with the full computer name of your VNA. For example, "My VNA". See **Change Computer Name**.

**See Also:**

- VNA Automation Interfaces
- The VNA Object Model
- Getting a Handle to an Object.
- Example Programs
- commands

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActivateWindow</td>
<td>IApplication</td>
<td>Makes a window object the Active Window.</td>
</tr>
<tr>
<td>AllowAllEvents</td>
<td>IApplication</td>
<td>Monitors all events</td>
</tr>
<tr>
<td>AllowEventCategory</td>
<td>IApplication</td>
<td>Monitors an event category</td>
</tr>
<tr>
<td>Method</td>
<td>IApplication</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AllowEventMessage</td>
<td></td>
<td>Monitors an event</td>
</tr>
<tr>
<td>AllowEventSeverity</td>
<td></td>
<td>Monitors an event severity level</td>
</tr>
<tr>
<td>BuildHybridKit</td>
<td></td>
<td>Defines the user kit as port1kit + port2kit.</td>
</tr>
<tr>
<td>Channel</td>
<td></td>
<td>Returns a handle to the channel object for the supplied channel number.</td>
</tr>
<tr>
<td>Configure</td>
<td></td>
<td>Restarts as an &quot;N-port&quot; VNA using the specified multiport test set.</td>
</tr>
<tr>
<td>CreateCustomMeasurementEx</td>
<td></td>
<td>Creates a new custom measurement with initialization.</td>
</tr>
<tr>
<td>CreateCustomMeasurement</td>
<td></td>
<td><strong>Superseded with</strong> CreateCustomMeasurementEx Method</td>
</tr>
<tr>
<td>CreateMeasurement</td>
<td></td>
<td>Creates a new measurement.</td>
</tr>
<tr>
<td>CreateSParameter</td>
<td></td>
<td>Creates a new S-Parameter measurement.</td>
</tr>
<tr>
<td>CreateSParameterEx</td>
<td></td>
<td><strong>Superseded with</strong> CreateSParameter Method</td>
</tr>
<tr>
<td>DeleteShortCut</td>
<td></td>
<td>Removes a macro (shortcut) from the list of macros</td>
</tr>
<tr>
<td>DisallowAllEvents</td>
<td></td>
<td>Monitors NO events</td>
</tr>
<tr>
<td>DoPrint</td>
<td></td>
<td>Prints the screen to the active Printer.</td>
</tr>
<tr>
<td>ExecuteShortcut</td>
<td></td>
<td>Executes a macro (shortcut) stored in the analyzer.</td>
</tr>
<tr>
<td>GetAuxIO</td>
<td></td>
<td>Returns a handle to the AuxIO interface</td>
</tr>
<tr>
<td>GetCalManager</td>
<td></td>
<td>Returns a handle to the CalManager interface</td>
</tr>
<tr>
<td>GetExternalTestSetIO</td>
<td></td>
<td>Returns a handle to the ExternalTestSet IO interface</td>
</tr>
<tr>
<td>GetIPConfigurationStruct</td>
<td></td>
<td>Returns an NA_IPConfiguration data structure which contains information about the current status of the VNA computer networking configuration.</td>
</tr>
<tr>
<td>GetLicenses</td>
<td></td>
<td>Returns the list of licenses.</td>
</tr>
<tr>
<td>GetMaterialHandlerIO</td>
<td></td>
<td>Returns a handle to the MaterialHandlerIO interface</td>
</tr>
<tr>
<td>GetShortcut</td>
<td></td>
<td>Returns the title and path of the specified macro (shortcut).</td>
</tr>
<tr>
<td>LANConfigurationInitialize</td>
<td></td>
<td>Resets the VNA LAN configuration.</td>
</tr>
<tr>
<td>LaunchCalWizard</td>
<td></td>
<td>Launches the Cal Wizard</td>
</tr>
<tr>
<td>LaunchDialog</td>
<td></td>
<td>Launches the specified dialog box.</td>
</tr>
</tbody>
</table>
ManualTrigger  IApplication  Triggers the analyzer when TriggerSignal = naTriggerManual.

Preset  IApplication  Resets the analyzer to factory defined default settings.

PrintToFile  IApplication  Saves the screen data to bitmap (.bmp) file of the screen.

PutShortcut  IApplication  Puts a Macro (shortcut) file into the analyzer.

Quit  IApplication  Ends the Network Analyzer application.

Recall  IApplication  Recalls a measurement state, calibration state, or both from the hard drive into the analyzer.

RecallKits  IApplication  Recalls the calibration kits definitions that were stored with the SaveKits command.

Reset  IApplication  Removes all existing windows and measurements.

RestoreCalKitDefaults  IApplication  Restores the factory defaults for the specified kit.

RestoreCalKitDefaultsAll  IApplication  Restores the factory defaults for all kits.

Save  IApplication  Saves instrument state and calibration files to disk.

SaveCitiDataData  IApplication5  Saves UNFORMATTED trace data to .cti file. **Superseded** with SaveData.

SaveCitiFormattedData  IApplication5  Saves FORMATTED trace data to .cti file. **Superseded** with SaveData.

SaveData  IApplication18  Saves trace data to files on disk.

SaveKits  IApplication  Saves all cal kits to disk.

SetFailOnOverRange  IApplication  Causes over range values to return an error code.

SetIPConfiguration  IApplication14  Modifies settings of the VNAs computer networking configuration.

ShowStatusBar  IApplication  Shows and Hides the Status Bar.

ShowStimulus  IApplication  Shows and Hides Stimulus information.

ShowTitleBars  IApplication  Shows and Hides the Title Bars.

ShowToolbar  IApplication  Shows and Hides the specified Toolbar.

UserPreset  IApplication7  Performs a User Preset.

UserPresetLoadFile  IApplication7  Loads an existing instrument state file (.sta or .cst) to be used for User Preset.
UserPresetSaveState | IApplication7 | Saves the current instrument settings as UserPreset.sta.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveCalKit</td>
<td>Returns a pointer to the kit identified by kitNumber.</td>
</tr>
<tr>
<td>ActiveChannel</td>
<td>Returns a handle to the Active Channel object.</td>
</tr>
<tr>
<td>ActiveMeasurement</td>
<td>Returns a handle to the Active Measurement object.</td>
</tr>
<tr>
<td>ActiveNAWindow</td>
<td>Returns a handle to the Active Window object.</td>
</tr>
<tr>
<td>ArrangeWindows</td>
<td>Sets or returns the arrangement of all the windows.</td>
</tr>
<tr>
<td>AuxiliaryTriggerCount</td>
<td>Returns the number of Aux trigger input / output connector pairs in the instrument.</td>
</tr>
<tr>
<td>CalKitType</td>
<td>Sets or returns the calibration kit type for to be used for calibration or for kit modification. Shared with the CalKit object.</td>
</tr>
<tr>
<td>Capabilities</td>
<td>Return capabilities of the remote VNA.</td>
</tr>
<tr>
<td>Channels</td>
<td>Collection for iterating through the channels</td>
</tr>
<tr>
<td>CoupledMarkers</td>
<td>Sets (or reads) coupled markers ON and OFF</td>
</tr>
<tr>
<td>CoupledMarkersMethod</td>
<td>Set and reads the scope of coupled markers.</td>
</tr>
<tr>
<td>DateTime</td>
<td>Returns the system date and time.</td>
</tr>
<tr>
<td>DisplayAutomationErrors</td>
<td>Enables or disables automation error messages from being displayed on the screen.</td>
</tr>
<tr>
<td>DisplayGlobalPassFail</td>
<td>Shows or hides the dialog which displays global pass/fail results.</td>
</tr>
<tr>
<td>E5091Testsets</td>
<td>Collection to control the E5091A testset.</td>
</tr>
<tr>
<td>ENRFile</td>
<td>Manages Noise ENR files.</td>
</tr>
<tr>
<td>ExternalALC</td>
<td>Sets or returns the source of the analyzer leveling control.</td>
</tr>
<tr>
<td>ExternalDevices</td>
<td>Collection to control External Devices</td>
</tr>
<tr>
<td>ExternalTestsets</td>
<td>Collection to control External Test sets.</td>
</tr>
<tr>
<td>FIFO</td>
<td>Controls FIFO settings</td>
</tr>
<tr>
<td>GlobalPowerLimit</td>
<td>Controls Global Power Limit settings</td>
</tr>
<tr>
<td>GPIBAddress</td>
<td>Sets and returns the VNA GPIB address.</td>
</tr>
</tbody>
</table>
GPIBMode  IApplication  Makes the analyzer the system controller or a talker/listener.

GridLineType  IApplication17  Set and return the line type of the window grid (solid | dotted)

IDString  IApplication  Returns the model, serial number and software revision of the analyzer

InterfaceControl  IApplication8  Control the Interface control features.

IOConfiguration  IApplication19  Provides access to IO Configuration commands

LANConfiguration  IApplication13  Returns the current status of the VNA computer networking configuration.

LXIDeviceIDState  IApplication14  Displays the LAN Status dialog with LAN Status Indicator showing IDENTIFY.

LocalLockoutState  IApplication4  Prevents use of the mouse, keyboard, and front panel while your program is running.

Measurement  IApplication  Create and manage measurements

Measurements  IApplication  Collection for iterating through the Application measurements.

MessageText  IApplication  Returns text for the specified eventID

NaWindows  IApplication  Collection for iterating through the Application windows.

NoiseSourceState  IApplication13  Sets and Reads the ON | OFF state of the noise source

NumberOfPorts  IApplication  Returns the number of hardware test ports on the VNA

Options  IApplication  Returns the options on the analyzer

PathConfigurationManager  IApplication11  Provides access to hardware configuration.

Port Extensions  IApplication  Superseded with Fixturing Object

Preferences  IApplication5  Preferences for many VNA settings.

ScpiStringParser  IApplication  Provides the ability to send a SCPI command from within the COM command.

SecurityLevel  IApplication4  Turns ON or OFF the display of frequency information.

SICL  IApplication5  Allows control of the VNA via SICL

SICLAddress  IApplication8  Sets and returns the VNA SICL address
**SourcePowerCalibrator**  
IApplication2  
Allows capability for performing source power calibrations.

**SourcePowerState**  
IApplication  
Turns Source Power ON and OFF.

**SystemImpedanceZ0**  
IApplication  
Sets the analyzer impedance value.

**SystemName**  
IApplication  
Returns the full computer name of the VNA.

**Touchscreen**  
IApplication12  
Enables and disables touchscreen.

**TriggerDelay**  
IApplication  
Sets or returns the delay time for a trigger.

**TriggerSetup**  
IApplication4  
Controls triggering for the entire VNA application.

**TriggerSignal**  
IApplication  
**Superseded with** Source Property

**TriggerType**  
IApplication  
**Superseded with** Scope Property

**UncertaintyManager**  
IApplication20  
Returns a handle to the (Dynamic) Uncertainty Manager application.

**UserPresetEnable**  
IApplication7  
'Checks' and 'clears' the enable box on the User Preset dialog box.

**VelocityFactor**  
IApplication  
Sets the velocity factor to be used with Electrical Delay, Port Extensions, and Time Domain marker distance calculations.

**Visible**  
IApplication  
Makes the Network Analyzer application visible or not visible.

**WindowState**  
IApplication  
Sets or returns the window setting of Maximized, Minimized, or Normal.

Shared with the NAWindow Object

<table>
<thead>
<tr>
<th>Events</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OnCalEvent</td>
<td>IApplication</td>
<td>Triggered by a calibration event.</td>
</tr>
<tr>
<td>OnChannelEvent</td>
<td>IApplication</td>
<td>Triggered by a channel event.</td>
</tr>
<tr>
<td>OnDisplayEvent</td>
<td>IApplication</td>
<td>Triggered by a display event.</td>
</tr>
<tr>
<td>OnHardwareEvent</td>
<td>IApplication</td>
<td>Triggered by a hardware event.</td>
</tr>
<tr>
<td>OnMeasurementEvent</td>
<td>IApplication</td>
<td>Triggered by a measurement event.</td>
</tr>
<tr>
<td>OnSCPIEvent</td>
<td>IApplication</td>
<td>Triggered by a SCPI event.</td>
</tr>
<tr>
<td>OnSystemEvent</td>
<td>IApplication</td>
<td>Triggered by a system event.</td>
</tr>
</tbody>
</table>
### IApplication History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IApplication</td>
<td>1.0</td>
</tr>
<tr>
<td>IApplication2</td>
<td>3.0</td>
</tr>
<tr>
<td>IApplication3</td>
<td>3.2</td>
</tr>
<tr>
<td>IApplication4</td>
<td>3.5</td>
</tr>
<tr>
<td>IApplication5</td>
<td>4.0</td>
</tr>
<tr>
<td>IApplication6</td>
<td>5.0</td>
</tr>
<tr>
<td>IApplication7</td>
<td>5.0</td>
</tr>
<tr>
<td>IApplication8</td>
<td>5.2</td>
</tr>
<tr>
<td>IApplication9</td>
<td>6.0</td>
</tr>
<tr>
<td>IApplication10</td>
<td>7.20</td>
</tr>
<tr>
<td>IApplication11</td>
<td>7.20</td>
</tr>
<tr>
<td>IApplication12</td>
<td>7.21</td>
</tr>
<tr>
<td>IApplication13</td>
<td>8.0</td>
</tr>
<tr>
<td>IApplication14</td>
<td>8.2</td>
</tr>
<tr>
<td>IApplication15</td>
<td>8.34</td>
</tr>
<tr>
<td>IApplication16</td>
<td>9.0</td>
</tr>
<tr>
<td>IApplication17</td>
<td>9.0</td>
</tr>
<tr>
<td>IApplication18</td>
<td>9.2</td>
</tr>
<tr>
<td>IApplication19</td>
<td>10.0</td>
</tr>
<tr>
<td>IApplication20</td>
<td>10.40</td>
</tr>
<tr>
<td>IApplication22</td>
<td>10.45</td>
</tr>
<tr>
<td>IApplication23</td>
<td>A.12.70</td>
</tr>
</tbody>
</table>
AuxiliaryTrigger Object

Description

These properties setup Auxiliary triggering on a channel.

Accessing the object

Use chan.AuxTrigger(n) to access the object.

where n= the connector pair to be used for Auxiliary Triggering.

- VNA models: Use 1 or 2

Use app.AuxiliaryTriggerCount to determine the number of auxiliary trigger pairs on the rear panel of a VNA.

```vba
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim AuxTrig as AuxTrigger
AuxTrig = chan.AuxTrigger(2)
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Triggering in the VNA
- Example Programs
<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>See History (below)</td>
<td></td>
</tr>
</tbody>
</table>

None

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay (Input)</td>
<td>Specifies the delay that should be applied by the VNA after the Aux trigger input is received and before the acquisition is made</td>
</tr>
<tr>
<td>Enable</td>
<td>Turns ON / OFF the trigger output.</td>
</tr>
<tr>
<td>HandshakeEnable (Input)</td>
<td>Turns handshake ON / OFF.</td>
</tr>
<tr>
<td>Number</td>
<td>Reads the number of the Aux I/O pair being used.</td>
</tr>
<tr>
<td>TriggerInPolarity (Input)</td>
<td>Specifies the polarity of the trigger IN signal to which the VNA will respond.</td>
</tr>
<tr>
<td>TriggerInType (Input)</td>
<td>Specifies the type of Aux trigger input being supplied to the VNA</td>
</tr>
<tr>
<td>TriggerOutDuration</td>
<td>Specifies the width of the pulse or the time that the Aux trigger output will be asserted</td>
</tr>
<tr>
<td>TriggerOutInterval</td>
<td>Specifies how often a trigger output signal is sent.</td>
</tr>
<tr>
<td>TriggerOutPolarity</td>
<td>Specifies the polarity of the trigger output signal being supplied by the VNA.</td>
</tr>
<tr>
<td>TriggerOutPosition</td>
<td>Specifies whether the Aux trigger out signal is sent Before or After the acquisition.</td>
</tr>
</tbody>
</table>

**IAuxTrigger History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAuxTrigger</td>
<td>7.2</td>
</tr>
</tbody>
</table>
BalancedMeasurement Object

Description

These properties set the measurement type that is used with balanced topologies.

Use the BalancedTopology Object to set the topology and port mappings for the DUT,

Accessing the BalancedMeasurement object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim balMeas As BalancedMeasurement
Set balMeas = app.ActiveMeasurement.BalancedMeasurement
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- About Balanced Measurements
- Example Programs

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BalancedMode</td>
<td>IBalancedMeasurement</td>
<td>Sets and returns whether the balanced transform is ON or OFF.</td>
</tr>
<tr>
<td>BalancedStimulus</td>
<td>IBalancedMeasurement2</td>
<td>Sets and returns the stimulus properties of a balanced DUT.</td>
</tr>
<tr>
<td>BalancedTopology</td>
<td>IBalancedMeasurement</td>
<td>Sets and returns the topology of a balanced DUT.</td>
</tr>
</tbody>
</table>
BalSMeasurement  IBalancedMeasurement3  Sets and returns the measurement for the Balanced - Single-ended topology.

BBalMeasurement  IBalancedMeasurement  Sets and returns the measurement for the Balanced - Balanced topology.


### IBalancedMeasurement History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBalancedMeasurement</td>
<td>5.0</td>
</tr>
<tr>
<td>IBalancedMeasurement2</td>
<td>8.2</td>
</tr>
<tr>
<td>IBalancedMeasurement3</td>
<td>9.70</td>
</tr>
<tr>
<td>IBalancedMeasurement4</td>
<td>14.20</td>
</tr>
</tbody>
</table>
BalancedStimulus Object

Description

These properties set the values that are unique to iTMSA - Opt S93460A/B.

All other properties for iTMSA use the standard VNA commands.

Accessing the BalancedStimulus object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim balStim As BalancedStimulus
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- About iTMSA
- Example iTMSA Program

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BalancedPortTrueState</td>
<td>IBalancedStimulus3</td>
<td>Sets the True Mode state of a balanced port.</td>
</tr>
<tr>
<td>BallPort1PhaseOffset</td>
<td>IBalancedStimulus</td>
<td>Sets balanced port 1 phase offset</td>
</tr>
<tr>
<td>BallPort1PowerOffset</td>
<td>IBalancedStimulus</td>
<td>Sets balanced port 1 power offset</td>
</tr>
<tr>
<td>BallPort1StartPhase</td>
<td>IBalancedStimulus2</td>
<td>Sets Phase start value for port 1</td>
</tr>
<tr>
<td>BallPort1StopPhase</td>
<td>IBalancedStimulus2</td>
<td>Sets Phase stop value for port 1</td>
</tr>
<tr>
<td>BallPort2PhaseOffset</td>
<td>IBalancedStimulus</td>
<td>Sets balanced port 2 phase offset</td>
</tr>
</tbody>
</table>
BalPort2PowerOffset  IBalancedStimulus  Sets balanced port 2 power offset
BalPort2StartPhase  IBalancedStimulus2  Sets Phase start value for port 2
BalPort2StopPhase  IBalancedStimulus2  Sets Phase start value for port 2
Mode  IBalancedStimulus  Sets Stimulus mode for balanced measurements
PhaseAsFixture  IBalancedStimulus  Sets the state of phase offset as a fixture
PhaseSwpAsFixture  IBalancedStimulus2  Enable Phase Sweep as fixture
PhaseSwpState  IBalancedStimulus2  Enable Phase Sweep
PowerAsFixture  IBalancedStimulus  Sets the state of power offset as a fixture

IBalancedStimulus History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBalancedStimulus</td>
<td>8.2</td>
</tr>
<tr>
<td>IBalancedStimulus2</td>
<td>8.5</td>
</tr>
<tr>
<td>IBalancedStimulus3</td>
<td>14.40</td>
</tr>
</tbody>
</table>
**BalancedTopology Object**

**Description**

The DUTTopology property sets and returns the topology of a balanced DUT.

The following methods *set* the port mappings for the DUT.

The remaining properties *return* the port mappings for the DUT.

Use the BalancedMeasurement object to set the measurement type.

**Accessing the BalancedTopology object**

```vba
dim app as AgilentPNA835x.Application
dim chan as Channel
set chan = app.ActiveChannel

dim balTopology as BalancedTopology
set balTopology = chan.BalancedTopology
```

**See Also:**

- VNA Automation Interfaces
- The VNA Object Model
- About Balanced Measurements
- Example Programs

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetBPort</td>
<td>IBalancedTopology3</td>
<td>Sets the physical port mappings for the single Balanced port DUT topology.</td>
</tr>
<tr>
<td>SetCustomDUTTopology</td>
<td>IBalancedTopology3</td>
<td>Sets a custom topology for instruments with greater than 4 physical ports.</td>
</tr>
<tr>
<td>SetBBPorts</td>
<td>IBalancedTopology</td>
<td>Sets the physical port mappings for the Balanced - Balanced DUT topology.</td>
</tr>
<tr>
<td>SetBSPorts</td>
<td>IBalancedTopology2</td>
<td>Sets the physical port mappings for the Balanced - Single-Ended DUT topology.</td>
</tr>
</tbody>
</table>

SetSBPorts | IBalancedTopology | Sets the physical port mappings for the Single-Ended - Balanced DUT topology.


<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B_BalPortNegative</td>
<td>IBalancedTopology3</td>
<td>Returns the negative balanced port number in the Balanced DUT topology.</td>
</tr>
<tr>
<td>B_BalPortPositive</td>
<td>IBalancedTopology3</td>
<td>Returns the positive balanced port number in the Balanced DUT topology.</td>
</tr>
<tr>
<td>BB_BalPort1Negative</td>
<td>IBalancedTopology</td>
<td>Returns the VNA port number that is connected to the Negative side of the DUT's logical Port 1.</td>
</tr>
<tr>
<td>BB_BalPort1Positive</td>
<td>IBalancedTopology</td>
<td>Returns the first positive balanced port number in the Balanced - Balanced topology.</td>
</tr>
<tr>
<td>BB_BalPort2Negative</td>
<td>IBalancedTopology</td>
<td>Returns the second negative balanced port number in the Balanced - Balanced topology.</td>
</tr>
<tr>
<td>BB_BalPort2Positive</td>
<td>IBalancedTopology</td>
<td>Returns the second positive balanced port number in the Balanced - Balanced topology.</td>
</tr>
<tr>
<td>BS_BalPortNegative</td>
<td>IBalancedTopology2</td>
<td>Returns the negative balanced port number in the Balanced - Single-ended topology.</td>
</tr>
<tr>
<td>BS_BalPortPositive</td>
<td>IBalancedTopology2</td>
<td>Returns the positive balanced port number in the Balanced - Single-ended topology.</td>
</tr>
<tr>
<td>BS_SEPort</td>
<td>IBalancedTopology2</td>
<td>Returns the single-ended port number in the Balanced - Single-ended topology.</td>
</tr>
<tr>
<td>BSS_BalPortNegative</td>
<td>IBalancedTopology4</td>
<td>Returns the negative balanced port number in the Balanced - Single-ended - Single-ended topology.</td>
</tr>
</tbody>
</table>


CustomBalNegativePort  IBalancedTopology3  Returns the negative physical port for the specified balanced port.

CustomBalPositivePort  IBalancedTopology3  Returns the positive physical port for the specified balanced port.

CustomPhysicalPortsSequence  IBalancedTopology3  Returns an array of physical port numbers corresponding to the CustomPortTypeSequence.

CustomPortTypeSequence  IBalancedTopology3  Returns a string containing characters "S" or "B" conveying the type of each logical port.

CustomSEPhysicalPort  IBalancedTopology3  Returns the physical port for the specified single-ended port.

CustomTopologyPortCount  IBalancedTopology3  Returns the number of configured logical ports.

DUTTopology  IBalancedTopology  Sets and returns the device topology setting.

IsBalanced  IBalancedTopology3  Returns true if the port is balanced.

IsSingleEnded  IBalancedTopology3  Returns true if the port is single ended.

SB_BalPortNegative  IBalancedTopology  Returns the negative balanced port number in the Single-Ended - Balanced topology.

SB_BalPortPositive  IBalancedTopology  Returns the positive balanced port number in the Single-Ended - Balanced topology.

SB_SEPort  IBalancedTopology  Returns the single ended port number in the Single-Ended - Balanced topology.

SSB_BalPortNegative  IBalancedTopology  Returns the negative balanced port number in the Single-Ended - Single-Ended - Balanced topology.


BalancedTopology History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBalancedTopology</td>
<td>5.0</td>
</tr>
<tr>
<td>IBalancedTopology2</td>
<td>9.70</td>
</tr>
<tr>
<td>IBalancedTopology3</td>
<td>12.70</td>
</tr>
<tr>
<td>IBalancedTopology4</td>
<td>14.20</td>
</tr>
</tbody>
</table>
CalFactorSegments Collection

Description

A collection object that provides a mechanism for iterating through the segments of a power sensor cal factor table. The Cal Factor table can contain up to 100 segments.

Accessing the CalFactorSegments collection

```vbnet
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim calFact As CalFactorSegments
```

See Also:

- PowerSensorCalFactorSegment Object
- About Source Power Cal
- Collections in the Analyzer
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a PowerSensorCalFactorSegment object to the collection</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle to a PowerSensorCalFactorSegment object in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes an object from the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of objects in the collection.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the Parent object (PowerSensor) of this collection.</td>
</tr>
</tbody>
</table>
**CalFactorSegmentsPMAR Collection**

**Description**

A collection object that provides a mechanism for iterating through the segments of a power sensor cal factor table. The Cal Factor table can contain up to 100 segments. This collection is used when the Power Meter is used as a Receiver.

**Accessing the CalFactorSegmentsPMAR collection**

Example: Create a PMAR Device and Measurement

**See Also:**

- PowerSensorCalFactorSegmentPMAR Object
- About PMAR
- Collections in the Analyzer
- The VNA Object Model

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a PowerSensorCalFactorSegmentPMAR object to the collection</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle to a PowerSensorCalFactorSegmentPMAR object in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes an PowerSensorCalFactorSegmentPMAR object from the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of PowerSensorCalFactorSegmentPMAR objects in the collection.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the Parent object of this collection.</td>
</tr>
</tbody>
</table>
CalibrateAllChannels Object

Description
Use this interface to Calibrate All Channels.
See Cal All Channels limitations.

Accessing the CalibrateAllChannels Object

```
Dim app as AgilentPNA835x.Application
Dim CalMgr
Set CalMgr = app.GetCalManager
Dim calAll
Set calAll = CalMgr.CalibrateAllChannels
' Then get a handle to the GuidedCal object
Dim guidedCal
Set guidedCal = calAll.GuidedCalibration
```

See Also:

- Example program
- Learn about Calibrate All Channels
- VNA Automation Interfaces
- The VNA Object Model
- Superseded commands

(Bold Methods or Properties provide access to a child object)

In the following table ICalibrateAllChannels is abbreviated to ICalAllChans

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>See History</td>
</tr>
</tbody>
</table>

980
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reset</strong></td>
<td>Resets all properties associated with the Cal All session to their default values.</td>
</tr>
<tr>
<td><strong>Properties</strong></td>
<td></td>
</tr>
<tr>
<td>CalibrationPorts</td>
<td>For each channel, sets and returns the ports to be calibrated.</td>
</tr>
<tr>
<td>Channels</td>
<td>Sets and returns the list of channels to be calibrated during the Cal All session.</td>
</tr>
<tr>
<td>GeneratedCalsets</td>
<td>Returns the cal set names that were produced by the cal all session.</td>
</tr>
<tr>
<td>GuidedCalibration</td>
<td>Provides access to the GuidedCal object. Use this to perform the Calibration.</td>
</tr>
<tr>
<td>IFBW</td>
<td>Sets and returns the IFBW for a Cal All calibration.</td>
</tr>
<tr>
<td>IndependentPowerCalibration</td>
<td>Returns a handle to an IndependentPowerCalibration object. Use this to add a power calibration for any port during a Cal All.</td>
</tr>
<tr>
<td>PathConfigurationElement</td>
<td>Sets and returns the Path Configuration settings for a Cal All calibration.</td>
</tr>
<tr>
<td>PowerLevel</td>
<td>Sets and returns the power level at which a Cal All calibration is to be performed.</td>
</tr>
<tr>
<td>PowerOffset</td>
<td>Sets and returns the power offset value for a Cal All calibration.</td>
</tr>
<tr>
<td>PropertyNames</td>
<td>Returns the settable properties for the current cal all session.</td>
</tr>
</tbody>
</table>
PropertyNamesByMeasurementClass
ICalAllChans
Returns the list of settable call properties associated with the specified measurement class.

PropertyValue
ICalAllChans
Sets and returns a value for a specific property name.

PropertyValues
ICalAllChans
Returns the valid property values for a specific property name.

ReceiverAttenuator
ICalAllChans
Sets and returns the Receiver Attenuator setting for a Cal All calibration.

SourceAttenuator
ICalAllChans
Sets and returns the Source Attenuator setting for a Cal All calibration.

SParameterCalPorts
ICalAllChans
Returns a list of ports to be calibrated.

UserCalsetPrefix
ICalAllChans
Sets and returns the prefix to be used when saving User CalSets that result from the Cal All session.

ICalibrateAllChannels History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalibrateAllChans</td>
<td>9.50</td>
</tr>
<tr>
<td>ICalibrateAllChannels2</td>
<td>12.80</td>
</tr>
</tbody>
</table>
Calibrator Object

See Also

- Example Programs
- Calibrator Methods and Properties
- ICalData Interface for putting and getting typed Calibration data.
- Superseded commands

Description

The Calibrator object, a child of the channel, is used to perform an **Unguided** calibration.

**Important!**

Do **NOT** use commands from the GuidedCalibration object when performing an Unguided calibration. Use ONLY the Calibrator object.

You can **NOT** perform a full 3 or 4-port using the Calibrator object. You must use the GuidedCalibration object.

There must be a measurement present for the calibrator to use or you will receive a "no measurement found" error. Therefore, to perform a 2-port cal, you must have any S-parameter measurement on the channel. For a 1-port measurement, you must have the measurement (S11 or S22) on the channel. The same is true for a response measurement.

There are a number of approaches to calibration with the calibrator object:

- You can collect data yourself and download it to the ACQUISITION buffer. The acquisition buffer holds the actual measured data for each standard. See the VNA data map.

  1. Calibrator.SetCalInfo
  2. Connect a standard
  3. Trigger a sweep
  4. Retrieve the data for the standard
  5. Download the data - calibrator.putStandard
6. Repeat for each standard
7. Calibrator.CalculateErrorCoefficients

- You can tell the calibrator to acquire a standard. In this case, the calibrator collects the data and places it in the ACQUISITION buffer.

1. Calibrator.SetCalInfo
2. Connect a standard
3. Calibrator.AcquireCalStandard
4. Repeat for each standard
5. Calibrator.CalculateErrorCoefficients

- You can put previously-retrieved error terms in the error correction buffer.

1. PutErrorTerm
2. Repeat for each term
3. Measurement.Caltype = pick one

- You can also "piece together" a 2-port cal from two 1-port cals (S11 and S22) and four response (thru) cals. The system will detect that all the standards needed for a 2-port cal have been acquired even though they may not have gathered at the same time.

**Accessing the Calibrator object**

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim cal As ICalibrator
Set cal = app.ActiveChannel.Calibrator
```

**See Also:**

- VNA Automation Interfaces
- The VNA Object Model
- Learn about reading and writing Calibration data.
<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcquireCalConfidenceCheckECAL</td>
<td>ICalibrator</td>
<td><strong>Superseded with</strong> AcquireCalConfidenceCheckECALEx</td>
</tr>
<tr>
<td>AcquireCalConfidenceCheckECALEx</td>
<td>ICalibrator4</td>
<td>Transfers ECAL confidence data into analyzer memory</td>
</tr>
<tr>
<td>AcquireCalStandard</td>
<td>ICalibrator</td>
<td><strong>Superseded with</strong> AcquireCalStandard2</td>
</tr>
<tr>
<td>AcquireCalStandard2</td>
<td>ICalibrator</td>
<td>Causes the analyzer to measure a calibration standard. Also provides for sliding load.</td>
</tr>
<tr>
<td>CalculateErrorCoefficients</td>
<td>ICalibrator</td>
<td>Generates Error Terms from standard and actual data in the error correction buffer.</td>
</tr>
<tr>
<td>DoECAL1Port</td>
<td>ICalibrator</td>
<td><strong>Superseded with</strong> DoECAL1PortEx</td>
</tr>
<tr>
<td>DoECAL1PortEx</td>
<td>ICalibrator4</td>
<td>Completes a 1 port ECAL</td>
</tr>
<tr>
<td>DoECAL2Port</td>
<td>ICalibrator</td>
<td><strong>Superseded with</strong> DoECAL2PortEx</td>
</tr>
<tr>
<td>DoECAL2PortEx</td>
<td>ICalibrator4</td>
<td>Completes a 2 port ECAL</td>
</tr>
<tr>
<td>DoneCalConfidenceCheckECAL</td>
<td>ICalibrator</td>
<td>Concludes an ECAL confidence check</td>
</tr>
<tr>
<td>DoReceiverPowerCal</td>
<td>ICalibrator5</td>
<td>Perform a receiver power cal.</td>
</tr>
<tr>
<td>DoResponseCal</td>
<td>ICalibrator9</td>
<td>Perform a response (normalization) cal.</td>
</tr>
<tr>
<td>GetCalKitTypeString</td>
<td>ICalibrator8</td>
<td>Returns information about the attached modules</td>
</tr>
<tr>
<td>GetECALModuleInfo</td>
<td>ICalibrator</td>
<td><strong>Superseded with</strong> Get ECALModuleInfoEx</td>
</tr>
<tr>
<td>Get ECALModuleInfoEx</td>
<td>ICalibrator4</td>
<td>Returns information about the attached module</td>
</tr>
<tr>
<td>getErrorTerm</td>
<td>ICalibrator</td>
<td><strong>Superseded with</strong> GetErrorTermByString</td>
</tr>
<tr>
<td>getStandard</td>
<td>ICalibrator</td>
<td><strong>Superseded with</strong> GetStandardByString</td>
</tr>
<tr>
<td>putErrorTerm</td>
<td>ICalibrator</td>
<td><strong>Superseded with</strong> PutErrorTermByString</td>
</tr>
<tr>
<td>putStandard</td>
<td>ICalibrator</td>
<td><strong>Superseded with</strong> PutStandardByString</td>
</tr>
<tr>
<td>SaveCalSets</td>
<td>ICalibrator</td>
<td><strong>Superseded with</strong> CalSet.Save</td>
</tr>
<tr>
<td>setCalInfo</td>
<td>ICalibrator</td>
<td>Specifies the type of calibration and prepares the internal state for the rest of the calibration.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
</table>

985
<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcquisitionDirection</td>
<td>ICalibrator</td>
<td>Specifies the direction in a 2-Port cal using one set of standards.</td>
</tr>
<tr>
<td>CalKitType</td>
<td>ICalibrator10</td>
<td>Sets and returns the name of the Cal Kit to use for unguided cal.</td>
</tr>
<tr>
<td>CalKitTypes</td>
<td>ICalibrator10</td>
<td>Returns the names of the first 50 mechanical cal kits in your VNA that can be used for unguided calibrations.</td>
</tr>
<tr>
<td>ECALCharacterization</td>
<td>ICalibrator2</td>
<td><strong>Superseded with</strong> ECALCharacterizationEx</td>
</tr>
<tr>
<td>ECALCharacterizationEx</td>
<td>ICalibrator4</td>
<td>Specifies which set of characterization data within an ECal module will be used for ECal operations with that module.</td>
</tr>
<tr>
<td>ECALCharacterizationIndexList</td>
<td>ICalibrator6</td>
<td>Returns a list of characterizations stored in the specified ECal module.</td>
</tr>
<tr>
<td>ECAL Isolation</td>
<td>ICalibrator</td>
<td>Specifies whether the acquisition of the ECal calibration should include isolation or not.</td>
</tr>
<tr>
<td>ECALModuleNumberList</td>
<td>ICalibrator6</td>
<td>Returns a list of index numbers to be used for referring to the ECal modules that are currently attached to the VNA.</td>
</tr>
<tr>
<td>ECALPortMap</td>
<td>ICalibrator3</td>
<td><strong>Superseded with</strong> ECALPortMapEx</td>
</tr>
<tr>
<td>ECALPortMapEx</td>
<td>ICalibrator4</td>
<td>Specifies which ports of the ECal module are connected to which ports of the VNA.</td>
</tr>
<tr>
<td>IsECALModuleFound</td>
<td>ICalibrator</td>
<td><strong>Superseded with</strong> IsECALModuleFoundEx</td>
</tr>
<tr>
<td>IsECALModuleFoundEx</td>
<td>ICalibrator4</td>
<td><strong>Superseded with</strong> ECALCharacterizationIndexList and ECALModuleNumberList</td>
</tr>
<tr>
<td>IsolationAveragingIncrement</td>
<td>ICalibrator7</td>
<td>Value to increase the channel's averaging factor.</td>
</tr>
<tr>
<td>OrientECALModule</td>
<td>ICalibrator3</td>
<td>Specifies if the VNA should perform orientation of the ECal module during calibration.</td>
</tr>
<tr>
<td>Simultaneous2Port Acquisition</td>
<td>ICalibrator</td>
<td>Allows the use of 2 sets of standards at the same time.</td>
</tr>
</tbody>
</table>

**ICalibrator History**
### ICalData Interface

**Description**

Contains methods for putting Calibration data in and getting Calibration data out of the analyzer using typed data. This interface transfers data more efficiently than variant data. However, this interface is only usable from VB6, C, & C++. All other programming languages must use the ICalSet interface.

There is also an **ICalData Interface** on the CalSet Object

Learn about reading and writing Calibration data.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getErrorTermComplex</td>
<td>Retrieves error term data</td>
</tr>
<tr>
<td>getStandardComplex</td>
<td>Retrieves calibration data from the acquisition data buffer (before error-terms are applied).</td>
</tr>
<tr>
<td>putErrorTermComplex</td>
<td>Puts error term data</td>
</tr>
<tr>
<td>putStandardComplex</td>
<td>Puts calibration data into the acquisition data buffer (before error-terms are applied).</td>
</tr>
</tbody>
</table>

**Properties**

None

**ICalData History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalibrator</td>
<td>1.0</td>
</tr>
<tr>
<td>ICalibrator2</td>
<td>3.1</td>
</tr>
<tr>
<td>ICalibrator3</td>
<td>3.1</td>
</tr>
<tr>
<td>ICalibrator4</td>
<td>3.5</td>
</tr>
<tr>
<td>ICalibrator5</td>
<td>5.0</td>
</tr>
<tr>
<td>ICalibrator6</td>
<td>5.26</td>
</tr>
<tr>
<td>ICalibrator7</td>
<td>7.21</td>
</tr>
<tr>
<td>ICalibrator8</td>
<td>8.1</td>
</tr>
<tr>
<td>ICalibrator9</td>
<td>9.1</td>
</tr>
<tr>
<td>ICalibrator10</td>
<td>9.2</td>
</tr>
<tr>
<td>Interface</td>
<td>Introduced with VNA Rev:</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>ICalData</td>
<td>1.0</td>
</tr>
</tbody>
</table>
CalKit Object

Description
The calkit object provides the properties and methods to access and modify a calibration kit.

Accessing a CalKit object
The active cal kit is the kit that is selected for use in Unguided calibrations. To get a handle to the active kit, use the app.ActiveCalKit property. To access the CalKit object for a specific cal kit, you must first make that kit the active kit using app.CalKitType.

The CalKit object behaves differently from other objects in that you can only have a handle to one cal kit -- the active cal kit. Therefore, when you change the CalKitType from either the Application object or the CalKit object, you may also be changing the object to which you may have other references.

For example, the following example specifies two CalKit type objects and in turn, assigns them to two different variables: ck1 and ck2.

```vba
Dim app As AgilentPNA835x.Application
Dim ck1 As calKit
Dim ck2 As calKit
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
app.CalKitType = naCalKit_User1
Set ck1 = app.ActiveCalKit
ck1.Name = "My CalKit1"
app.CalKitType = naCalKit_User2
Set ck2 = app.ActiveCalKit
ck2.Name = "My CalKit2"

Print "ck1: " & ck1.Name
Print "ck2: " & ck2.Name
```

When the pointer to each of these kits is read (printed), they each have a pointer to the last kit to be assigned to the active cal kit:

ck1: My CalKit2
ck2: My CalKit2

See Also:

- VNA Automation Interfaces
- The VNA Object Model
Example Programs

*(Bold Methods or Properties provide access to a child object)*

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getCalStandard</td>
<td>Returns a handle to a calibration standard for modifying its definitions.</td>
</tr>
<tr>
<td>GetStandardsForClass</td>
<td>Returns the calibration standard numbers for a specified calibration class.</td>
</tr>
<tr>
<td>SetStandardsForClass</td>
<td>Sets the calibration standard numbers for a specified calibration class.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalKitType</td>
<td>Sets or returns the calibration kit type to be used for calibration or for kit modification.</td>
</tr>
<tr>
<td></td>
<td>Shared with the Application object.</td>
</tr>
<tr>
<td>Name</td>
<td>Sets and returns the name of the cal kit</td>
</tr>
<tr>
<td>PortLabel</td>
<td>Labels the ports for the kit; only affects the cal wizard annotation.</td>
</tr>
<tr>
<td>StandardForClass</td>
<td><strong>Superseded with</strong>  Use GetStandardForClass and SetStandardForClass.</td>
</tr>
<tr>
<td></td>
<td>Maps a standard device to a cal class.</td>
</tr>
</tbody>
</table>

ICalKit History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalKit</td>
<td>1.0</td>
</tr>
</tbody>
</table>
CalManager Object

Description

Use this interface to list, save, and delete Cal Sets.

Accessing the CalManager object

Get a handle to a the CalManager with the app.GetCalManager Method.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim mgr as ICalManager
Set mgr = app.GetCalManager
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
- Superseded commands

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowChannelToSweepDuringCalAcquisition</td>
<td>ICalManager5</td>
<td>Specifies the channel to sweep during a Calibration.</td>
</tr>
<tr>
<td><strong>CalibrateAllChannels</strong></td>
<td>ICalManager9</td>
<td>Provides access to CalibrateAllChannels object.</td>
</tr>
<tr>
<td>CascadeS2PFiles</td>
<td>ICalManager10</td>
<td>Creates a single S2P file from two existing files.</td>
</tr>
<tr>
<td>Function</td>
<td>Module</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CharacterizeFixture</td>
<td>ICalManager10</td>
<td>Characterizes an adapter/fixture based on two calsets.</td>
</tr>
<tr>
<td>CreateCalSet</td>
<td>ICalManager</td>
<td>Creates a new Cal Set</td>
</tr>
<tr>
<td><strong>CreateCustomCal</strong></td>
<td>ICalManager2</td>
<td>Creates an FCA cal object.</td>
</tr>
<tr>
<td><strong>CreateCustomCalEx</strong></td>
<td>ICalManager5</td>
<td>Creates a custom cal object.</td>
</tr>
<tr>
<td>Deembed</td>
<td>ICalManager8</td>
<td>De-embeds a fixture from an existing Cal Set based on an S2P file.</td>
</tr>
<tr>
<td>DeleteCalSet</td>
<td>ICalManager</td>
<td>Deletes a Cal Set</td>
</tr>
<tr>
<td>DisplayNAWindowDuringCalAcquisition</td>
<td>ICalManager5</td>
<td>Set the 'show' state of the window to be displayed during a calibration.</td>
</tr>
<tr>
<td>DisplayOnlyCalWindowDuringCalAcquisition</td>
<td>ICalManager5</td>
<td>Clears the flags for windows to be shown during calibrations.</td>
</tr>
<tr>
<td>Embed</td>
<td>ICalManager8</td>
<td>Embeds a fixture into an existing Cal Set based on an S2P file.</td>
</tr>
<tr>
<td>ENREmbedAdapter</td>
<td>ICalManager11</td>
<td>Generate a new ENR file by embedding an adapter to an existing ENR file.</td>
</tr>
<tr>
<td>EnumerateCalSets</td>
<td>ICalManager4</td>
<td>Returns an array of Cal Set names being stored on the VNA.</td>
</tr>
<tr>
<td>GetCalSetByGUID</td>
<td>ICalManager</td>
<td>Get a handle to a Cal Set</td>
</tr>
<tr>
<td>GetCalSetCatalog</td>
<td>ICalManager</td>
<td><strong>Superseded with</strong> EnumerateCalSets</td>
</tr>
<tr>
<td>Method</td>
<td>Module</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------</td>
<td>------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GetCalSetUsageInfo</td>
<td>ICalManager</td>
<td>Returns the Cal Set ID and Error Term ID currently in use</td>
</tr>
<tr>
<td>GetCalTypes</td>
<td>ICalManager2</td>
<td>Query for a list of available calibration types.</td>
</tr>
<tr>
<td>GetEcalUserCharacterizer</td>
<td>ICalManager6</td>
<td>Returns the ECalUserCharacterizer object.</td>
</tr>
<tr>
<td>SaveCalSets</td>
<td>ICalManager</td>
<td><strong>Superseded with</strong> CalSet.Save</td>
</tr>
<tr>
<td>SweepOnlyCalChannelDuringCalAcquisition</td>
<td>ICalManager5</td>
<td>Clears ALL flags for channels to sweep during calibration.</td>
</tr>
<tr>
<td>ZeroTermsInS4PFile</td>
<td>ICalManager12</td>
<td>Creates a new S4P file.</td>
</tr>
</tbody>
</table>

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalibrateAllChannelsEx</td>
<td>ICalManager13</td>
<td>Provides access to CalibrateAllChannels object for setting up more than one CalibrateAllChannels instances. This command takes an input calibration number. See the programming example.</td>
</tr>
<tr>
<td>CalSets Collection</td>
<td>ICalManager</td>
<td>Collection for iterating through all the Cal Sets in the analyzer.</td>
</tr>
<tr>
<td>ECalModules Collection</td>
<td>ICalManager7</td>
<td>Collection of ECal Modules that are connected to the VNA.</td>
</tr>
<tr>
<td>GuidedCalibration</td>
<td>ICalManager3</td>
<td>Used to perform a Guided Calibration.</td>
</tr>
</tbody>
</table>
PhaseReferenceCalibration

ICalManager9 Used to perform a Phase Reference Calibration.

ICalManager History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalManager</td>
<td>2.0</td>
</tr>
<tr>
<td>CalManager2</td>
<td>3.1</td>
</tr>
<tr>
<td>CalManager3</td>
<td>3.5</td>
</tr>
<tr>
<td>CalManager4</td>
<td>5.0</td>
</tr>
<tr>
<td>ICalManager5</td>
<td>8.0</td>
</tr>
<tr>
<td>ICalManager6</td>
<td>8.3</td>
</tr>
<tr>
<td>ICalManager7</td>
<td>8.5</td>
</tr>
<tr>
<td>ICalManager8</td>
<td>9.33</td>
</tr>
<tr>
<td>ICalManager9</td>
<td></td>
</tr>
<tr>
<td>ICalManager10</td>
<td>10.15</td>
</tr>
<tr>
<td>ICalManager11</td>
<td>12.50.01</td>
</tr>
<tr>
<td>ICalManager12</td>
<td></td>
</tr>
<tr>
<td>ICalManager13</td>
<td>12.90</td>
</tr>
</tbody>
</table>
CalSet Object

See ICalData Interface for putting and getting typed Cal Set data.

**Description**

Use this interface to query and or change the contents of a Cal Set.

**Accessing the CalSet object**

Get a handle to a CalSet object by using the CalSets collection. This is done through the CalManager object with the app.GetCalManager Method.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim calst As ICalSet
Set calst = app.GetCalManager.CalSets.Item(1)

' OR Get a handle by CalSet Name
Set calst = app.GetCalManager.CalSets.Item("MyCalSet")
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Reading and Writing Calibration data
- Example Programs
- Superseded commands

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CloseCalSet</td>
<td>ICalSet</td>
<td><strong>Obsolete</strong> - No longer necessary.</td>
</tr>
<tr>
<td>ComputeErrorTerms</td>
<td>ICalSet</td>
<td>Computes error terms for the CalType specified by a preceding OpenCal Set call.</td>
</tr>
<tr>
<td>Copy</td>
<td>ICalSet</td>
<td>Creates a new Cal Set and copies the current Cal Set data into it.</td>
</tr>
<tr>
<td>EnumerateItems</td>
<td>ICalSet6</td>
<td>Returns a list of all name-value pairs (items) in the Cal Set.</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>getErrorTerm</td>
<td>ICalSet</td>
<td><strong>Superseded with</strong> putErrorTermByString</td>
</tr>
<tr>
<td>GetErrorTermByString</td>
<td>ICalSet2</td>
<td>Returns variant error term data by specifying the string name of the error term.</td>
</tr>
<tr>
<td>getErrorTermList</td>
<td>ICalSet</td>
<td><strong>Superseded with</strong> getErrorTermList2</td>
</tr>
<tr>
<td>getErrorTermList2</td>
<td>ICalSet2</td>
<td>Returns a list of error term names found in a calset.</td>
</tr>
<tr>
<td>GetErrorTermStimulus</td>
<td>ICalSet7</td>
<td>Returns the stimulus values over which the specific error term was acquired.</td>
</tr>
<tr>
<td>GetGUID</td>
<td>ICalSet</td>
<td>Returns the GUID identifying a Cal Set</td>
</tr>
<tr>
<td>getStandard</td>
<td>ICalSet</td>
<td><strong>Superseded with</strong> getStandardByString</td>
</tr>
<tr>
<td>getStandardByString</td>
<td>ICalSet2</td>
<td>Returns variant standard acquisition data by specifying the string name of the standard.</td>
</tr>
<tr>
<td>getStandardsList</td>
<td>ICalSet</td>
<td><strong>Superseded with</strong> getStandardList2</td>
</tr>
<tr>
<td>getStandardList2</td>
<td>ICalSet2</td>
<td>Returns a list of standard names found in a Cal Set.</td>
</tr>
<tr>
<td>HasCalType</td>
<td>ICalSet</td>
<td>Verifies that the Cal Set object contains the error terms required to apply the specified CalType to an appropriate measurement.</td>
</tr>
<tr>
<td>OpenCalSet</td>
<td>ICalSet</td>
<td><strong>Obsolete</strong> - No longer necessary.</td>
</tr>
<tr>
<td>putErrorTerm</td>
<td>ICalSet</td>
<td><strong>Superseded with</strong> putErrorTermByString</td>
</tr>
<tr>
<td>PutErrorTermByString</td>
<td>ICalSet2</td>
<td>Writes variant error term data by specifying the string name of the error term.</td>
</tr>
<tr>
<td>PutErrorTermStimulus</td>
<td>ICalSet7</td>
<td>Adds stimulus data to the specified buffer.</td>
</tr>
<tr>
<td>putStandard</td>
<td>ICalSet</td>
<td><strong>Superseded with</strong> putStandardByString</td>
</tr>
<tr>
<td>putStandardByString</td>
<td>ICalSet2</td>
<td>Writes variant standard acquisition data by specifying the string name of the standard.</td>
</tr>
<tr>
<td>RemoveItem</td>
<td>ICalSet6</td>
<td>Removes a name-value pair from the Cal Set.</td>
</tr>
<tr>
<td>Save</td>
<td>ICalSet</td>
<td>Saves the current Cal Set to disk.</td>
</tr>
<tr>
<td>StringToNACalClass</td>
<td>ICalSet</td>
<td>Converts string values from GetStandardsList into enumeration data</td>
</tr>
<tr>
<td>StringToNAErrorTerm2</td>
<td>ICalSet</td>
<td>Converts string values from GetErrorTermList into enumeration data</td>
</tr>
</tbody>
</table>

**Properties**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>996</td>
</tr>
<tr>
<td>Function</td>
</tr>
<tr>
<td>---------------------------</td>
</tr>
<tr>
<td>AlternateSweep</td>
</tr>
<tr>
<td>Attenuator</td>
</tr>
<tr>
<td>AttenuatorMode</td>
</tr>
<tr>
<td>ChannelClients</td>
</tr>
<tr>
<td>ContentDescriptor</td>
</tr>
<tr>
<td>CouplePorts</td>
</tr>
<tr>
<td>CWFrequency</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>DwellTime</td>
</tr>
<tr>
<td>FrequencyOffsetCWOVERRIDE</td>
</tr>
<tr>
<td>FrequencyOffsetDivisor</td>
</tr>
<tr>
<td>FrequencyOffsetFrequency</td>
</tr>
<tr>
<td>FrequencyOffsetMultiplier</td>
</tr>
<tr>
<td>FrequencyOffsetState</td>
</tr>
<tr>
<td>IFBandwidth</td>
</tr>
<tr>
<td>Item</td>
</tr>
<tr>
<td>LastModified</td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>OutputPorts</td>
</tr>
<tr>
<td>PowerSlope</td>
</tr>
<tr>
<td>Properties</td>
</tr>
<tr>
<td>ReceiverAttenuator</td>
</tr>
<tr>
<td>StartFrequency</td>
</tr>
<tr>
<td>StartPower</td>
</tr>
</tbody>
</table>
StimulusValues ICalSet3 Returns x-axis values for stimulus or response frequencies

StopFrequency ICalSet3 Returns the stop frequency of the channel.

StopPower ICalSet3 Returns the stop power of the VNA when sweep type is set to Power Sweep.

SweepGenerationMode ICalSet3 Returns the method being used to generate a sweep: analog or stepped.

SweepTime ICalSet3 Returns the sweep time of the analyzer.

SweepType ICalSet3 Returns the type of X-axis sweep that is performed on a channel.

TestPortPower ICalSet3 Returns the RF power level for the channel.

TestSetType ICalSet5 Returns the Test Set type used for the Cal Set.

**ICalSet History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalSet</td>
<td>2.0</td>
</tr>
<tr>
<td>ICalSet2</td>
<td>3.0</td>
</tr>
<tr>
<td>ICalSet3</td>
<td>3.2</td>
</tr>
<tr>
<td>ICalSet4</td>
<td>6.0</td>
</tr>
<tr>
<td>ICalSet5</td>
<td>6.2</td>
</tr>
<tr>
<td>ICalSet6</td>
<td>9.30</td>
</tr>
<tr>
<td>ICalSet7</td>
<td>9.40</td>
</tr>
<tr>
<td>ICalSet8</td>
<td>10.15</td>
</tr>
</tbody>
</table>

**ICalData Interface**

**Description**

Use this interface as an alternative to the ICalSet Interface to avoid using variants when transmitting data to and from the Cal Set.

Learn about reading and writing Calibration data.
<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>get ErrorTermComplex</td>
<td>ICalData2</td>
<td>See History Superseded with getErrorTermComplexByString</td>
</tr>
<tr>
<td>getErrorTermComplexByString</td>
<td>ICalData3</td>
<td>Returns typed error term data by specifying the string name of the error term.</td>
</tr>
<tr>
<td>getStandardComplex</td>
<td>ICalData2</td>
<td>Superseded with getStandardComplexByString</td>
</tr>
<tr>
<td>getStandardComplexByString</td>
<td>ICalData3</td>
<td>Returns typed standard acquisition data by specifying the string name of the standard.</td>
</tr>
<tr>
<td>putErrorTermComplex</td>
<td>ICalData2</td>
<td>Superseded with putErrorTermComplexByString</td>
</tr>
<tr>
<td>putErrorTermComplexByString</td>
<td>ICalData3</td>
<td>Writes typed error term data by specifying the string name of the error term.</td>
</tr>
<tr>
<td>putStandardComplex</td>
<td>ICalData2</td>
<td>Superseded with putStandardComplexByString</td>
</tr>
<tr>
<td>putStandardComplexByString</td>
<td>ICalData3</td>
<td>Writes typed standard acquisition data by specifying the string name of the standard.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The original ICalData Interface was introduced with VNA 1.0 on the Calibrator Object.</td>
</tr>
<tr>
<td>ICalData2</td>
<td>2.0</td>
</tr>
<tr>
<td>ICalData3</td>
<td>3.1</td>
</tr>
</tbody>
</table>
CalSets Collection

Description

A collection object that provides a mechanism for iterating through all the Cal Sets in the analyzer. There is no ordering to the items in the collection. Therefore make no assumptions about the formatting of the collection.

For the Item and Remove methods, you can specify either the Cal Set string name, or the integer item of the Cal Set in the collection.

Accessing the CalSets collection

Get a handle to the CalSets collection through the CalManager object with the app.GetCalManager Method.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim calsts As CalSets
Set calsts = app.GetCalManager.CalSets
```

See Also:

- CalSet Object
- Collections in the Analyzer
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exists</td>
<td>ICalSets2</td>
<td>Returns whether the specified Cal Set exists</td>
</tr>
<tr>
<td>Item</td>
<td>ICalSets</td>
<td>Returns a handle to a CalSet object in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>ICalSets</td>
<td>Deletes the Cal Set residing at position index in the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of Cal Sets in the collection.</td>
</tr>
</tbody>
</table>
### CalSets History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalSets</td>
<td>1.0</td>
</tr>
<tr>
<td>ICalSets2</td>
<td>9.33</td>
</tr>
</tbody>
</table>
## CalStandard Object

### Description

Contains all of the settings that are required to modify a calibration standard.

For more information, read *Specifying Calibration Standards and Kits for Keysight Vector Network Analyzers (Application Note 1287-11)*

### Accessing the CalStandard object

Get a handle to a standard with the `calkit.GetCalStandard` Method.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim std As ICalStandard
Set std = app.ActiveCalKit.GetCalStandard(1)
std.Delay = 0.00000003
```

### See Also:

- [VNA Automation Interfaces](#)
- [The VNA Object Model](#)
- [Reading and Writing Calibration data](#)
- [Example Programs](#)

### Methods

None

### Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C0</td>
<td>ICalStandard</td>
<td>Sets and Returns the C0 (C-zero) value (the first capacitance value) for the calibration standard, when the Type is set to &quot;naOpen&quot;.</td>
</tr>
<tr>
<td>C1</td>
<td>ICalStandard</td>
<td>Sets and Returns the C1 value (the second capacitance value) for the calibration standard, when the Type is set to &quot;naOpen&quot;.</td>
</tr>
<tr>
<td>C2</td>
<td>ICalStandard</td>
<td>Sets and Returns the C2 value (the third capacitance value) for the calibration standard, when the Type is set to &quot;naOpen&quot;.</td>
</tr>
</tbody>
</table>
C3  ICalStandard  Sets and Returns the C3 value (the fourth capacitance value) for the calibration standard, when the Type is set to "naOpen".

Delay  ICalStandard  Sets and Returns the electrical delay value for the calibration standard.

L0  ICalStandard  Sets and Returns the L0 (L-zero) value (the first inductance value) for the calibration standard, when the Type is set to "naShort".

L1  ICalStandard  Sets and Returns the L1 value (the second inductance value) for the calibration standard, when the Type is set to "naShort".

L2  ICalStandard  Sets and Returns the L2 value (the third inductance value) for the calibration standard, when the Type is set to "naShort".

L3  ICalStandard  Sets and Returns the L3 value (the third inductance value) for the calibration standard, when the Type is set to "naShort".

Label  ICalStandard  Sets and Returns the label for the calibration standard.

Loss  ICalStandard  Sets and Returns the insertion loss for the calibration standard.

Maximum Frequency  ICalStandard  Sets and Returns the maximum frequency for the calibration standard.

Medium  ICalStandard  Sets and Returns the media type of the calibration standard.

Minimum Frequency  ICalStandard  Sets and Returns the minimum frequency for the calibration standard.

Type  ICalStandard  Sets and Returns the type of calibration standard. Selections are: naOpen, naShort, naLoad, naThru, naArbitraryImpedance and naSliding.

TZReal  ICalStandard  Sets and Returns the TZReal value (the Real Terminal Impedance value) for the calibration standard, when the Type is set to "naArbitraryImpedance".

TZImag  ICalStandard  Sets and Returns the TZImag value (the Imaginary Terminal Impedance value) for the calibration standard, when the Type is set to "naArbitraryImpedance".

Z0  ICalStandard  Sets and Returns the characteristic impedance for the calibration standard.

ICalStandard History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalStandard</td>
<td>1.0</td>
</tr>
<tr>
<td>CalStandard2</td>
<td>3.0</td>
</tr>
</tbody>
</table>
Capabilities Object

Description

These properties return capabilities of the remote VNA.

Accessing the Capabilities object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim cap As Capabilities
Set cap = app.Capabilities
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- ICapabilities History
- Example Programs

### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetPortNumber Method</td>
<td>ICapabilities4</td>
<td>Returns the port number for the specified string port name.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AvailableMeasurementClasses</td>
<td>ICapabilities7</td>
<td>Returns the measurement classes on the VNA</td>
</tr>
<tr>
<td>CpuRevision</td>
<td>ICapabilities6</td>
<td>Returns the CPU speed of the VNA</td>
</tr>
<tr>
<td>DirectoryPath</td>
<td>ICapabilities10</td>
<td>Returns the directory location.</td>
</tr>
<tr>
<td>DspRevision</td>
<td>ICapabilities6</td>
<td>Returns the DSP Revision number</td>
</tr>
<tr>
<td>DspFpgaRevision</td>
<td>ICapabilities6</td>
<td>Returns the DSP FPGA Revision number</td>
</tr>
<tr>
<td>FirmwareMajorRevision</td>
<td>ICapabilities</td>
<td>Returns integer portion of firmware revision number.</td>
</tr>
</tbody>
</table>
FirmwareMinorRevision | ICapabilities | Return decimal portion of firmware revision number.
FirmwareSeries | ICapabilities | Returns the Alpha portion of the firmware revision number.
GPIBPortCount | ICapabilities | Returns the number of GPIB ports (1 or 2).
HasDirectReceiverAccess | ICapabilities | Returns whether or not the analyzer has direct receiver access (front-panel jumpers).
HasLowFrequencyExtension | ICapabilities | Returns whether or not the VNA has the low frequency extension (LFE) installed.
IFBWList | ICapabilities | Returns a list of supported IFBW values.
InternalDCReceiverCount | ICapabilities | Returns the number of internal DC receivers in the analyzer.
InternalDCReceiverNames | ICapabilities | Returns a list of names of the internal DC receivers.
InternalDCSourceCount | ICapabilities | Returns the number of internal DC sources in the analyzer.
InternalDCSourceNames | ICapabilities | Returns a list of names of the internal DC sources.
InternalSourcePortCount | ICapabilities | Returns the number of internal source ports.
InternalSourcePortNames | ICapabilities | Returns a list of internal source port names.
InternalTestsetPortCount | ICapabilities | Returns the number of VNA test ports.
InternalTestsetPortNames | ICapabilities | Returns a list of internal test port names.
IsFrequencyOffsetPresent | ICapabilities | Returns the presence of Frequency Offset Option S93080A/B (True or False).
IsReceiverStepAttenuatorPresent | ICapabilities | Returns the presence of receiver step attenuators (True or False).
IsReferenceBypassSwitchPresent | ICapabilities | Returns the presence of the reference switch (True or False).
MaximumFrequency | ICapabilities | Returns the maximum frequency of the VNA.
MaximumNumberOfChannels | ICapabilities | Returns the maximum possible number of Channels.
<table>
<thead>
<tr>
<th>Method</th>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaximumNumberOfPoints</td>
<td>ICapabilities</td>
<td>Returns the maximum possible number of data points.</td>
</tr>
<tr>
<td>MaximumNumberOfTracesPerWindow</td>
<td>ICapabilities2</td>
<td>Returns the maximum possible number of traces per window.</td>
</tr>
<tr>
<td>MaximumNumberOfWindows</td>
<td>ICapabilities2</td>
<td>Returns the maximum possible number of windows.</td>
</tr>
<tr>
<td>MaximumReceiverStepAttenuator</td>
<td>ICapabilities</td>
<td>Returns the maximum amount of receiver attenuation.</td>
</tr>
<tr>
<td>MaximumSourceALCPower</td>
<td>ICapabilities</td>
<td>Returns the maximum amount of source ALC power.</td>
</tr>
<tr>
<td>MaximumSourceStepAttenuator</td>
<td>ICapabilities</td>
<td>Returns the maximum amount of source attenuation.</td>
</tr>
<tr>
<td>MeasurementClassProperties</td>
<td>ICapabilities8</td>
<td>Returns a handle to the MeasurementClassProperties Object.</td>
</tr>
<tr>
<td>MinimumFrequency</td>
<td>ICapabilities</td>
<td>Returns the minimum frequency of the VNA.</td>
</tr>
<tr>
<td>MinimumNumberOfPoints</td>
<td>ICapabilities</td>
<td>Returns the minimum possible number of data points.</td>
</tr>
<tr>
<td>MinimumReceiverStepAttenuator</td>
<td>ICapabilities</td>
<td>Returns the minimum amount of receiver attenuation.</td>
</tr>
<tr>
<td>MinimumSourceALCPower</td>
<td>ICapabilities</td>
<td>Returns the minimum amount of source ALC power.</td>
</tr>
<tr>
<td>ModelNumber</td>
<td>ICapabilities11</td>
<td>Returns the model number of the VNA.</td>
</tr>
<tr>
<td>NoiseReceiverNoiseBWList</td>
<td>ICapabilities13</td>
<td>Returns the list of supported Noise Bandwidths values when using a noise receiver (option 029).</td>
</tr>
<tr>
<td>PowerRange</td>
<td>ICapabilities15</td>
<td>Returns the PowerRange object.</td>
</tr>
<tr>
<td>PresetMaxFrequency</td>
<td>ICapabilities13</td>
<td>Returns the maximum specified frequency of the analyzer. Does not include any oversweep.</td>
</tr>
<tr>
<td>PresetMinFrequency</td>
<td>ICapabilities13</td>
<td>Returns the minimum specified frequency of the analyzer. Does not include any oversweep.</td>
</tr>
<tr>
<td>ReceiverCount</td>
<td>ICapabilities</td>
<td>Returns the number of receivers in the VNA.</td>
</tr>
</tbody>
</table>
ReceiverStepAttenuatorStepSize | ICapabilities | Returns the step size of the attenuator.
ReceiverTemperature Property | ICapabilities9 | Returns the temperature on the receiver board.
ResBWList | ICapabilities8 | Returns a list of Res BW values that are supported by IM Spectrum.
ResBWListSA | ICapabilities17 | Returns the list of supported Resolution BW values for the SA channel.
SerialNumber | ICapabilities11 | Returns the serial number of the VNA.
SourceCount | ICapabilities | Returns the number of sources.
SourcePortCount | ICapabilities4 | Returns the number of source ports.
SourcePortNames | ICapabilities4 | Returns the string names of source ports.
SourceStepAttenuatorStepSize | ICapabilities5 | Returns a value indicating the step size of the source attenuator.
StandardReceiverNoiseBWList | ICapabilities13 | Returns the list of supported Noise Bandwidths values when using the NA receiver for noise measurements (option 028).
TestPortNames | ICapabilities13 | Returns a list of test port names including external test set ports.

ICapabilities History

<table>
<thead>
<tr>
<th>I Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICapabilities</td>
<td>3.5</td>
</tr>
<tr>
<td>ICapabilities2</td>
<td>5.23</td>
</tr>
<tr>
<td>ICapabilities3</td>
<td>6.0</td>
</tr>
<tr>
<td>ICapabilities4</td>
<td>7.20</td>
</tr>
<tr>
<td>ICapabilities5</td>
<td>8.04</td>
</tr>
<tr>
<td>ICapabilities6</td>
<td>9.10</td>
</tr>
<tr>
<td>ICapabilities7</td>
<td>9.33</td>
</tr>
<tr>
<td>ICapabilities8</td>
<td>9.40</td>
</tr>
<tr>
<td>ICapabilities9</td>
<td>9.50</td>
</tr>
<tr>
<td>ICapabilities10</td>
<td>10.0</td>
</tr>
<tr>
<td>ICapabilities11</td>
<td>10.0</td>
</tr>
<tr>
<td>ICapabilities12</td>
<td>10.40</td>
</tr>
<tr>
<td>ICapabilities13</td>
<td>10.40</td>
</tr>
<tr>
<td>ICapabilities14</td>
<td>10.45</td>
</tr>
<tr>
<td>ICapabilities15</td>
<td>12.70</td>
</tr>
<tr>
<td>ICapabilities16</td>
<td>12.80</td>
</tr>
<tr>
<td>ICapabilities17</td>
<td>13.25</td>
</tr>
</tbody>
</table>
Channel Object

See SourcePowerCalData Interface for putting and getting typed source power calibration data.

Description

The channel object is like the engine that produces data. Channel settings consist of stimulus values like frequency, power, IF bandwidth, and number of points.

Accessing the Channel object

You can get a handle to a channel in a number of ways. But first you have to make sure that the channel exists. When you first startup the analyzer, there is one S11 measurement on channel 1. Thus there is only one channel in existence. You can do the following:

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim chan As IChannel
Set chan = app.ActiveChannel
```

or

```
Set chan = app.Channels(2)
```

The first method returns the channel object that is driving the active measurement. If there is no measurement, there may not be a channel. Once a channel is created, it does not go away. So if there once was a measurement (hence a channel), the channel will still be available.

If there is no channel you can create one in a couple ways. You can do the following:

```
Pna.CreateMeasurement( ch1, "S11", port1, window2)
```

or

```
Pna.Channels.Add(2)
```

The latter will have no visible effect on the analyzer. It will simply create channel 2 if it does not already exist.

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
**Superseded commands**

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td>IChannel</td>
<td>Aborts the current measurement sweep on the channel.</td>
</tr>
<tr>
<td>ApplySourcePowerCorrectionTo</td>
<td>IChannel11</td>
<td>Copies an existing Source Power Calibration to another channel.</td>
</tr>
<tr>
<td>AveragingRestart</td>
<td>IChannel</td>
<td>Clears and restarts averaging of the measurement data.</td>
</tr>
<tr>
<td>Continuous</td>
<td>IChannel</td>
<td>The channel continuously responds to trigger signals.</td>
</tr>
<tr>
<td>CopyToChannel</td>
<td>IChannel2</td>
<td>Sets up another channel as a copy of this object's channel.</td>
</tr>
<tr>
<td>FrequencySpanFull</td>
<td>IChannel25</td>
<td>Sets the frequency span to the entire frequency range of the analyzer.</td>
</tr>
<tr>
<td><strong>GetConverter</strong></td>
<td>IChannel17</td>
<td>Returns a handle to a Converter object.</td>
</tr>
<tr>
<td>GetErrorCorrection</td>
<td>IChannel8</td>
<td>Returns the channel error correction state.</td>
</tr>
<tr>
<td>GetNumberOfGroups</td>
<td>IChannel4</td>
<td>Returns the number of groups a channel has yet to acquire.</td>
</tr>
<tr>
<td>GetPortNumber</td>
<td>IChannel13</td>
<td>Returns the port number for the specified string port name.</td>
</tr>
<tr>
<td><strong>GetRxLevelingConfiguration</strong></td>
<td>IChannel17</td>
<td>Returns a handle to a RxLevelingConfiguration object.</td>
</tr>
<tr>
<td>getSourcePowerCalData</td>
<td>IChannel</td>
<td><strong>Superseded with</strong> GetSourcePowerCalDataEx</td>
</tr>
<tr>
<td>getSourcePowerCalDataEx</td>
<td>IChannel4</td>
<td>Returns requested source power calibration data, if it exists.</td>
</tr>
<tr>
<td>GetSupportedALCModes</td>
<td>IChannel10</td>
<td>Returns a list of supported ALC modes</td>
</tr>
<tr>
<td>GetXAxisValues</td>
<td>IChannel</td>
<td>Returns the channel's X-axis values into a dimensioned Variant array.</td>
</tr>
<tr>
<td>GetXAxisValues2</td>
<td>IChannel</td>
<td>Returns the channel's X-axis values into a dimensioned NON-Variant array.</td>
</tr>
<tr>
<td>Hold</td>
<td>IChannel</td>
<td>Puts the Channel in Hold - not sweeping.</td>
</tr>
</tbody>
</table>
Next_IFBandwidth | IChannel | A function that returns the Next higher IF Bandwidth value.
--- | --- | ---
NumberOfGroups | IChannel | Sets the Number of trigger signals the channel will receive.
Preset | IChannel | Resets the channel to factory defined settings.
PreviousIFBandwidth | IChannel | Returns the previous IF Bandwidth value.
putSourcePowerCalData | IChannel | Superseded with Put SourcePowerCalDataEx Method
putSourcePowerCalDataEx | IChannel4 | Inputs source power calibration data to this channel for a specific source port.
SelectCalSet | IChannel | Specifies the Cal Set to use for the Channel
Single | IChannel | Channel responds to one trigger signal from any source (internal, external, or manual). Then channel switches to Hold.
UnselectCalset | IChannel24 | Unselects a Cal Set from the specified channel.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALCLevelingMode</td>
<td>IChannel10</td>
<td>Set or return the ALC leveling mode.</td>
</tr>
<tr>
<td>AlternateSweep</td>
<td>IChannel</td>
<td>Sets sweeps to either alternate or chopped.</td>
</tr>
<tr>
<td>Attenuator</td>
<td>IChannel</td>
<td>Sets or returns the value of the attenuator control for the specified port number.</td>
</tr>
<tr>
<td>AttenuatorMode</td>
<td>IChannel</td>
<td>Sets or returns the mode of operation of the attenuator control for the specified port number.</td>
</tr>
<tr>
<td>AuxiliaryTrigger</td>
<td>IChannel10</td>
<td>Provides access to Auxiliary Triggering</td>
</tr>
<tr>
<td>Averaging</td>
<td>IChannel</td>
<td>Turns trace averaging ON or OFF for all measurements on the channel.</td>
</tr>
<tr>
<td>AveragingCount</td>
<td>IChannel</td>
<td>Returns the number of sweeps that have been averaged into the measurements.</td>
</tr>
<tr>
<td>AveragingFactor</td>
<td>IChannel</td>
<td>Specifies the number of measurement sweeps to combine for an average.</td>
</tr>
<tr>
<td>AverageMode</td>
<td>IChannel16</td>
<td>Sets Point or Sweep averaging.</td>
</tr>
<tr>
<td>BalancedTopology</td>
<td>IChannel6</td>
<td>Provides access to the topology of a balanced DUT.</td>
</tr>
<tr>
<td>Calibrator</td>
<td>IChannel4</td>
<td>Provides access to Unguided calibration.</td>
</tr>
<tr>
<td>CalSet</td>
<td>IChannel7</td>
<td>Provides access to the contents of a Cal Set</td>
</tr>
</tbody>
</table>
centerFrequency  IChannel  Sets or returns the center frequency of the channel.
               Shared with the Segment Object

CenterFrequencyStepSize  IChannel25 Sets and reads the center frequency step size of the analyzer.

CenterFrequencyStepSizeMode  IChannel25 Sets and reads how the center frequency step size is determined.

channelNumber  IChannel  Returns the Channel number.
               Shared with the Measurement Object

Converter  IChannel21 Provides access to a mixer/converter object.

CorrectionMethods  IChannel21 Provides access to channel correction properties.

CoupleChannelParams  IChannel5  Turns ON and OFF Time Domain Trace Coupling.

CouplePorts  IChannel  Turns ON and OFF port power coupling.

CustomChannelConfiguration  IChannel12 Provides access to custom application objects.

CWFrequency  IChannel  Set the Continuous Wave (CW) frequency.

DCStimulus  IChannel23 Provides access to the DCStimulus object.

DefinedRoles  IChannel22 Returns the roles for which sources can be used for the channel.

DwellTime  IChannel  Sets or returns the dwell time for the channel.
               Shared with the Segment Object

ErrorCorrection  IChannel7  Attempts to sets error correction ON or OFF for all of the measurements on the channel.

ExternalTriggerDelay  IChannel6 Sets or returns the external trigger delay value for the channel.

FastCWPointCount  IChannel16 Enables Fast CW sweep and sets the number of data points for the channel.

Fixturing  IChannel6 Provides access to Port Ext, Embedding, and De-embedding functions.

FOM Collection  IChannel9 Provides access to Frequency Offset Measurements

FrequencyOffsetDivisor  IChannel2
FrequencyOffsetFrequency  IChannel2
FrequencyOffsetMultiplier  IChannel2  Superseded with FOM and FOMRange
<table>
<thead>
<tr>
<th>Property</th>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FrequencyOffsetCWOVERRIDE</td>
<td>IChannel2</td>
<td>Sets or returns the frequency offset of the channel.</td>
</tr>
<tr>
<td>FrequencyOffsetState</td>
<td>IChannel2</td>
<td>Sets or returns the frequency offset state of the channel.</td>
</tr>
<tr>
<td>FrequencySpan</td>
<td>IChannel</td>
<td>Sets or returns the frequency span of the channel.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Segment Object.</td>
</tr>
<tr>
<td>FrequencyStep</td>
<td>IChannel24</td>
<td>Sets the frequency step size across the selected frequency range.</td>
</tr>
<tr>
<td>IFBandwidth</td>
<td>IChannel</td>
<td>Sets or returns the IF Bandwidth of the channel.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Segment Object.</td>
</tr>
<tr>
<td>IFConfiguration</td>
<td>IChannel4</td>
<td>Provides access to the IF gain and source path settings for the H11 Option.</td>
</tr>
<tr>
<td>IsBlocked</td>
<td>IChannel19</td>
<td>Returns whether or not a channel is blocked from sweeping.</td>
</tr>
<tr>
<td>IsContinuous</td>
<td>IChannel4</td>
<td>Returns whether or not a channel is in continuous mode.</td>
</tr>
<tr>
<td>IsHold</td>
<td>IChannel4</td>
<td>Returns whether or not a channel is in hold mode.</td>
</tr>
<tr>
<td>LowFrequencyExtension</td>
<td>IChannel26</td>
<td>Enables low frequency extension (LFE).</td>
</tr>
<tr>
<td>MeasurementClass</td>
<td>IChannel15</td>
<td>Returns the measurement class name.</td>
</tr>
<tr>
<td>MultiDimensionalSweep</td>
<td>IChannel27</td>
<td>Provides access to the MultiDimensionalSweep Object.</td>
</tr>
<tr>
<td>NumberOfPoints</td>
<td>IChannel</td>
<td>Sets or returns the Number of Points of the channel.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Segment Object.</td>
</tr>
<tr>
<td>Parent</td>
<td>IChannel</td>
<td>Returns a handle to the parent object of the channel.</td>
</tr>
<tr>
<td>PathConfiguration</td>
<td>IChannel10</td>
<td>Provides access to path configuration switches and setting.</td>
</tr>
<tr>
<td>PathConfigurationManager</td>
<td>IChannel10</td>
<td>Provides access to path configuration file management.</td>
</tr>
<tr>
<td>PhaseControl</td>
<td>IChannel21</td>
<td>Provides access to the PhaseControl object.</td>
</tr>
<tr>
<td>PointSweepState</td>
<td>IChannel16</td>
<td>Turns point sweep ON or OFF for all measurements on the channel.</td>
</tr>
<tr>
<td>PowerSlope</td>
<td>IChannel</td>
<td>Sets or returns the Power Slope value.</td>
</tr>
<tr>
<td>PowerSlopeState</td>
<td>IChannel18</td>
<td>Turns power slope ON or OFF</td>
</tr>
<tr>
<td>PulseGenerator</td>
<td>IChannel10</td>
<td>Provides access to pulse generator configuration.</td>
</tr>
</tbody>
</table>
PulseGeneratorID
IChannel23  Returns the ID for the specified Pulse Generator name.

PulseGeneratorNames
IChannel23  Returns a list of configured Pulse Generator names.

PulseMeasControl
IChannel20  Provides access to pulse measurement settings.

R1InputPath
IChannel2  Throws internal reference switch.

ReceiverAttenuator
IChannel  Sets or returns the value of the specified receiver attenuator control.

ReduceIFBandwidth
IChannel5  Sets or returns the state of the Reduced IF Bandwidth at Low Frequencies setting.

RoleDevice
IChannel22  Sets and returns the source to be used in the specified role.

RXLevelingConfiguration
IChannel21  Provides access to the ReceiverLeveling Object

Segments
IChannel  Provides access to the Collection for iterating through the sweep segments of a channel.

SignalProcessingModuleFour
IChannel10  Provides access to the SignalProcessingModuleFour object.

SourcePortCount
IChannel13  Returns the number of source ports.

SourcePortFixedFrequency
IChannel27  Set and read the fixed frequency value for a specific port.

SourcePortMode
IChannel9  Sets the state of the VNA sources. (AUTO | ON | OFF)

SourcePortNames
IChannel13  Returns the string names of source ports.

SourcePortStartFrequency
IChannel27  Set and read the start frequency value for a specific port.

SourcePortStopFrequency
IChannel27  Set and read the stop frequency value for a specific port.

SourcePowerCalPowerOffset
IChannel4  Sets or returns a power level offset from the VNA test port power.

SourcePowerCorrection
IChannel  Turns source power correction ON or OFF for a specific source port.

StartFrequency
IChannel  Sets or returns the start frequency of the channel.

Shared with the Segment Object

StartPower
IChannel  Sets the start power of the analyzer when sweep type is set to Power Sweep.
<table>
<thead>
<tr>
<th>Method</th>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StartPowerEx</td>
<td>IChannel13</td>
<td>Sets and reads the power sweep start power value for a specific port.</td>
</tr>
<tr>
<td>StopFrequency</td>
<td>IChannel</td>
<td>Sets or returns the stop frequency of the channel.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Shared with the Segment Object</strong></td>
</tr>
<tr>
<td>StopPower</td>
<td>IChannel</td>
<td>Sets the Stop Power of the analyzer when sweep type is set to Power Sweep.</td>
</tr>
<tr>
<td>StopPowerEx</td>
<td>IChannel13</td>
<td>Sets and reads the power sweep stop power value for a specific port.</td>
</tr>
<tr>
<td>SweepDelay</td>
<td>IChannel19</td>
<td>Sets the time to wait just before acquisition begins for each sweep.</td>
</tr>
<tr>
<td>SweepGenerationMode</td>
<td>IChannel</td>
<td>Sets the method used to generate a sweep: continuous ramp (analog) or discrete steps (stepped).</td>
</tr>
<tr>
<td>SweepSpeedMode</td>
<td>IChannel14</td>
<td>Set or returns the sweep speed mode.</td>
</tr>
<tr>
<td>SweepTime</td>
<td>IChannel</td>
<td>Sets the Sweep time of the analyzer.</td>
</tr>
<tr>
<td>SweepType</td>
<td>IChannel</td>
<td>Sets the type of X-axis sweep that is performed on a channel.</td>
</tr>
<tr>
<td>TestPortPower</td>
<td>IChannel</td>
<td>Sets or returns the RF power level for the channel.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Shared with the Segment Object</strong></td>
</tr>
<tr>
<td>TriggerMode</td>
<td>IChannel</td>
<td>Determines the measurement that occurs when a trigger signal is sent to the channel.</td>
</tr>
<tr>
<td>UserRangeMax</td>
<td>IChannel</td>
<td><strong>Superseded</strong> - Use meas.UserRangeMax</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sets the stimulus stop value for the specified User Range.</td>
</tr>
<tr>
<td>UserRangeMin</td>
<td>IChannel</td>
<td><strong>Superseded</strong> - Use meas.UserRangeMin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sets the stimulus start value for the specified User Range.</td>
</tr>
<tr>
<td>XAxisPointSpacing</td>
<td>IChannel</td>
<td>Sets X-Axis point spacing for the active channel.</td>
</tr>
</tbody>
</table>

**ICHannel History**
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IChannel</td>
<td>1.0</td>
</tr>
<tr>
<td>IChannel2</td>
<td>3.0</td>
</tr>
<tr>
<td>IChannel3</td>
<td>4.0</td>
</tr>
<tr>
<td>IChannel4</td>
<td>4.0</td>
</tr>
<tr>
<td>IChannel5</td>
<td>4.2</td>
</tr>
<tr>
<td>IChannel6</td>
<td>5.0</td>
</tr>
<tr>
<td>IChannel7</td>
<td>5.2</td>
</tr>
<tr>
<td>IChannel8</td>
<td>6.0</td>
</tr>
<tr>
<td>IChannel9</td>
<td>7.0</td>
</tr>
<tr>
<td>IChannel10</td>
<td>7.2</td>
</tr>
<tr>
<td>IChannel11</td>
<td>7.5</td>
</tr>
<tr>
<td>IChannel12</td>
<td>8.0</td>
</tr>
<tr>
<td>IChannel13</td>
<td>8.2</td>
</tr>
<tr>
<td>IChannel14</td>
<td>8.33</td>
</tr>
<tr>
<td>IChannel15</td>
<td>8.33</td>
</tr>
<tr>
<td>IChannel16</td>
<td>8.35</td>
</tr>
<tr>
<td>IChannel17</td>
<td>8.55</td>
</tr>
<tr>
<td>IChannel19</td>
<td>9.20</td>
</tr>
<tr>
<td>IChannel20</td>
<td>9.20</td>
</tr>
<tr>
<td>IChannel21</td>
<td>9.30</td>
</tr>
<tr>
<td>IChannel22</td>
<td>9.42</td>
</tr>
<tr>
<td>IChannel23</td>
<td>9.50</td>
</tr>
<tr>
<td>IChannel24</td>
<td>10.40</td>
</tr>
<tr>
<td>IChannel25</td>
<td>10.45</td>
</tr>
<tr>
<td>IChannel26</td>
<td>12.80</td>
</tr>
<tr>
<td>IChannel27</td>
<td>13.25</td>
</tr>
</tbody>
</table>

**ISourcePowerCalData Interface**

**Description**

Contains methods for putting source power calibration data in and getting source power calibration data out of the analyzer using typed data. The methods in this interface transfer data more efficiently than methods that use variant data. However, this interface is only usable from VB6, C, & C++. All other programming languages must use the methods on the Channel Object.
**Note:** Interface *ISourcePowerCalData* is abbreviated as *ISPCD* in the following table.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getSourcePowerCalDataScalar</td>
<td>ISPCD</td>
<td>Superseded - use <em>PutSourcePowerCalDataScalarEx</em> Method</td>
</tr>
<tr>
<td>getSourcePowerCalDataScalarEx</td>
<td>ISPCD2</td>
<td>Returns requested source power calibration data, if it exists.</td>
</tr>
<tr>
<td>putSourcePowerCalDataScalar</td>
<td>ISPCD</td>
<td>Superseded - use <em>PutSourcePowerCalDataEx</em> Method</td>
</tr>
<tr>
<td>putSourcePowerCalDataScalarEx</td>
<td>ISPCD2</td>
<td>Inputs source power calibration data to a channel, for a specific source port.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**ISourcePowerCalData History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISourcePowerCalData</td>
<td>2.0</td>
</tr>
<tr>
<td>ISourcePowerCalData2</td>
<td>4.0</td>
</tr>
</tbody>
</table>
Channels Collection

Description

A collection object that provides a mechanism for iterating through the channels.

Collections are, by definition, unordered lists of like objects. You cannot assume that Channels.Item(1) is always Channel 1.

Accessing the Channels collection

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim chans As Channels
Set chans = app.Channels
```

See Also:

- Channel Object
- Collections in the Analyzer
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>IChannels</td>
<td>An alternate way to create a measurement.</td>
</tr>
<tr>
<td>Hold</td>
<td>IChannels</td>
<td>Places all channels in Hold trigger mode.</td>
</tr>
<tr>
<td>Item</td>
<td>IChannels</td>
<td>Use to get a handle to a channel in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>IChannels3</td>
<td>Delete a channel by specifying the index in the collection.</td>
</tr>
<tr>
<td>RemoveChannelNumber</td>
<td>IChannels3</td>
<td>Delete a channel by specifying the channel number.</td>
</tr>
<tr>
<td>Resume</td>
<td>IChannels2</td>
<td>Resumes the trigger mode of all channels that was in effect before sending the channels.Hold method.</td>
</tr>
<tr>
<td>Properties</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Count</td>
<td>Returns the number of channels in the analyzer.</td>
<td></td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the current Application.</td>
<td></td>
</tr>
<tr>
<td>UnusedChannelNumbers</td>
<td>IChannels2 Returns an array of channel numbers that are NOT in use.</td>
<td></td>
</tr>
<tr>
<td>UsedChannelNumbers</td>
<td>IChannels2 Returns an array of channel numbers that are in use.</td>
<td></td>
</tr>
</tbody>
</table>

**IChannels History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IChannels</td>
<td>1.0</td>
</tr>
<tr>
<td>IChannels2</td>
<td>9.30</td>
</tr>
<tr>
<td>IChannels3</td>
<td></td>
</tr>
</tbody>
</table>
### ComColors Object

**Description**

Provides access to the methods and properties used to modify the VNA Display and Print colors.

**Accessing the ComColors object**

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835X.Application")
Set displayColors = app.Preferences.DisplayColors
'or
'Set printColors = app.Preferences.PrintColors
displayColors.ActiveLabels = 657930
```

**See Also:**

- ComTraceColors Object
- Modify Display Colors Example
- About VNA Display Colors
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoadTheme</td>
<td>IColors</td>
<td>Load a color theme from a disc file.</td>
</tr>
<tr>
<td>ResetTheme</td>
<td>IColors</td>
<td>Resets the current theme to the default VNA colors.</td>
</tr>
<tr>
<td>StoreTheme</td>
<td>IColors</td>
<td>Saves the current color theme to a disc file.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveLabels</td>
<td>IColors Sets labels and grid frame colors in the active window.</td>
</tr>
<tr>
<td>ActiveBackground</td>
<td>IColors2 Set and return the background color for the active window on the VNA display or hardcopy print.</td>
</tr>
</tbody>
</table>
**Background**  
IColors  
Set and return the background color for the inactive windows on the VNA display or hardcopy print.

**FailedTraces**  
IColors  
Set and return the limit line color of failed traces.

**Grid**  
IColors  
Set and return the inner lines of all grid in all windows.

**InactiveLabels**  
IColors  
Set and return the Inactive (not selected) Window Labels.

**Trace**  
IColors  
Provides access to the ComTraceColors Object for setting colors for the first 8 traces

**IColors History**

<table>
<thead>
<tr>
<th>I Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IColors</td>
<td>9.0</td>
</tr>
<tr>
<td>IColors2</td>
<td>9.2</td>
</tr>
</tbody>
</table>
ComTraceColors Object

Description

Provides access to the methods and properties used to modify the VNA Display and Print colors.

Both the Display and Print ComColor objects contain 8 Trace objects (1 to 8).

'1st Trace' is NOT always Tracel (Tr1). For example, the first trace in a window might be Tr2 which is drawn with the "1st Trace" pen.

The first 8 traces are drawn with the defined pen colors. The next eight traces reuse the same colors, and so forth. For example, if all traces are numbered sequentially, the 9th and 17th traces are drawn using the same color as the 1st trace.

Accessing the ComTraceColors object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835X.Application")
Set displayColors = app.Preferences.DisplayColors
' or
'Set printColors = app.Preferences.PrintColors
dim Trace1
Set Trace1 = displayColors.Trace(1)
Trace1.DataAndLimits = RGB(1,251,1)
```

See Also:

- Modify Display Colors Example
- About VNA Display Colors
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
### Methods

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>See History</td>
<td>See History</td>
</tr>
</tbody>
</table>

None

### Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataAndLimits</td>
<td>ITraceColors</td>
<td>Set and return the color of Data and Limit Lines for nth trace in a window.</td>
</tr>
<tr>
<td>Markers</td>
<td>ITraceColors</td>
<td>Set and return the color of data trace markers for nth trace in a window.</td>
</tr>
<tr>
<td>Memory</td>
<td>ITraceColors</td>
<td>Set and return the memory trace color for nth trace in a window.</td>
</tr>
<tr>
<td>MemoryMarkers</td>
<td>ITraceColors</td>
<td>Set and return the color of memory trace markers for nth trace in a window.</td>
</tr>
</tbody>
</table>

### ITraceColors History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITraceColors</td>
<td>9.0</td>
</tr>
</tbody>
</table>
**Converter Object**

**Note:** The Converter Object replaces the IMixer Interface.

**Description**

Contains the methods and properties to setup a mixer for ALL VNA Mixer/Converter applications.

**Accessing the Converter Interface**

```vba
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim converter as Converter
Set converter = chan.Converter
```

**Scratch vs Applied Mixer Properties**

Each mixer configuration has two sets of properties:

1. **Scratch mixer** contains the properties that have been set, but NOT YET applied. Send the Apply Method to copy these properties to the Applied mixer.

2. **Applied mixer** contains the properties that makeup the current mixer configuration.

Power settings are immediately applied to both the Scratch and Applied mixer.

A successful Calculate also performs an Apply.

**Note:** Queries always return the Applied mixer properties. Therefore, first send Apply Method before querying new settings.

**See Also**

- VNA Automation Interfaces
- The VNA Object Model
- Mixer Setup UI topic
<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddSegment</td>
<td>Converter5</td>
<td>Add segments to the segment table (FCA Only)</td>
</tr>
<tr>
<td>Apply</td>
<td>Converter</td>
<td>Applies mixer settings.</td>
</tr>
<tr>
<td>AssignSourceToRole</td>
<td>Converter</td>
<td>Assigns a configured source to the specified role.</td>
</tr>
<tr>
<td>Calculate</td>
<td>Converter</td>
<td>Automatically calculate Input and Output frequencies for mixer setup.</td>
</tr>
<tr>
<td>DeleteAllSegments</td>
<td>Converter5</td>
<td>Remove all segments from the segment table. (FCA Only)</td>
</tr>
<tr>
<td>DeleteSegment</td>
<td>Converter5</td>
<td>Remove segments from the segment table. (FCA Only)</td>
</tr>
<tr>
<td>DiscardChanges</td>
<td>Converter</td>
<td>Cancels changes that have been made to the Converter setup.</td>
</tr>
<tr>
<td>GetSourceByRole</td>
<td>Converter</td>
<td>Returns the name of a source that is assigned to the specified role.</td>
</tr>
<tr>
<td>GetSourceRoles</td>
<td>Converter</td>
<td>Returns the defined role names (&quot;RF2&quot;, &quot;LO1&quot;).</td>
</tr>
<tr>
<td>LoadFile</td>
<td>Converter</td>
<td>Loads a previously-configured mixer attributes file (.mxr)</td>
</tr>
<tr>
<td>ReCalculate</td>
<td>Converter5</td>
<td>Repeats the last calculation that was performed.</td>
</tr>
<tr>
<td>SaveFile</td>
<td>Converter</td>
<td>Saves the settings for the mixer/converter test setup to a mixer attributes file.</td>
</tr>
<tr>
<td>SegmentCalculate</td>
<td>Converter5</td>
<td>Performs calculate on a specific segment. (FCA Only)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveXAxisRange</td>
<td>Converter Sets or returns the swept frequency range to display on the X-axis.</td>
</tr>
<tr>
<td>Feature</td>
<td>Converter</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>AvoidSpurs</td>
<td>Converter5</td>
</tr>
<tr>
<td><strong>ConverterEmbeddedLO</strong></td>
<td>Converter2</td>
</tr>
<tr>
<td>DeviceInputPort</td>
<td>Converter6</td>
</tr>
<tr>
<td>DeviceOutputPort</td>
<td>Converter6</td>
</tr>
<tr>
<td>EnablePhase</td>
<td>Converter3</td>
</tr>
<tr>
<td>IFDenominator</td>
<td>Converter4</td>
</tr>
<tr>
<td>IFNumerator</td>
<td>Converter4</td>
</tr>
<tr>
<td>IFSideband</td>
<td>Converter4</td>
</tr>
<tr>
<td>IFStartFrequency</td>
<td>Converter4</td>
</tr>
<tr>
<td>IFStopFrequency</td>
<td>Converter4</td>
</tr>
<tr>
<td>IncludeReverseSweep</td>
<td>Converter4</td>
</tr>
<tr>
<td>InputDenominator</td>
<td>Converter</td>
</tr>
<tr>
<td>InputFixedFrequency</td>
<td>Converter</td>
</tr>
<tr>
<td>InputNumerator</td>
<td>Converter</td>
</tr>
<tr>
<td>InputPower</td>
<td>Converter</td>
</tr>
<tr>
<td>InputRangeMode</td>
<td>Converter</td>
</tr>
<tr>
<td>InputStartFrequency</td>
<td>Converter</td>
</tr>
<tr>
<td>InputStartPower</td>
<td>Converter4</td>
</tr>
</tbody>
</table>
InputStopFrequency Converter Sets or returns the stop frequency of the mixer input.

InputStopPower Converter Sets and returns the Stop Power value of the mixer Input Power.

IsInputGreaterThanLO Converter Specifies whether to use the Input frequency that is greater than the LO or less than the LO.

LODenominator Converter Sets or returns the denominator value of the LO Fractional Multiplier.

LOFixedFrequency Converter Sets or returns the fixed frequency of the specified LO.

LOName Converter Sets or returns the LO name.

LONumerator Converter Sets or returns the numerator value of the LO Fractional Multiplier.

LOPower Converter Sets or returns the value of the LO Power.

LORangeMode Converter Sets or returns the LO sweep mode to fixed or swept.

LOStage Converter Returns the number of stages.

LOStartFrequency Converter Sets or returns the start frequency of the specified LO.

LOStartPower Converter Sets or returns the start value of a LO Power sweep.

LOStopFrequency Converter Sets or returns the start frequency of the specified LO.

LOStopPower Converter Sets or returns the stop value of a LO Power sweep.

NominalIncidentPowerState Converter Sets or returns whether to use nominal power or measure actual incident power. (SMC ONLY)

NormalizePoint Converter Sets or returns the data point used for normalizing an SMC phase measurement. (SMC Only)

OutputFixedFrequency Converter Sets or returns the fixed frequency of the mixer output.
<table>
<thead>
<tr>
<th>Description</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OutputRangeMode</td>
<td>Converter</td>
<td>Sets or returns the Output sweep mode.</td>
</tr>
<tr>
<td>OutputSideband</td>
<td>Converter</td>
<td>Sets or returns the value of the output sideband.</td>
</tr>
<tr>
<td>OutputStartFrequency</td>
<td>Converter</td>
<td>Sets or returns the start frequency of the mixer output.</td>
</tr>
<tr>
<td>OutputStopFrequency</td>
<td>Converter</td>
<td>Sets or returns the stop frequency of the mixer output.</td>
</tr>
<tr>
<td>SegmentCount</td>
<td>Converter5</td>
<td>Read the number of segments. <em>(FCA Only)</em></td>
</tr>
<tr>
<td>SegmentFixedFrequency</td>
<td>Converter5</td>
<td>Set and read the CW Frequency for mixer segments in CW Sweep mode. <em>(FCA Only)</em></td>
</tr>
<tr>
<td>SegmentFixedPower</td>
<td>Converter5</td>
<td>Set and return the fixed power level for all ranges for mixer segments. <em>(FCA Only)</em></td>
</tr>
<tr>
<td>SegmentIFBandwidth</td>
<td>Converter5</td>
<td>Set and return the IF Bandwidth for the sweep segment. <em>(FCA Only)</em></td>
</tr>
<tr>
<td>SegmentIsInputGreaterThanLO</td>
<td>Converter5</td>
<td>Set and return whether to use the Input frequency that is greater than the LO. <em>(FCA Only)</em></td>
</tr>
<tr>
<td>SegmentMixingMode</td>
<td>Converter5</td>
<td>Set and return whether to set the mixing mode to high side or low side. <em>(FCA Only)</em></td>
</tr>
<tr>
<td>SegmentPoints</td>
<td>Converter5</td>
<td>Sets and returns the number of data points to be measured in the sweep segment. <em>(FCA Only)</em></td>
</tr>
<tr>
<td>SegmentRangeMode</td>
<td>Converter5</td>
<td>Sets or returns the sweep mode of the segment (all ranges). <em>(FCA Only)</em></td>
</tr>
<tr>
<td>SegmentStartFrequency</td>
<td>Converter5</td>
<td>Set and return the Start frequency for mixer segments. <em>(FCA Only)</em></td>
</tr>
<tr>
<td>SegmentState</td>
<td>Converter5</td>
<td>Set and return the ON/OFF state for mixer segments. <em>(FCA Only)</em></td>
</tr>
<tr>
<td>SegmentStopFrequency</td>
<td>Converter5</td>
<td>Set and return the Stop frequency for mixer segments. <em>(FCA Only)</em></td>
</tr>
</tbody>
</table>

**Converter History**
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Converter</td>
<td>8.55</td>
</tr>
<tr>
<td>Converter2</td>
<td>9.0</td>
</tr>
<tr>
<td>Converter3</td>
<td>9.2</td>
</tr>
<tr>
<td>Converter4</td>
<td>9.30</td>
</tr>
<tr>
<td>Converter5</td>
<td>9.33</td>
</tr>
<tr>
<td>Converter6</td>
<td>9.40</td>
</tr>
</tbody>
</table>
ConverterEmbeddedLO Object

Description

Provides access to the properties that allow IMDx and IMSpectrum measurements of converters that contain an embedded LO.

This interface contains all the same properties and methods of the Embedded LO interface (used for FCA measurements) EXCEPT access to the EmbeddedLODiagnostic Object.

Accessing the ConverterEmbeddedLO Interface

Access the Interface through the Converter Object.

```vba
Dim app
Set app = CreateObject("AgilentPNA835x.Application")
app.Reset
' Create a Measurement object, in this case using the IMeasurement interface
Dim meas
app.CreateCustomMeasurement 1, "Swept IMD Converters", -1
set meas = app.activemeasurement

Dim converter
set converter = app.ActiveChannel.GetConverter

Dim elo
set elo = converter.ConverterEmbeddedLO
elo.IsOn = 1
```

See Also:

VNA Automation Interfaces

The VNA Object Model

Making Embedded LO Measurements
### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResetLOFrequency</td>
<td>ICELO</td>
<td>Reset LO Delta frequency.</td>
</tr>
<tr>
<td>ResetTuningParameters</td>
<td>ICELO</td>
<td>Resets the tuning parameters to their defaults.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadbandTuningSpan</td>
<td>ICELO</td>
<td>Set broadband sweep span.</td>
</tr>
<tr>
<td>IsOn</td>
<td>ICELO</td>
<td>Set and return Embedded LO ON</td>
</tr>
<tr>
<td>LOFrequencyDelta</td>
<td>ICELO</td>
<td>Sets and returns LO delta frequency.</td>
</tr>
<tr>
<td>MaxPreciseTuningIterations</td>
<td>ICELO</td>
<td>Sets and returns precise tuning iterations.</td>
</tr>
<tr>
<td>NormalizePoint</td>
<td>ICELO</td>
<td>Sets and returns tuning point.</td>
</tr>
<tr>
<td>PreciseTuningTolerance</td>
<td>ICELO</td>
<td>Sets and returns precise tuning tolerance.</td>
</tr>
<tr>
<td>TuningIFBW</td>
<td>ICELO</td>
<td>Sets and returns the IF Bandwidth for tuning sweeps.</td>
</tr>
<tr>
<td>TuningMode</td>
<td>ICELO</td>
<td>Sets and returns the method used to determine the embedded LO Frequency.</td>
</tr>
<tr>
<td>TuningSweepInterval</td>
<td>ICELO</td>
<td>Set how often a tuning sweep is performed.</td>
</tr>
</tbody>
</table>

### ICELO History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICELO</td>
<td>9.00</td>
</tr>
</tbody>
</table>
CorrectionMethods Object

Description

These methods and properties control various error-correction settings for a channel.

Accessing the object

```vba
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim corrMethods as CorrectionMethods
corrMethods = chan.CorrectionMethods
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResetPortValues</td>
<td>ICorrectionMethods2</td>
<td>Resets the full and response list to their default values.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CorrectionSubsettingState</td>
<td>ICorrectionMethods2</td>
</tr>
<tr>
<td>FullyCorrectedPorts</td>
<td>ICorrectionMethods2</td>
</tr>
<tr>
<td>MatchCorrectPower</td>
<td>ICorrectionMethods</td>
</tr>
<tr>
<td>ResponseCorrectedPorts</td>
<td>ICorrectionMethods2</td>
</tr>
</tbody>
</table>

IAuxTrigger History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICorrectioMethods</td>
<td>9.30</td>
</tr>
<tr>
<td>ICorrectionMethods2</td>
<td>12.70</td>
</tr>
</tbody>
</table>
DCStimulus Object

Description
The DCStimulus object allows you to make DC Source settings for each channel.

Accessing the DCStimulus Object
You can obtain a handle to a DCStimulus object through the Channel object.

```vba
dim app
Set app = CreateObject("AgilentPNA835x.Application")
dim chan
Set chan = app.ActiveChannel
chan.NumberofPoints = 3
dim DC
Set DC = chan.DCStimulus
DC.EnableAllOutput = True
DC.State("AO1") = True
```

See Also
- Learn about DC Source Control
- Configure an External DC Device
- The VNA Object Model
## Methods

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

## Properties

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnableAllOutput</td>
</tr>
<tr>
<td>LimitMax</td>
</tr>
<tr>
<td>LimitMin</td>
</tr>
<tr>
<td>ListData</td>
</tr>
<tr>
<td>Sources</td>
</tr>
<tr>
<td>Start</td>
</tr>
<tr>
<td>State</td>
</tr>
<tr>
<td>Stop</td>
</tr>
<tr>
<td>Target</td>
</tr>
</tbody>
</table>

## DCStimulus History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDCStimulus</td>
<td>9.5</td>
</tr>
<tr>
<td>IDCStimulus2</td>
<td>12.80</td>
</tr>
</tbody>
</table>
DifferentialIQ Object

Description

Controls the Differential I/Q application settings.

Accessing the Diff IQ and Diff IQMeas objects

```vbnet
Dim app as AgilentPNA835x.Application
app.CreateCustomMeasurementEx 2, "Differential I/Q", "IPwrF1"
Dim DIQ
Set DIQ = app.ActiveChannel.CustomChannelConfiguration
Dim DiqMeas
Set DiqMeas = app.ActiveMeasurement.CustomMeasurementConfiguration
```

See Also:

- Differential I/Q application
- VNA Automation Interfaces
- The VNA Object Model

Diff IQ Setup Dialog

![Diff IQ Setup Dialog](image)

- AddRange Method
- DeleteRange Method
- RangeCount Method
### Range Settings Dialog

<table>
<thead>
<tr>
<th>Frequency</th>
<th>RangeStartFrequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>RangeStopFrequency</td>
<td></td>
</tr>
<tr>
<td>RangeIFBW Property</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Coupling</th>
<th>RangeCoupleState Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>RangeCoupleId Property</td>
<td></td>
</tr>
<tr>
<td>RangeOffset Property</td>
<td></td>
</tr>
<tr>
<td>RangeOffsetUp Property</td>
<td></td>
</tr>
<tr>
<td>RangeMultiplier Property</td>
<td></td>
</tr>
<tr>
<td>RangeDivisor Property</td>
<td></td>
</tr>
</tbody>
</table>

### Source Configuration Dialog

<table>
<thead>
<tr>
<th>Power</th>
<th>SourceState Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>PowerSweepState Property</td>
<td></td>
</tr>
<tr>
<td>PortStartPower Property</td>
<td></td>
</tr>
<tr>
<td>PortStopPower Property</td>
<td></td>
</tr>
<tr>
<td>PortLevelingMode Property</td>
<td></td>
</tr>
<tr>
<td>PortAttenuator Property</td>
<td></td>
</tr>
<tr>
<td>AutoRangeState Property</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Phase</th>
<th>PortPhaseState Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>PhaseSweepState Property</td>
<td></td>
</tr>
<tr>
<td>PortPhaseStart Property</td>
<td></td>
</tr>
<tr>
<td>PortPhaseStop Property</td>
<td></td>
</tr>
<tr>
<td>PortReference Property</td>
<td></td>
</tr>
<tr>
<td>PortPhaseParameter Property</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Match Correction</th>
<th>MatchState Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>MatchTestReceiver Property</td>
<td></td>
</tr>
<tr>
<td>MatchRefReceiver Property</td>
<td></td>
</tr>
<tr>
<td>MatchFrequencyRange Property</td>
<td></td>
</tr>
</tbody>
</table>
**Edit Parameters Dialog**

- DefineParameter Method
- DeleteParameter Method
- ParameterList Property
- Save Method (DIQ2)
- Load Method (DIQ2)

**DIQ History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDIQ</td>
<td>10.25</td>
</tr>
<tr>
<td>IDIQ2</td>
<td>10.40</td>
</tr>
</tbody>
</table>
Display Object

Description

Controls the settings of the front panel screen.

Accessing the Display object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim disp As IDisplay
Set disp = app.Display
```

See Also:

- PNA Automation Interfaces
- The PNA Object Model

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>See History</strong></td>
</tr>
</tbody>
</table>

None

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PowerSpinResolution</td>
<td>IDisplay</td>
<td>Set and read the power level knob resolution.</td>
</tr>
</tbody>
</table>

IDisplay History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDisplay</td>
<td>13.25</td>
</tr>
</tbody>
</table>
E5091Testsets Collection

Description

Two testsets can be connected and controlled by the VNA at any time.

The item number in the testsets collection is set by the DIP switches on the testset rear-panel. The valid item numbers are 1 and 2. If the testset DIP switches are set to 1, then item number in the collection is 1, and so forth. See your E5091A documentation for more information.

If the specified testset is not connected to USB or not ON, then setting Enabled = True will return an error. All other properties can be set when the testset is not connected.

Accessing the E5091Testsets collection

Child of the Application Object. Get a handle to one of the E5091Testset objects by specifying an item of the collection.

```vba
Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")
Dim testsets As E5091Testsets
Set testsets = pna.E5091Testsets
Dim tset1 As E5091Testset
Set tset1 = testsets(1)
```

See Also:

- E5091Testset Control COM Example
- E5091Testset Object
- Collections in the Analyzer
- The VNA Object Model

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>Use to get a handle to a testset in the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of items in a collection of objects.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the current naNetworkAnalyzer application.</td>
</tr>
</tbody>
</table>
## E5091 Testsets History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE5091Testsets</td>
<td>5.2</td>
</tr>
</tbody>
</table>
E5091Testset Object

Description

There can be two test sets connected and controlled by the VNA at any time.

The item number in the testsets collection is set by the DIP switches on the test set rear-panel. The valid item numbers are 1 and 2. If the test set DIP switches are set to 1, then item number in the collection is 1, and so forth. See your E5091A documentation for more information.

If the specified test set is not connected to USB or not ON, then setting Enabled = True will return an error. All other properties can be set when the test set is not connected.

Accessing the E5091Testset object

Child of the Application Object. Get a handle to a E5091Testset object by specifying an item of the collection.

```
Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")
Dim testsets As E5091Testsets
Set testsets = pna.E5091Testsets
Dim tset1 As E5091Testset
Set tset1 = testsets(1)
```

See Also:

- E5091Testset Control COM Example
- E5091 TestSet Control
- E5091Testsets Collection
- TestsetControl Object (for different test sets)
- The VNA Object Model
<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ControlLines</td>
<td>Sets the control lines of the specified E5091A.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Enables and disables (ON/OFF) the port mapping and control line output of the specified testset.</td>
</tr>
<tr>
<td>ID</td>
<td>Returns the test set ID number.</td>
</tr>
<tr>
<td>NumberOfPorts</td>
<td>Reads the number of ports (7 or 9) that are on the specified E5091A test set.</td>
</tr>
<tr>
<td>OutputPort</td>
<td>Switches an input to one of the valid outputs on the specified E5091A.</td>
</tr>
<tr>
<td>ShowProperties</td>
<td>Turns ON and OFF the display of the test set control status bar.</td>
</tr>
</tbody>
</table>

**E5091Testset History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE5091Testset</td>
<td>5.2</td>
</tr>
</tbody>
</table>
ECalModule Object

Allows access to ECal modules that are connected to the VNA.

Accessing the ECalModule object

Get a handle to a ECalModule object by using the ECalModules collection. This is done through the CalManager object with the app.GetCalManager Method.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim pna
pna.Preset
Const chanNum = 1
pna.Channels(chanNum).StopFrequency = 20E9 ' for a 20 GHz ECal mod
Const pnaPortNumber = 1
Const ecalCharacterizationNum = 0
Dim calMgr
Set calMgr = pna.GetCalManager
Dim ecalPortNumber ' The returned ECal port number is a 1-based number
' (1 = Port A, 2 = Port B, etc)
ecalPortNumber = calMgr.ECalModules(1).AutoOrient(chanNum, pnaPortNumber, ecalCharacterizationNum)
MsgBox "ECal port number attached to PNA port 1 = " & ecalPortNumber
```

See Also:

- ECalModules Collection
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>See History</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AutoOrient</td>
<td>IECalModule</td>
<td>Returns the orientation (which ECal port is connected to which VNA port) outside of the context of a calibration.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**IECalModule History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IECalModule</td>
<td>8.50</td>
</tr>
</tbody>
</table>
ECalModules Collection

Description

A collection that provides access to ECal modules that are connected to the VNA.

Accessing the ECalModules collection

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim eCalMods As ECalModules
Set eCalMods = app.GetCalManager.ECalModules
```

See Also:

- Using ECal
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>IECalModules</td>
<td>Use to get a handle to a ECalModule Object in the collection.</td>
</tr>
<tr>
<td>OutputSNPFromECal</td>
<td>IECalModules2</td>
<td>Read S parameter of ECal Thur from the ECal memory and save it as s2p file.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>IECalModules</td>
<td>Returns the number of objects in the collection.</td>
</tr>
<tr>
<td>Parent</td>
<td>IECalModules</td>
<td>Returns a handle to the CalManager Object</td>
</tr>
</tbody>
</table>

IECalModules History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IECalModules</td>
<td>8.50</td>
</tr>
<tr>
<td>IECalModules2</td>
<td>12.50.01</td>
</tr>
</tbody>
</table>
ECalUserCharacterizer Object

Description

Controls the settings used to perform an ECal User Characterization. An S-Parameter channel must already be calibrated. These commands will then measure the ECal module with adapters, cables, or fixtures to be included in the User Characterization, allow descriptive text to be entered, then save the User Characterization to the ECal module.

Up to 12 User Characterizations can be stored in an ECal module.

You can NOT perform a remote User Characterization of a 4-port ECal module using a 2-port VNA. This can only be done from the front panel user interface.

Accessing the ECalUserCharacterizer Interface

Access the Interface through the ICalManager Object.

```vba
dim app as agilentPNA835x.application
set app = createobject("agilentPNA835x.application")

dim mgr as icalmanager
set mgr = app.getcalmanager
dim ecalcharacterizer
set ecalcharacterizer = mgr.getecalusercharacterizer()
```

See Also:

Example: Perform an ECal User Characterization

VNA Automation Interfaces

The VNA Object Model

About User Characterization

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcquireStep</td>
<td>ECalUserCharacterizer</td>
<td>Measure the ECal module.</td>
</tr>
<tr>
<td>GenerateSteps</td>
<td>ECalUserCharacterizer</td>
<td>Returns the number of steps required to complete the calibration.</td>
</tr>
</tbody>
</table>
GetStepDescription | ECalUserCharacterizer | Returns the description of the specified step in the calibration process.

Initialize       | ECalUserCharacterizer | Superseded with InitializeEx Method

InitializeEx   | ECalUserCharacterizer2 | Initiates a User Characterization of an ECal module.

SaveToFileDiskMemory | ECalUserCharacterizer2 | Saves the User Characterization to VNA disk memory.

SaveToFileECal | ECalUserCharacterizer | Saves the User Characterization to the ECal module.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CharacterizationNumber</td>
<td>Sets and reads the number to which the user characterization will be stored in the ECal module.</td>
</tr>
<tr>
<td>ConnectorType</td>
<td>Sets or queries the connector type for the specified port.</td>
</tr>
<tr>
<td>ECalID</td>
<td>Select the model and serial number of the ECal module to be characterized.</td>
</tr>
<tr>
<td>InSituCharacterization</td>
<td>Sets or returns whether the CalPod module will be characterized as an in situ device.</td>
</tr>
<tr>
<td>PortDescription</td>
<td>Sets and reads the description of the adapters, cable, or fixture to be included in the user characterization.</td>
</tr>
<tr>
<td>SupportsInSituCharacterization</td>
<td>Returns whether the device is a CalPod module.</td>
</tr>
</tbody>
</table>
UserDescriptionOfPNA  
ECalUserCharacterizer  
Sets and reads a user description of the VNA used to perform the User Characterization.

UserName  
ECalUserCharacterizer  
Sets and reads the description of the person and/or company who is producing the ECal user characterization.

ValidConnectorTypes  
ECalUserCharacterizer  
Returns a list of connector names that are valid for use with user-characterized ECal modules.

IECalUserCharacterizer History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IECalUserCharacterizer</td>
<td>8.33</td>
</tr>
<tr>
<td>IECalUserCharacterizer2</td>
<td>9.00</td>
</tr>
</tbody>
</table>
EmbeddedLO Object

Description

Provides access to the properties that allow measurement of mixers that contain an embedded LO.

Accessing the EmbeddedLO Interface

Access the Interface through the IMixer Object.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset

' FCA Measurements can't share the channel with standard measurements
' Because preset creates a single measurement in channel 1, we first delete the
' standard measurement
Dim standardMeas As IMeasurement
Set standardMeas = app.ActiveMeasurement
standardMeas.Delete

' Create a Measurement object, in this case using the IMeasurement interface
Dim meas As IMeasurement
Set meas = app.CreateCustomMeasurementEx(1, "SMC_Forward.SMC_ForwardMeas", "SC21")

' See if this measurement object supports IMixer
Dim mixer As IMixer
Dim embeddedLO
Set embeddedLO = mixer.EmbeddedLO
```

Note: The Find Now feature on the Embedded LO dialog is performed remotely by doing a 'Single Sweep" with ELO enabled. See example program.

See an example program that shows how to create and calibrate a standard SMC or VMC measurement or a fixed output SMC measurement.

See Also:

Example program

VNA Automation Interfaces

The VNA Object Model

Making Embedded LO Measurements
### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResetLOFrequency</td>
<td>IEmbeddedLO</td>
<td>Reset LO Delta frequency.</td>
</tr>
<tr>
<td>ResetTuningParameters</td>
<td>IEmbeddedLO</td>
<td>Resets the tuning parameters to their defaults.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadbandTuningSpan</td>
<td>IEmbeddedLO</td>
<td>Set broadband sweep span.</td>
</tr>
<tr>
<td>EmbeddedLODiagnostic</td>
<td>IEmbeddedLO</td>
<td>Provides access to the status of tuning sweeps.</td>
</tr>
<tr>
<td>IsOn</td>
<td>IEmbeddedLO</td>
<td>Set and return Embedded LO ON</td>
</tr>
<tr>
<td>LOFrequencyDelta</td>
<td>IEmbeddedLO</td>
<td>Sets and returns LO delta frequency.</td>
</tr>
<tr>
<td>MaxPreciseTuningIterations</td>
<td>IEmbeddedLO</td>
<td>Sets and returns precise tuning iterations.</td>
</tr>
<tr>
<td>NormalizePoint</td>
<td>IEmbeddedLO</td>
<td>Sets and returns tuning point.</td>
</tr>
<tr>
<td>PreciseTuningTolerance</td>
<td>IEmbeddedLO</td>
<td>Sets and returns precise tuning tolerance.</td>
</tr>
<tr>
<td>TuningIFBW</td>
<td>IEmbeddedLO</td>
<td>Sets and returns the IF Bandwidth for tuning sweeps.</td>
</tr>
<tr>
<td>TuningMode</td>
<td>IEmbeddedLO</td>
<td>Sets and returns the method used to determine the embedded LO Frequency.</td>
</tr>
<tr>
<td>TuningSweepInterval</td>
<td>IEmbeddedLO</td>
<td>Set how often a tuning sweep is performed.</td>
</tr>
</tbody>
</table>

### IEmbeddedLO History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEmbeddedLO</td>
<td>7.21</td>
</tr>
</tbody>
</table>
EmbeddedLODiagnostic Object

Description

Allows access to the properties that provide information about the broadband and precise tuning of an embedded LO.

Accessing the EmbeddedLODiagnostic Interface

Access the Interface through the EmbeddedLO Object.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset

' FCA Measurements can't share the channel with standard measurements
' Because preset creates a single measurement in channel 1, we first delete the standard measurement
Dim standardMeas As IMeasurement
Set standardMeas = app.ActiveMeasurement
standardMeas.Delete

' Create a Measurement object, in this case using the IMeasurement interface
Dim meas As IMeasurement
Set meas = app.CreateCustomMeasurementEx(1, "SMC_Forward.SMC_ForwardMeas", "SC21")

' See if this measurement object supports IMixer
Dim mixer As IMixer
Dim embeddedLO
Set embeddedLO = mixer.EmbeddedLO

Dim embeddedLODiagnostic
Set embeddedLODiagnostic = embeddedLO.EmbeddedLODiagnostic
```

See an example program that shows how to create and calibrate a standard SMC or VMC measurement or a fixed output SMC measurement.

See Also:

VNA Automation Interfaces

The VNA Object Model

Making Embedded LO Measurements

EmbeddedLO Object
### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear</td>
<td>IELODiag</td>
<td>Clear current diagnostic information.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsMarkerOn</td>
<td>IELODiag</td>
<td>Was a marker was used for a tuning sweep?</td>
</tr>
<tr>
<td>LODeltaFound</td>
<td>IELODiag</td>
<td>Returns the LO frequency delta from this tuning sweep.</td>
</tr>
<tr>
<td>NumberOfSweeps</td>
<td>IELODiag</td>
<td>Get number of tuning sweeps.</td>
</tr>
<tr>
<td>MarkerAnnotation</td>
<td>IELODiag</td>
<td>Get the marker annotation.</td>
</tr>
<tr>
<td>MarkerPosition</td>
<td>IELODiag</td>
<td>Get the marker X-axis position.</td>
</tr>
<tr>
<td>Parameter</td>
<td>IELODiag</td>
<td>Returns the tuning sweep parameter name.</td>
</tr>
<tr>
<td>StatusAsString</td>
<td>IELODiag</td>
<td>Get result of the last tuning sweeps.</td>
</tr>
<tr>
<td>StepData</td>
<td>IELODiag</td>
<td>Get a tuning sweep data.</td>
</tr>
<tr>
<td>StepTitle</td>
<td>IELODiag</td>
<td>Returns the tuning sweep title.</td>
</tr>
<tr>
<td>XAxisAnnotation</td>
<td>IELODiag</td>
<td>Get the tuning sweep X axis annotation.</td>
</tr>
<tr>
<td>XAxisStart</td>
<td>IELODiag</td>
<td>Get the Start sweep value.</td>
</tr>
<tr>
<td>XAxisStop</td>
<td>IELODiag</td>
<td>Get the Stop sweep value.</td>
</tr>
<tr>
<td>YAxisAnnotation</td>
<td>IELODiag</td>
<td>Get the tuning sweep Y axis annotation.</td>
</tr>
</tbody>
</table>

### History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEmbeddedLODiagnostic</td>
<td>7.21</td>
</tr>
</tbody>
</table>
ENRFile Object

Description

Provide commands for creating or editing an ENR file. This is rarely necessary as ENR files, which contain factory calibrated data, are typically provided by the manufacturer of the noise source.

Learn more about Noise Figure Application

Accessing the ENRFile object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim enr As ENRFile
Set enr = app.ENRFile
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Program

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetENRData</td>
<td>IENRFile</td>
<td>Read the ENR calibration data from VNA memory.</td>
</tr>
<tr>
<td>PutENRData</td>
<td>IENRFile</td>
<td>Write the ENR calibration data to VNA memory.</td>
</tr>
<tr>
<td>LoadENRFile</td>
<td>IENRFile</td>
<td>Recalls an ENR file from disk into VNA Memory.</td>
</tr>
<tr>
<td>SaveENRFile</td>
<td>IENRFile</td>
<td>Saves an ENR file from VNA memory to disk.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENRID</td>
<td>IENRFile</td>
<td>Sets and returns ID of ENR table.</td>
</tr>
<tr>
<td>ENRSN</td>
<td>IENRFile</td>
<td>Sets and returns the serial number of the noise source.</td>
</tr>
<tr>
<td>NoiseCalTemperature</td>
<td>IENRFile</td>
<td>Read the cal temperature of the noise source.</td>
</tr>
<tr>
<td>Interface</td>
<td>Introduced with VNA Rev:</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>------------------------</td>
<td></td>
</tr>
<tr>
<td>IENRFile</td>
<td>8.0</td>
<td></td>
</tr>
</tbody>
</table>
ExternalDCDevice Object

Description

The ExternalDCDevice object allows you to set unique properties for a DC Meter or DC Source.

Accessing the ExternalDCDevice Object

You can obtain a handle to a ExternalDCDevice Object through ExtendedProperties

```vbnet
dim app
Set app = CreateObject("AgilentPNA835x.Application")
dim externalDevices
Set externalDevices = app.ExternalDevices
dim devicecount
devicecount = externalDevices.count

'****** Configure a DC Meter ************
externalDevices.Add "MyDCMeter"
dim newExtDCMeter
Set newExtDCMeter = externalDevices.Item("MyDCMeter")
newExtDCMeter.DeviceType = "DC Meter"
newExtDCMeter.Driver = "DCMeter"
'newExtDCMeter.Active = True
newExtDCMeter.IOConfiguration= "GPIB0::16::INSTR"
dim extDCMtr
Set extDCMtr = newExtDCMeter.ExtendedProperties
extDCMtr.PointDwell = .05

'****** Configure a DC Supply ************
externalDevices.Add "MyDCSupply"
```
dim newExtDCSupply
Set newExtDCSupply = externalDevices.Item("MyDCSupply")
newExtDCSupply.DeviceType = "DC Source"
newExtDCSupply.Driver = "DCSource"
'newExtDCSupply.Active = True
newExtDCSupply.IOConfiguration= "GPIB0::16::INSTR"

dim extDCSupply
Set extDCSupply = newExtDCSupply.ExtendedProperties
extDCSupply.PointDwell = .05

See Also:
Configure an External Device
The VNA Object Model

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AbortSweepCmd</td>
<td>IExternalDCDevice3</td>
<td>Sets and returns the Abort Sweep command for an external DC Source and an</td>
</tr>
<tr>
<td></td>
<td></td>
<td>external DC Meter.</td>
</tr>
<tr>
<td>AfterSweepCmd</td>
<td>IExternalDCDevice3</td>
<td>Sets and returns the After Sweep command for an external DC Source and an</td>
</tr>
<tr>
<td></td>
<td></td>
<td>external DC Meter.</td>
</tr>
<tr>
<td>BeforeSweepCmd</td>
<td>IExternalDCDevice3</td>
<td>Sets and returns the Before Sweep command for an external DC Source and an</td>
</tr>
<tr>
<td></td>
<td></td>
<td>external DC Meter.</td>
</tr>
<tr>
<td>CurrentLimit</td>
<td>IExternalDCDevice2</td>
<td>Sets and returns the current limit value.</td>
</tr>
<tr>
<td>DCCorrection</td>
<td>IExternalDCDevice</td>
<td>Sets and returns correction ON/OFF.</td>
</tr>
<tr>
<td>DCOffset</td>
<td>IExternalDCDevice</td>
<td>Sets and returns the offset correction value.</td>
</tr>
</tbody>
</table>
DCScale  IExternalDCDevice  Sets and returns the scale correction value.
DCTYPE   IExternalDCDevice  Sets and returns the DC Type (Units).
ErrorQuery IExternalDCDevice3 Sets and returns the Error Query command for an external DC Source and an external DC Meter.
ExitCmd   IExternalDCDevice3 Sets and returns the Disable I/O command for an external DC Source and an external DC Meter.
IDQuery   IExternalDCDevice3 Sets and returns the ID Query command for an external DC Source and an external DC Meter.
InitCmd   IExternalDCDevice3 Sets and returns the Enable I/O command for an external DC Source and an external DC Meter.
LimitCmd  IExternalDCDevice4 Sets and returns the limit type (current or voltage) of the external DC Source.
MaxOutput IExternalDCDevice3 Sets and returns the “Define Max As” value for an external DC Source.
MaxOutputState IExternalDCDevice3 Sets and returns the “Define Max As” ON/OFF state for an external DC Source.
MinOutput IExternalDCDevice3 Sets and returns the “Define Min As” value for an external DC Source.
MinOutputState IExternalDCDevice3 Sets and returns the “Define Min As” ON/OFF state for an external DC Source.
PointCmd  IExternalDCDevice3 Sets and returns the Point Read Commands for an external DC Source or Point Set Commands for an external DC Meter.
PointDwell IExternalDCDevice  Sets and returns the "Dwell Before/After Point" value.
SweepDwell IExternalDCDevice  Sets and returns the "Dwell Before Sweep" value.
VoltageLimit IExternalDCDevice2 Sets and returns the voltage limit value.

**ExternalDCDevice History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IExternalDCDevice</td>
<td>9.5</td>
</tr>
<tr>
<td>IExternalDCDevice2</td>
<td>12.70</td>
</tr>
<tr>
<td>IExternalDCDevice3</td>
<td>12.80</td>
</tr>
<tr>
<td>IExternalDCDevice4</td>
<td>13.25</td>
</tr>
</tbody>
</table>

1060
ExternalDevice Object

Description

ExternalDevice objects allow you to set properties and methods for each external device.

Accessing the ExternalDevice Object

Obtain a handle to an External Device by specifying an item in the External Devices collection.

Although the following example shows how to create a PMAR external device, you can substitute any supported external device for "PMAR" and DeviceType: "Power Meter".

Supported External Device Objects

ExternalSource Object

ExternalDCDevice Object

PowerSensorAsReceiver Object

ExternalPulseGenerator Object

dim app
Set app = CreateObject("AgilentPNA835x.Application")
dim externalDevices
Set externalDevices = app.ExternalDevices
dim devicecount
devicecount = externalDevices.count
externalDevices.Add "NewPMAR"
dim newExternalDevice
Set newExternalDevice = externalDevices.Item("NewPMAR")
newExternalDevice.DeviceType = "Power Meter"
newExternalDevice.IOConfiguration= "GPIB0::14::INSTR"
newExternalDevice.IOEnable = true
See Also:

The VNA Object Model

Learn how to Configure an External Device

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoadFile</td>
<td>IExternalDevice2</td>
<td>Recalls an external device configuration file from the VNA hard drive.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Currently, only DC Meter and DC Supply devices are supported.</td>
</tr>
<tr>
<td>SaveFile</td>
<td>IExternalDevice2</td>
<td>Saves an external device configuration file to the VNA hard drive.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Currently, only DC Meter and DC Supply devices are supported.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>Sets and returns the state of activation for an external device.</td>
</tr>
<tr>
<td>DeviceType</td>
<td>Sets and returns the DeviceType (source, power meter) for the external device.</td>
</tr>
<tr>
<td>Driver</td>
<td>Sets and returns the external device driver (model).</td>
</tr>
<tr>
<td>ExtendedProperties</td>
<td>Provides access to properties that are unique to the external device type.</td>
</tr>
<tr>
<td>IOConfiguration</td>
<td>Sets and returns the method of communication and address for the external device.</td>
</tr>
<tr>
<td>IOEnable</td>
<td>Sets and returns whether an external device is available for IO communication with the VNA.</td>
</tr>
<tr>
<td>Name</td>
<td>Sets and returns the name of the External Device.</td>
</tr>
<tr>
<td>TimeOut</td>
<td>Sets and returns the time out value for communication with the external device.</td>
</tr>
<tr>
<td>Interface</td>
<td>Introduced with VNA Rev:</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>IExternalDevice</td>
<td>9.0</td>
</tr>
<tr>
<td>IExternalDevice2</td>
<td>9.50</td>
</tr>
</tbody>
</table>
ExternalDevices Collection

Description

ExternalDevices collection provides access to an ExternalDevice object.

Accessing the ExternalDevices collection

The ExternalDevices collection is a property of the main Application Object. You can obtain a handle to an External Device by specifying an item in the collection.

```vba
dim app
Set app = CreateObject("AgilentPNA835x.Application")
dim externalDevices
Set externalDevices = app.ExternalDevices
dim devicecount
devicecount = externalDevices.count
externalDevices.Add "NewPMAR"
dim newExternalDevice
Set newExternalDevice = externalDevices.Item("NewPMAR")
newExternalDevice.DeviceType = "Power Meter"
newExternalDevice.IOConfiguration= "GPIB0::14::INSTR"
newExternalDevice.IOEnable = true
```

See Also:

Example: Create a PMAR Device and Measurement

Configure an External Device

ExternalDevice Object

The VNA Object Model
### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>IExternalDevices</td>
<td>Adds an external device to the system.</td>
</tr>
<tr>
<td>Item</td>
<td>IExternalDevices</td>
<td>Use to get a handle to an external device in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>IExternalDevices</td>
<td>Removes an external device from the system</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>IExternalDevices</td>
<td>Returns the number of devices in the ExternalDevices collection.</td>
</tr>
<tr>
<td>DeviceNames</td>
<td>IExternalDevices2</td>
<td>Returns the device names in the collection</td>
</tr>
<tr>
<td>HasItem</td>
<td>IExternalDevices</td>
<td>Returns a value indicating whether the specified external devices is configured.</td>
</tr>
<tr>
<td>IsDevicePresent</td>
<td>IExternalDevices2</td>
<td>Returns whether the named device is present on the bus for which it is configured.</td>
</tr>
<tr>
<td>Items</td>
<td>IExternalDevices</td>
<td>Returns an array of configured devices in the system.</td>
</tr>
<tr>
<td>Parent</td>
<td>IExternalDevices</td>
<td>Returns a handle to the current application object.</td>
</tr>
</tbody>
</table>

### IExternalDevices History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IExternalDevices</td>
<td>9.0</td>
</tr>
<tr>
<td>IExternalDevices2</td>
<td>9.50</td>
</tr>
</tbody>
</table>
ExternalPulseGenerator Object

Description

The ExternalPulseGenerator object allows you to set unique properties and methods for each external pulse generator.

Accessing the ExternalPulseGenerator Object

You can obtain a handle to an ExternalSource Object through ExtendedProperties.

```vba
dim app
Set app = CreateObject("AgilentPNA835x.Application")
dim externalDevices
Set externalDevices = app.ExternalDevices
dim devicecount
devicecount = externalDevices.count
externalDevices.Add "81110"
dim newExternalDevice
Set newExternalDevice = externalDevices.Item("81110")
newExternalDevice.DeviceType = "Pulse Generator"
newExternalDevice.Driver = "AGPULSEGEN"
newExternalDevice.Active = True
newExternalDevice.IOConfiguration= "GPIB0::19::INSTR"
dim extPulseGen
Set extPulseGen = newExternalDevice.ExtendedProperties
extPulseGen.HighAmplitude = 3
```

See Also:

Configure an External Device

The VNA Object Model
### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HighAmplitude</td>
<td>ExternalPulseGenerator</td>
<td>Sets the High amplitude (voltage).</td>
</tr>
<tr>
<td>LoadImpedance</td>
<td>ExternalPulseGenerator</td>
<td>Sets the Load impedance.</td>
</tr>
<tr>
<td>LowAmplitude</td>
<td>ExternalPulseGenerator</td>
<td>Sets the Low amplitude (voltage).</td>
</tr>
<tr>
<td>PrimaryMode</td>
<td>ExternalPulseGenerator2</td>
<td>Sets the primary mode for the ext. pulse generator.</td>
</tr>
<tr>
<td>OutputChannel</td>
<td>ExternalPulseGenerator</td>
<td>Sets the Output channel (port) of the pulse generator.</td>
</tr>
<tr>
<td>SourceImpedance</td>
<td>ExternalPulseGenerator</td>
<td>Sets the Source impedance.</td>
</tr>
</tbody>
</table>

**ExternalPulseGenerator History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IExternalPulseGenerator</td>
<td>9.50</td>
</tr>
<tr>
<td>ExternalPulseGenerator2</td>
<td>9.50</td>
</tr>
</tbody>
</table>
ExternalSMUDevice Object

Description

ExternalSMUDevice objects allow you to set properties and methods for each external SMU device.

Accessing the ExternalDevice Object

Obtain a handle to an External Device by specifying an item in the External Devices collection.

dim app
Set app = CreateObject("AgilentPNA835x.Application")
dim externalSMUDevices
Set externalDevices = app.ExternalDevices
dim devicecount
devicecount = externalDevices.count
externalDevices.Add "NewSMU"
dim newExternalSMUDevice
Set newExternalSMUDevice = externalDevices.Item("NewSMU")
newExternalSMUDevice.DeviceType = "SMU"
newExternalSMUDevice.Active = True
newExternalSMUDevice.IOConfiguration= "GPIB0::14::INSTR"
newExternalSMUDevice.IOEnable = true
dim extSMU
Set extSMU = newExternalSMUDevice.ExtendedProperties
extSMU.PointDwell = .05

See Also:
The VNA Object Model
Learn how to Configure an External SMU Device

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>See History</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

None

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ChanActive</td>
<td>IExternalSMUDevice Sets and returns the state of activation for an external device.</td>
</tr>
</tbody>
</table>

ExternalSMUDevice History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IExternalSMUDevice</td>
<td>12.85</td>
</tr>
</tbody>
</table>
ExternalSource Object

Description

The ExternalSource object allows you to set unique properties and methods for each external source.

Accessing the ExternalSource Object

You can obtain a handle to an ExternalSource Object through ExtendedProperties

```vba
Dim app
Set app = CreateObject("AgilentPNA835x.Application")
Dim externalDevices
Set externalDevices = app.ExternalDevices
Dim devicecount
devicecount = externalDevices.count
externalDevices.Add "NewPSG"
Dim newExternalDevice
Set newExternalDevice = externalDevices.Item("NewPSG")
newExternalDevice.DeviceType = "Source"
newExternalDevice.IOConfiguration= "GPIB0::14::INSTR"
Dim PSG
Set PSG = newExternalDevice.ExtendedProperties
PSG.DwellPerPoint = 5
```

See Also:

Configure an External Device

The VNA Object Model
<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DwellPerPoint</td>
<td>Sets and returns the dwell time for an external source.</td>
</tr>
<tr>
<td>EnableModulationControl</td>
<td>Enables/disables control of modulation in an external source.</td>
</tr>
<tr>
<td>IsEnhancedModulationControl</td>
<td>Enables/disables control of modulation in an external source capable of downloading modulation files.</td>
</tr>
<tr>
<td>TriggerMode</td>
<td>Sets and returns the trigger mode (Software / Hardware) for an external source.</td>
</tr>
<tr>
<td>TriggerPort</td>
<td>Sets and returns the VNA port through which an external source is to be triggered.</td>
</tr>
</tbody>
</table>

ExtendedProperties History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IExternalSource</td>
<td>9.0</td>
</tr>
<tr>
<td>IExternalSource2</td>
<td>10.49</td>
</tr>
<tr>
<td>IExternalSource3</td>
<td>13.60</td>
</tr>
</tbody>
</table>
ExternalTestsets Collection

Description

ExternalTestsets collection provides access to a TestsetControl object. Only one external testset can be controlled by the VNA at any time.

Accessing the ExternalTestsets collection

The ExternalTestsets collection is a property of the main Application Object. You can obtain a handle to a testset by specifying an item in the collection.

Visual Basic Example

```vba
Dim pna
Dim testsets As ExternalTestsets
Dim tset1 As TestsetControl
Set pna = CreateObject("AgilentPNA835x.Application")
Set testsets = pna.ExternalTestsets
Set tset1 = testsets(1)
' make COM calls on tset1 object
End Sub
```

See Also:

ExternalTestset Control COM Example

About External TestSet Control

TestsetControl Object

The VNA Object Model
### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a testset to the collection and loads a test set configuration file.</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle to a testset in the collection.</td>
</tr>
<tr>
<td>TestsetCatalog</td>
<td>Returns a list of supported test sets.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of items in a collection of objects.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the current naNetworkAnalyzer application.</td>
</tr>
</tbody>
</table>

### ExternalTestsets History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IExternalTestsets</td>
<td>6.0</td>
</tr>
<tr>
<td>IExternalTestsets</td>
<td>6.2</td>
</tr>
</tbody>
</table>
FIFO Object

Description

These properties control the First IN, First OUT (FIFO) buffer settings for the PNA-X and N5264B.

The 4 GB FIFO data buffer is available with Option S93118A/B on the PNA-X and N5264B.

Accessing the FIFO object

```vba
Dim app as AgilentPNA835x.Application
Dim fifo as FIFO
Set fifo = app.FIFO
```

See Also:

- About FIFO
- FIFO example program
- VNA Automation Interfaces
- The VNA Object Model

### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear</td>
<td>IFIFO Clears the FIFO buffer</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Property</td>
<td>IFIFO</td>
<td>Reads the next specified number of data points from the FIFO buffer.</td>
</tr>
<tr>
<td>DataByteCount</td>
<td>IFIFO2</td>
<td>Reads the FIFO data byte count.</td>
</tr>
<tr>
<td>DataCount Property</td>
<td>IFIFO</td>
<td>Returns the total number of data points in the FIFO buffer.</td>
</tr>
<tr>
<td>DataAsBytes</td>
<td>IFIFO2</td>
<td>Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of bytes.</td>
</tr>
<tr>
<td>DataInCompactForm</td>
<td>IFIFO</td>
<td>Reads data from the FIFO buffer. Same as Data but in a compact form.</td>
</tr>
</tbody>
</table>
**DataAsFloat32**  
IFIFO2  
Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of 32-bit floating point (Float32) numbers.

**DataAsInt16**  
IFIFO2  
Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of 16-bit integers.

**DataAsInt32**  
IFIFO2  
Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of 32-bit integers.

**State**  
IFIFO  
Turns FIFO ON and OFF

### IFIFO History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFIFO</td>
<td>8.35</td>
</tr>
<tr>
<td>IFIFO2</td>
<td>12.80</td>
</tr>
</tbody>
</table>
Fixturing Object

Description

Contains the properties for Embedding and De-embedding test fixtures.

Accessing the Fixturing object

```vba
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim fixt as Fixturing
Set fixt = chan.Fxturing
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- About Fixturing
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoPortExtMeasure</td>
<td>IFxturing2 Measures either an OPEN or SHORT standard.</td>
</tr>
<tr>
<td>AutoPortExtReset</td>
<td>IFxturing2 Clears old port extension delay and loss data.</td>
</tr>
<tr>
<td>NetworkPortMap</td>
<td>IFxturing2 Set the port mapping for a 4-port SNP file to be embedded.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoPortExtConfig</td>
<td>IFxturing2</td>
<td>Sets the frequency span that is used to calculate Automatic Port Extension.</td>
</tr>
<tr>
<td>AutoPortExtDCOffset</td>
<td>IFxturing2</td>
<td>Specifies whether or not to include DC Offset as part of Automatic port extension.</td>
</tr>
<tr>
<td>AutoPortExtLoss</td>
<td>IFxturing2</td>
<td>Specifies whether or not to include loss correction as part of Automatic Port Extension.</td>
</tr>
<tr>
<td>AutoPortExtSearchStart</td>
<td>IFxturing2</td>
<td>Set the start frequency for custom user span.</td>
</tr>
</tbody>
</table>
AutoPortExtSearchStop  IFixturing2  Set the stop frequency for custom user span.
AutoPortExtState  IFixturing2  Enables and disables automatic port extensions on the specified port.
CmnModeZConvPortImag  IFixturing2  Sets imaginary value for common port impedance conversion.
CmnModeZConvPortReal  IFixturing2  Sets real value for common port impedance conversion.
CmnModeZConvState  IFixturing2  Turns ON/OFF common port impedance conversion.
CmnModeZConvPortZ0  IFixturing2  Sets impedance value for common port impedance conversion.
DiffPortMatch_C  IFixturing2  Sets Capacitance value of the differential matching circuit.
DiffPortMatch_G  IFixturing2  Sets Conductance value of the differential matching circuit.
DiffPortMatch_L  IFixturing2  Sets Inductance value of the differential matching circuit.
DiffPortMatch_R  IFixturing2  Sets Resistance value of the differential matching circuit.
DiffPortMatchMode  IFixturing2  Sets type of circuit to embed.
DiffPortMatchUserFilename  IFixturing2  Specifies the 4-port touchstone file for user-defined differential matching circuit.
DiffPortMatchState  IFixturing2  Turns ON/OFF differential matching circuit function.
DiffZConvPortImag  IFixturing2  Sets imaginary value for differential port impedance conversion.
DiffZConvPortReal  IFixturing2  Sets real value for differential port impedance conversion.
DiffZConvPortZ0  IFixturing2  Sets impedance value for differential port impedance conversion.
DiffZConvState  IFixturing2  Turns ON/OFF differential port impedance conversion.
Embed4PortA  IFixturing2  Returns VNA portA connections.
Embed4PortB  IFixturing2  Returns VNA portB connections.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embed4PortC</td>
<td>IFixturing2</td>
<td>Returns VNA portC connections.</td>
</tr>
<tr>
<td>Embed4PortD</td>
<td>IFixturing2</td>
<td>Returns VNA portD connections.</td>
</tr>
<tr>
<td>Embed4PortList</td>
<td>IFixturing2</td>
<td>Specifies all VNA port connections.</td>
</tr>
<tr>
<td>Embed4PortNetworkFilename</td>
<td>IFixturing2</td>
<td>Specifies *.s4p filename.</td>
</tr>
<tr>
<td>Embed4PortNetworkMode</td>
<td>IFixturing2</td>
<td>Specify embed, de-embed, or none.</td>
</tr>
<tr>
<td>Embed4PortState</td>
<td>IFixturing2</td>
<td>Turns ON or OFF 4-port Network Embed/De-embed.</td>
</tr>
<tr>
<td>Embed4PortTopology</td>
<td>IFixturing2</td>
<td>Specifies the VNA / DUT topology.</td>
</tr>
<tr>
<td>EnablePowerCompensation</td>
<td>IFixturing5</td>
<td>Compensates source power for combined loss through all fixturing functions.</td>
</tr>
<tr>
<td>EnableSnPDataExtrapolation</td>
<td>IFixturing6</td>
<td>Turns ON and OFF SNP file extrapolation for both 2-port and 4-port embedding/de-embedding.</td>
</tr>
<tr>
<td>FixturingState</td>
<td>IFixturing</td>
<td>Turns Fixturing ON and OFF on this channel.</td>
</tr>
<tr>
<td>NetworkPortMapA</td>
<td>IFixturing6</td>
<td>Read the port mapping of in A port for a 4-port SNP file to be embedded.</td>
</tr>
<tr>
<td>NetworkPortMapB</td>
<td>IFixturing6</td>
<td>Read the port mapping of in B port for a 4-port SNP file to be embedded.</td>
</tr>
<tr>
<td>NetworkPortMapC</td>
<td>IFixturing6</td>
<td>Read the port mapping of out A port for a 4-port SNP file to be embedded.</td>
</tr>
<tr>
<td>NetworkPortMapD</td>
<td>IFixturing6</td>
<td>Read the port mapping of out B port for a 4-port SNP file to be embedded.</td>
</tr>
<tr>
<td>Port2PdeembedCktModel</td>
<td>IFixturing</td>
<td>Sets and returns the 2 port De-embedding circuit model for the specified port number.</td>
</tr>
<tr>
<td>Port2PdeembedState</td>
<td>IFixturing</td>
<td>Turns 2 port de-embedding ON and OFF on this channel.</td>
</tr>
<tr>
<td>PortArbzImag</td>
<td>IFixturing3</td>
<td>Sets and returns the imaginary impedance value for the specified single-ended port number.</td>
</tr>
<tr>
<td>PortArbzReal</td>
<td>IFixturing3</td>
<td>Sets and returns the real impedance value for the specified single-ended port number.</td>
</tr>
<tr>
<td>PortArbzState</td>
<td>IFixturing</td>
<td>Turns single-ended port impedance ON and OFF on the specified channel.</td>
</tr>
<tr>
<td>PortArbzZ0</td>
<td>IFixturing3</td>
<td>Sets and returns the real and imaginary impedance value for the specified single-ended port number.</td>
</tr>
</tbody>
</table>
PortCoupleToSystemMedia IFixturing4 Couples to system Media type
PortCoupleToSystemVelocity IFixturing4 Couples to system Velocity Factor
PortDelay IFixturing Sets and returns the Port Delay value for the specified port number.
PortDistance IFixturing4 Sets Port Ext in distance
PortNADistanceUnit IFixturing4 Sets distance units
PortExtState IFixturing Turns Port Extension ON and OFF on this channel.
PortExtUse1 IFixturing Sets and returns the USE1 ON/OFF state for the Loss1 and Freq1 values for the specified port number.
PortExtUse2 IFixturing Sets and returns the USE2 ON/OFF state for the Loss2 and Freq2 values for the specified port number.
PortFreq1 IFixturing Sets and returns the 1st Port Frequency value for the specified port number.
PortFreq2 IFixturing Sets and returns the 2nd Port Frequency value for the specified port number.
PortLoss1 IFixturing Sets and returns the 1st Port Loss value for the specified port number.
PortLoss2 IFixturing Sets and returns the 2nd Port Loss value for the specified port number.
PortLossDC IFixturing Sets and returns the Port Loss at DC value for the specified port number.
PortMatching_C IFixturing7 Sets and returns the Capacitance, 'C' value for the specified port number.
PortMatching_G IFixturing7 Sets and returns the Conductance, 'G' value for the specified port number.
PortMatching_L IFixturing7 Sets and returns the Inductance, 'L' value for the specified port number.
PortMatching_R IFixturing7 Sets and returns the Resistance, 'R' value for the specified port number.
PortMatchingCktModel IFixturing Sets and returns the Port Matching circuit model for the specified port number.
PortMatchingState  IFixturing  Turns Port Matching ON and OFF on this channel.

PortMedium  IFixturing4  Sets Media per port

PortVelocityFactor  IFixturing4  Set Velocity Factor per port

PortWGCutoffFreq  IFixturing4  Sets waveguide cutoff frequency per port

Reverse2PortAdapter  IFixturing6  Set and read whether or not to reverse ports on a 2-port fixture or adapter to be de-embedded.

strPort2Pdeembed_S2PFile  IFixturing  Sets and returns the 2 port De-embedding 'S2P' file name for the specified port number.

strPortMatch_S2PFile  IFixturing  Sets and returns the Port Matching 'S2P' file name for the specified port number.

**IFixturing History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFixturing</td>
<td>5.0</td>
</tr>
<tr>
<td>IFixturing2</td>
<td>5.2</td>
</tr>
<tr>
<td>IFixturing3</td>
<td>5.25</td>
</tr>
<tr>
<td>IFixturing4</td>
<td>8.50</td>
</tr>
<tr>
<td>IFixturing5</td>
<td>9.20</td>
</tr>
<tr>
<td>IFixturing6</td>
<td>9.33</td>
</tr>
<tr>
<td>IFixturing7</td>
<td>12.70</td>
</tr>
</tbody>
</table>
FOM Collection

Description

The FOM collection provides access to the source and receiver range objects which are used for configuring frequency offset measurements.

The FOM range items are typically numbered as follows:

1. Primary
2. Source
3. Receivers
4. Source2 (if present)
5. Source3 (if present)

External devices can appear in the list of range names. Learn more.

Accessing the FOM Collection and FOMRange objects

Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel

Dim ifom as FOM
Set ifom = chan.FOM

ifom.item(2).Coupled = false

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- About FOM
- Example Programs
<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>IFOM</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DisplayRange</td>
<td>IFOM</td>
<td>Sets the range to be displayed on the VNA x-axis.</td>
</tr>
<tr>
<td>FOMRange</td>
<td>IFOM</td>
<td>Object</td>
</tr>
<tr>
<td>RangeCount</td>
<td>IFOM</td>
<td>Returns the number of FOM ranges available on the VNA.</td>
</tr>
<tr>
<td>State</td>
<td>IFOM</td>
<td>Turns Frequency Offset ON and OFF.</td>
</tr>
</tbody>
</table>

**FOM History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFOM</td>
<td>7.10</td>
</tr>
</tbody>
</table>
FOMRange Object

Description

The FOM Range object provides access to the properties and methods for configuring a specific Range for frequency offset measurements.

Accessing an FOMRange object

Get a handle to a FOM Range by specifying an item in the FOM collection.

The FOM range items are typically numbered as follows:

1. Primary
2. Source
3. Receivers
4. Source2 (if present)
5. Source3 (if present)

External devices can appear in the list of range names. Learn more.

```vba
Dim app as AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim ranges as FOM
Set ranges = app.ActiveChannel.FOM
ranges.item(2).Coupled = False
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- About FOM
- Example Programs
<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupled</td>
<td>IFOMRange</td>
<td>Sets and returns the state of coupling (ON or OFF) of this range to the primary range.</td>
</tr>
<tr>
<td>CWFrequency</td>
<td>IFOMRange</td>
<td>Set the Continuous Wave (CW) frequency.</td>
</tr>
<tr>
<td>Divisor</td>
<td>IFOMRange</td>
<td>Sets and returns the Divisor value to be used when coupling this range to the primary range.</td>
</tr>
<tr>
<td>Multiplier</td>
<td>IFOMRange</td>
<td>Sets and returns the Multiplier value to be used when coupling this range to the primary range.</td>
</tr>
<tr>
<td>Name</td>
<td>IFOMRange</td>
<td>Returns the name of this FOM range object.</td>
</tr>
<tr>
<td>Offset</td>
<td>IFOMRange</td>
<td>Sets and returns the offset value to be used when coupling this range to the primary range.</td>
</tr>
<tr>
<td>rangeNumber</td>
<td>IFOMRange</td>
<td>Returns the index number of the range within the FOM collection.</td>
</tr>
<tr>
<td>Segments</td>
<td>IFOMRange</td>
<td>Collection - Used to add segment sweep capability to a range.</td>
</tr>
<tr>
<td>StartFrequency</td>
<td>IFOMRange</td>
<td>Sets or returns the start frequency of this FOM Range.</td>
</tr>
<tr>
<td>StopFrequency</td>
<td>IFOMRange</td>
<td>Sets or returns the stop frequency of this FOM Range.</td>
</tr>
<tr>
<td>Sweep Type</td>
<td>IFOMRange</td>
<td>Sets the type of range sweep.</td>
</tr>
</tbody>
</table>

**Note:** Use the Start Power and Stop Power settings from the channel object.

### FOM History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFOM</td>
<td>7.10</td>
</tr>
</tbody>
</table>
### Gain Compression Measurement Object

**Description**

Controls the Gain Compression Analysis settings.

**Accessing the GainCompressionMeas object**

```vba
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset
app.ActiveMeasurement.Delete
' create a GCA measurement
app.CreateCustomMeasurementEx 1, "Gain Compression","CompIn21", 1
Set meass = app.Measurements ' get the measurements collection
Set ana = meass(1).CustomMeasurementConfiguration 'get the measurement
ana.AnalysisEnable = true ' enable the analysis mode
```

**See Also:**

- **Example Program** Create and Cal a Gain Compression Measurement (includes Analysis)
- GainCompression Object
- GainCompressionCal Object
- About Gain Compression Application
- VNA Automation Interfaces
- The VNA Object Model
### Method

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>See History</td>
</tr>
</tbody>
</table>

None

### Property

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AnalysisCWFreq</td>
<td>IGainCompressionMeas</td>
<td>Set CW frequency.</td>
</tr>
<tr>
<td>AnalysisEnable</td>
<td>IGainCompressionMeas</td>
<td>Enable a compression analysis trace.</td>
</tr>
<tr>
<td>AnalysisIsDiscreteFreq</td>
<td>IGainCompressionMeas</td>
<td>Set to discrete or interpolated CW frequencies.</td>
</tr>
<tr>
<td>AnalysisXAxis</td>
<td>IGainCompressionMeas</td>
<td>Sets X-axis display.</td>
</tr>
</tbody>
</table>

### IGainCompressionMeas History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGainCompressionMeas</td>
<td>9.0</td>
</tr>
</tbody>
</table>
Gain Compression Object

Description

Controls the Gain Compression Application settings.

Accessing the GainCompression object

```vbnet
Dim app as AgilentPNA835x.Application
app.CreateCustomMeasurementEx(1, "Gain Compression", "CompIn21", 1)
Dim GCA
Set GCA = app.ActiveChannel.CustomChannelConfiguration
```

See Also:

- Example Programs
  - Create and Cal a Gain Compression Measurement
  - Create and Cal a GCX Measurement
- GainCompressionCal Object
- About Gain Compression Application
- VNA Automation Interfaces
- The VNA Object Model

**Note:** Set the Start/Stop Frequency and Start/Stop Power Settings using the Channel Object.

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetRaw2DDData</td>
<td>IGainCompression</td>
<td>Reads Gain Compression data from specified location.</td>
</tr>
<tr>
<td>GetDataIm</td>
<td>IGainCompression</td>
<td>Reads Imaginary part of specified frequency or power points.</td>
</tr>
<tr>
<td>GetDataRe</td>
<td>IGainCompression</td>
<td>Reads REAL part of specified frequency or power points.</td>
</tr>
<tr>
<td>SetPortMap</td>
<td>IGainCompression</td>
<td>Maps the VNA ports to the DUT ports</td>
</tr>
<tr>
<td>Property</td>
<td>Interface</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AcquisitionMode</td>
<td>IGainCompression</td>
<td>Set and read the method by which gain compression data is acquired.</td>
</tr>
<tr>
<td>CompressionAlgorithm</td>
<td>IGainCompression</td>
<td>Set and read the algorithm method used to compute gain compression.</td>
</tr>
<tr>
<td>CompressionBackoff</td>
<td>IGainCompression</td>
<td>Set and read value for the BackOff compression algorithm.</td>
</tr>
<tr>
<td>CompressionDeltaX</td>
<td>IGainCompression</td>
<td>Set and read the 'X'' value in the delta X/Y compression algorithm.</td>
</tr>
<tr>
<td>CompressionDeltaY</td>
<td>IGainCompression</td>
<td>Set and read the 'Y'' value in the delta X/Y compression algorithm.</td>
</tr>
<tr>
<td>CompressionInterpolation</td>
<td>IGainCompression</td>
<td>Sets whether or not to interpolate the final power level when the measured compression level deviates from the specified level.</td>
</tr>
<tr>
<td>CompressionLevel</td>
<td>IGainCompression</td>
<td>Set and read the decrease in gain which indicates that the amplifier is compressing.</td>
</tr>
<tr>
<td>DeviceInputPort</td>
<td>IGainCompression</td>
<td>Read the VNA port number which is connected to the DUT input.</td>
</tr>
<tr>
<td>DeviceOutputPort</td>
<td>IGainCompression</td>
<td>Read the VNA port number which is connected to the DUT output.</td>
</tr>
<tr>
<td>EndOfSweepOperation</td>
<td>IGainCompression</td>
<td>Set and read the action which should be taken at the end of the last frequency or power sweep in the measurement.</td>
</tr>
<tr>
<td>InputLinearPowerLevel</td>
<td>IGainCompression</td>
<td>Set and read the input power level that should produce linear gain.</td>
</tr>
<tr>
<td>MaximumNumberOfPoints</td>
<td>IGainCompression</td>
<td>Returns the maximum possible number of data points.</td>
</tr>
<tr>
<td>NumberOfFrequencyPoints</td>
<td>IGainCompression</td>
<td>Set and read the number of data points in each frequency sweep.</td>
</tr>
<tr>
<td>NumberOfPowerPoints</td>
<td>IGainCompression</td>
<td>Set and read the number of data points in each power sweep.</td>
</tr>
<tr>
<td>ReadDCAtCompression</td>
<td>IGainCompression5</td>
<td>Set and read the DC meter at the compression point.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>ReverseLinearPowerLevel</td>
<td>Set and read the reverse power level to the DUT.</td>
<td></td>
</tr>
<tr>
<td>SafeSweepCoarsePowerAdjustment</td>
<td>Set and read the Safe Sweep COARSE power adjustment.</td>
<td></td>
</tr>
<tr>
<td>SafeSweepDCParameter</td>
<td>Set and read the name of the external DC device.</td>
<td></td>
</tr>
<tr>
<td>SafeSweepEnable</td>
<td>Set and read the (ON</td>
<td>OFF) state of Safe Sweep mode.</td>
</tr>
<tr>
<td>SafeSweepFinePowerAdjustment</td>
<td>Set and read the Safe Sweep FINE power adjustment.</td>
<td></td>
</tr>
<tr>
<td>SafeSweepFineThreshold</td>
<td>Set and read the compression level in which Safe Sweep changes from the COARSE power adjustment to the FINE power adjustment.</td>
<td></td>
</tr>
<tr>
<td>SafeSweepMaximumDCLimit</td>
<td>Set and read the maximum limit of the external DC device.</td>
<td></td>
</tr>
<tr>
<td>SafeSweepMaximumLimit</td>
<td>When the DUT Output reaches this value, the input power to the DUT is no longer incremented at that frequency.</td>
<td></td>
</tr>
<tr>
<td>SaturationLevel</td>
<td>Set and read the deviation dB from the maximum Pout.</td>
<td></td>
</tr>
<tr>
<td>SearchFailures</td>
<td>Read number of points that did not achieve compression.</td>
<td></td>
</tr>
<tr>
<td>SearchSummary</td>
<td>Returns if a compression search is complete, and if the search was a success or failure.</td>
<td></td>
</tr>
<tr>
<td>SmartSweepMaximumIterations</td>
<td>Set and read the maximum number of iterations to be used to find the compression level in a SMART sweep.</td>
<td></td>
</tr>
<tr>
<td>SmartSweepSettlingTime</td>
<td>Set and read SMART sweep settling time.</td>
<td></td>
</tr>
<tr>
<td>SmartSweepShowIterations</td>
<td>Set and read whether to show results for each SMART sweep iteration.</td>
<td></td>
</tr>
<tr>
<td>SmartSweepTolerance</td>
<td>Set and read the level of tolerance to be used to find the compression level in a SMART sweep.</td>
<td></td>
</tr>
</tbody>
</table>
TotalIterations  IGainCompression2  Returns the total number of iteration required for the SMART Sweep.

TotalNumberOfPoints  IGainCompression  Set and read the total number of data points.(Freq x Power)

**IGainCompression History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGainCompression</td>
<td>8.0</td>
</tr>
<tr>
<td>IGainCompression2</td>
<td>8.2</td>
</tr>
<tr>
<td>IGainCompression3</td>
<td>9.0</td>
</tr>
<tr>
<td>IGainCompression4</td>
<td>9.2</td>
</tr>
<tr>
<td>IGainCompression5</td>
<td>10.49</td>
</tr>
<tr>
<td>IGainCompression6</td>
<td>13.60</td>
</tr>
</tbody>
</table>
Gain CompressionCal Object

Description

Sets properties that are unique to a Gain Compression Cal (opt S93086A/B).

The remaining commands to perform a GCA Cal use the Guided Calibration commands.

Accessing the GainCompressionCal object

```vba
Dim app as AgilentPNA835x.Application
Set GCAcal = pna.GetCalmanager.CreateCustomCalEx(channelNum)
Set GCACalExtension = GCAcal.CustomCalConfiguration
GCACalExtension.PowerLevel = 5
```

See Also:

- Example Program Create and Cal a Gain Compression Measurement
- GainCompression Object
- About Gain Compression Application
- The VNA Object Model
- VNA Automation Interfaces

### Method

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>See History</td>
<td></td>
</tr>
</tbody>
</table>

None

### Property

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PowerLevel</td>
<td>IGainCompressionCal</td>
<td>Set and read the power level at which to perform the Source Power Cal.</td>
</tr>
<tr>
<td>PowerSensorCalKitType</td>
<td>IGainCompressionCal2</td>
<td>Set and read the cal kit to be used for calibrating at the port 1 reference plane.</td>
</tr>
<tr>
<td>PowerSensorConnectorType</td>
<td>IGainCompressionCal2</td>
<td>Superseded by PowerSensorConnectorType on the GuidedCal Object.</td>
</tr>
</tbody>
</table>
## IGainCompressionCal History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGainCompressionCal</td>
<td>8.0</td>
</tr>
<tr>
<td>IGainCompressionCal2</td>
<td>8.04</td>
</tr>
</tbody>
</table>
Gating Object

Description

Contains the methods and properties that control Time Domain Gating.

Accessing the Gating Object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim gate As Gating
Set gate = app.ActiveMeasurement.Gating
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Time Domain Topics
- Example Programs

Methods

None

Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center</td>
<td>IGating</td>
<td>Sets or returns the Center time.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Transform Object</td>
</tr>
<tr>
<td>CoupledParameters</td>
<td>IGating2</td>
<td>Select Gating parameters to couple</td>
</tr>
<tr>
<td>Shape</td>
<td>IGating</td>
<td>Specifies the shape of the gate filter.</td>
</tr>
<tr>
<td>Span</td>
<td>IGating</td>
<td>Sets or returns the Span time.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Transform Object</td>
</tr>
</tbody>
</table>
Start IGating Sets or returns the Start time.

Shared with the Transform Object

State IGating Turns an Object ON and OFF.

Stop IGating Sets or returns the Stop time.

Shared with the Transform Object

Type IGating Specifies the type of gate filter used.

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGating</td>
<td>1.0</td>
</tr>
<tr>
<td>IGating2</td>
<td>4.2</td>
</tr>
</tbody>
</table>
GlobalPowerLimit Object

Description

Provides access to the properties and methods for setting power limits for VNA ports.

Accessing the GlobalPowerLimit object

Dim app as AgilentPNA835x.Application
Dim gpl as IGlobalPowerLimit
Set gpl = app.GlobalPowerLimit

See Also:

- About GlobalPowerLimit
- VNA Automation Interfaces
- The VNA Object Model

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limit</td>
<td>IGlobalPowerLimit</td>
<td>Sets and returns the power limit for the specified port.</td>
</tr>
<tr>
<td>Lock</td>
<td>IGlobalPowerLimit</td>
<td>Enables or disables the ability to change the power limit values through the user interface.</td>
</tr>
<tr>
<td>State</td>
<td>IGlobalPowerLimit</td>
<td>Turns GlobalPowerLimit ON and OFF</td>
</tr>
</tbody>
</table>

IGlobalPowerLimit History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGlobalPowerLimit</td>
<td>9.0</td>
</tr>
</tbody>
</table>
GroupDelayAperture Object

Description
Contains the methods and properties that set Group Delay Aperture.

Accessing the GroupDelayAperture Object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")

Dim GDAperture As GroupDelayAperture
Set GDAperture = app.ActiveMeasurement.GroupDelayAperture

See Also:
- Group Delay Measurements
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

Methods

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>IGroupDelayAperture Sets group delay aperture using a fixed frequency range.</td>
</tr>
<tr>
<td>Percent</td>
<td>IGroupDelayAperture Sets group delay aperture using a percent of the channel frequency span.</td>
</tr>
<tr>
<td>Points</td>
<td>IGroupDelayAperture Sets group delay aperture using a fixed number of data points.</td>
</tr>
</tbody>
</table>

IGroupDelayAperture History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGroupDelayAperture</td>
<td>A.09.20</td>
</tr>
</tbody>
</table>
GuidedCalibration Object

Description

Contains the methods and properties used to perform a Guided Calibration.

**Important!**

Do NOT use commands from the Calibrator (Unguided calibration) object when performing a Guided calibration. Use ONLY the GuidedCalibration object.

The ONLY exception is:

Use `OrientECalModule_Property` and `ECalPortMapEx_Property` on the Calibrator Object to specify orientation for both guided and unguided calibrations.

A Guided Calibration must be performed on the Active Channel. To activate a channel, activate any measurement on that channel. Do this using `meas.Activate`, which requires you already have a handle to the measurement.

**Accessing the GuidedCalibration object**

For standard S-parameter channels:

```plaintext
Dim app as AgilentPNA835x.Application
Dim CalMgr
Set CalMgr = App.GetCalManager
Dim guidedCal
Set guidedCal = CalMgr.GuidedCalibration
```

To calibrate an Application channel, see `CreateCustomCalEx` Method.
**THRU Pairs sequence**

The Smart/Guided Cal logic always determines the best calibration based on your specified connectors and ports.

The following three commands overwrite the SmartCal logic. Send these commands ONLY if you have a deliberate reason for overwriting the SmartCal logic.

- ThruPortList
- PathThruMethod
- PathCalMethod

When sending one or more of these commands, they must be sent in the following sequence with the other commands listed here.

**Note:** The `Initialize` command is sent before and after these three commands.

1. ConnectorType (ports)
2. CalKitType (ports)
3. Initialize
4. ThruPortList
5. PathThruMethod
6. PathCalMethod
7. calMethod = guidedCal.PathCalMethod (ports) (recommended)
8. Initialize

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcquireStep</td>
<td>IGuidedCalibration</td>
</tr>
</tbody>
</table>
ApplyDeltaMatchFromCalSet IGuidedCalibration2 Apply a cal as Delta Match Cal.
GenerateErrorTerms IGuidedCalibration Generates the error terms for the calibration.
GenerateGlobalDeltaMatchSequence IGuidedCalibration2 Initiates a global delta match calibration.
GenerateSteps IGuidedCalibration Request to generate a connection list and return the number of steps required.
GetCompatibleCalKits IGuidedCalibration5 Returns the list of cal kits for the connector type.
GetIsolationPaths IGuidedCalibration3 Gets the list of port pairings for which isolation standards will be measured during calibration.
GetStepDescription IGuidedCalibration Query description of a step.
Initialize IGuidedCalibration Initial setup with channel context for the remote cal object.
ResetStep IGuidedCalibration10 Resets the specified guided cal connection step as unmeasured.
SetIsolationPaths IGuidedCalibration3 Sets the list of port pairings for which isolation standards will be measured during calibration.
SetupMeasurementsForStep IGuidedCalibration4 Show the Cal Window, or custom Cal Window, before acquiring a Cal standard.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalKitType</td>
<td>IGuidedCalibration</td>
<td>Sets the cal kit for the port.</td>
</tr>
<tr>
<td>CompatibleCalKits</td>
<td>IGuidedCalibration</td>
<td>Superseded with GetCompatibleCalKits Method</td>
</tr>
<tr>
<td>ConnectorType</td>
<td>IGuidedCalibration</td>
<td>Sets the connector type for the port.</td>
</tr>
<tr>
<td>CustomCalConfiguration</td>
<td>IGuidedCalibration4</td>
<td>Provides access to additional Properties and Methods which extends the GuidedCal Object.</td>
</tr>
</tbody>
</table>

See History
<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsolationAveragingIncrement</td>
<td>IGuidedCalibration3</td>
<td>Value by which to increment the channel's averaging factor during measurement of isolation standards.</td>
</tr>
<tr>
<td>IterationCountForStep</td>
<td>IGuidedCalibration10</td>
<td>Returns the number of iterative measurement acquisitions that has been made for the specified step.</td>
</tr>
<tr>
<td>MinimumIterationsForStep</td>
<td>IGuidedCalibration10</td>
<td>Returns the minimum number of required iterative measurement acquisitions for the specified step.</td>
</tr>
<tr>
<td>PathCalMethod</td>
<td>IGuidedCalibration3</td>
<td>Specifies the calibration method for each port pair.</td>
</tr>
<tr>
<td>PathThruMethod</td>
<td>IGuidedCalibration3</td>
<td>Specifies the calibration <strong>THRU</strong> method for each port pair.</td>
</tr>
<tr>
<td>PerformPowerCalibration</td>
<td>IGuidedCalibration7</td>
<td>Perform Guided Power Cal.</td>
</tr>
<tr>
<td>PortsNeedingDeltaMatch</td>
<td>IGuidedCalibration2</td>
<td>Returns port numbers that need delta match cal.</td>
</tr>
<tr>
<td>PowerCalibrationPowerLevel</td>
<td>IGuidedCalibration6</td>
<td>Sets power level for power cal in several applications.</td>
</tr>
<tr>
<td>PowerSensorCalKitType</td>
<td>IGuidedCalibration6</td>
<td>Sets Cal Kit for power cal in several applications.</td>
</tr>
<tr>
<td>PowerSensorConnectorType</td>
<td>IGuidedCalibration6</td>
<td>Sets Power sensor connector for power cal in several applications.</td>
</tr>
<tr>
<td>PowerTableFilename</td>
<td>IGuidedCalibration9</td>
<td>Loads a file that defines a power table.</td>
</tr>
<tr>
<td>SlidingLoadAcquisitionBehavior</td>
<td>IGuidedCalibration10</td>
<td>Specifies the behavior for guided cal steps that involve a sliding load.</td>
</tr>
<tr>
<td>ThruCalMethod</td>
<td>IGuidedCalibration</td>
<td><strong>Superseded with</strong> PathCalMethod and PathThruMethod</td>
</tr>
<tr>
<td>ThruPortList</td>
<td>IGuidedCalibration</td>
<td>Sets the thru connection port pairs.</td>
</tr>
<tr>
<td>UncertaintyEnabled</td>
<td>IGuidedCalibration11</td>
<td>Enables Dynamic Uncertainty.</td>
</tr>
<tr>
<td>UseCalWindow</td>
<td>IGuidedCalibration</td>
<td><strong>Obsolete</strong> - Use Custom Cal Window commands.</td>
</tr>
<tr>
<td>ValidConnectorTypes</td>
<td>IGuidedCalibration</td>
<td>Gets Valid Connector Types.</td>
</tr>
</tbody>
</table>
## IGguidedCalibration History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGguidedCalibration</td>
<td>5.0</td>
</tr>
<tr>
<td>IGguidedCalibration2</td>
<td>5.25</td>
</tr>
<tr>
<td>IGguidedCalibration3</td>
<td>7.11</td>
</tr>
<tr>
<td>IGguidedCalibration4</td>
<td>8.0</td>
</tr>
<tr>
<td>IGguidedCalibration5</td>
<td>9.0</td>
</tr>
<tr>
<td>IGguidedCalibration6</td>
<td>9.10</td>
</tr>
<tr>
<td>IGguidedCalibration7</td>
<td>9.30</td>
</tr>
<tr>
<td>IGguidedCalibration8</td>
<td></td>
</tr>
<tr>
<td>IGguidedCalibration9</td>
<td>9.42</td>
</tr>
<tr>
<td>IGguidedCalibration10</td>
<td>9.85</td>
</tr>
<tr>
<td>IGguidedCalibration11</td>
<td>10.40</td>
</tr>
</tbody>
</table>
GuidedCalibrationPowerSensor Object

Description

Contains the methods and properties to configure multiple power sensors to be used during a guided power calibration.

Note: These commands are supported ONLY on standard channels.

Accessing the GuidedCalibrationPowerSensor Object

A GuidedCalibrationPowerSensor object is accessed as an item of the GuidedCalibrationPowerSensors Collection.

```vba
Option explicit
Dim app as AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")
Dim CalMgr
Set CalMgr = App.GetCalManager
Dim guidedCal
Set guidedCal = CalMgr.GuidedCalibration
Dim powerSensors
Dim port: port = 1
Set powerSensors = guidedCal.GuidedCalibrationPowerSensors(port)
Dim powerSensor2
Set powerSensor2 = GuidedCalibrationPowerSensors.Item(2)
powerSensor2.Name="26GHzSensor"
```

See Also:

- GuidedCalibrationPowerSensors Collection
- See Example Programs
  - Perform a Guided Power Cal using Multiple Power Sensors
  - Create a PMAR Device and Measurement
- VNA Automation Interfaces
- The VNA Object Model

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>IGuidedCalibrationPowerSensors</td>
<td>Sets and returns the name of this power sensor.</td>
</tr>
<tr>
<td>PowerSensorCalKitType</td>
<td>IGuidedCalibrationPowerSensors</td>
<td>Set and read the cal kit to be used for calibrating at the reference plane.</td>
</tr>
<tr>
<td>PowerSensorConnectorType</td>
<td>IGuidedCalibrationPowerSensors</td>
<td>Sets or returns the connector type of the power sensor.</td>
</tr>
<tr>
<td>StartFrequency</td>
<td>IGuidedCalibrationPowerSensors</td>
<td>Sets or returns the start frequency of the power sensor coverage.</td>
</tr>
<tr>
<td>StopFrequency</td>
<td>IGuidedCalibrationPowerSensors</td>
<td>Sets or returns the stop frequency of the power sensor coverage.</td>
</tr>
</tbody>
</table>

IGuidedCalibrationPowerSensors History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGuidedCalibrationPowerSensors</td>
<td>9.33</td>
</tr>
</tbody>
</table>
GuidedCalibrationPowerSensors Collection

Description

Contains the methods and properties to enable and configure multiple power sensors to be used during a guided calibration.

Note: Guided Power Cal, and "multiple sensors" are allowed ONLY on standard channels.

Accessing the GuidedCalibrationPowerSensors Collection

```
Option explicit

Dim app as AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")

Dim CalMgr
Set CalMgr = App.GetCalManager

Dim guidedCal
Set guidedCal = CalMgr.GuidedCalibration

Dim port: port = 1

Dim powerSensors
Set powerSensors = guidedCal.GuidedCalibrationPowerSensors(port)

powerSensors = powerSensors.Add "26GHzPowerSensor"
```

See Also:

- GuidedCalibrationPowerSensor Object
- VNA Automation Interfaces
- The VNA Object Model
- See Example Program
### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>IGuidedCalibrationPowerSensors</td>
<td>Add a power sensor by name to the collection.</td>
</tr>
<tr>
<td>Item</td>
<td>IGuidedCalibrationPowerSensors</td>
<td>Specify a power sensor by name in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>IGuidedCalibrationPowerSensors</td>
<td>Remove a power sensor from the collection.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>IGuidedCalibrationPowerSensors</td>
<td>Return the number of power sensors in the collection.</td>
</tr>
<tr>
<td>UseMultipleSensors</td>
<td>IGuidedCalibrationPowerSensors</td>
<td>Enable the use of multiple power sensors.</td>
</tr>
</tbody>
</table>

### IGuidedCalibrationPowerSensors History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGuidedCalibrationPowerSensors</td>
<td>9.33</td>
</tr>
</tbody>
</table>
HWAuxIO Object

Description

Contains the methods and properties that control the rear panel Material Handler I/O and Power I/O connector.

Note: The Aux I/O connector is mentioned in these topics, but NO VNA models supported by this VNA firmware have that connector.

Accessing the HWAuxIO object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim AuxIO As HWAuxIO
Set AuxIO = app.GetAuxIO
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_InputVoltage</td>
<td>IHWAuxIO</td>
<td>Superseded by <code>get InputVoltageEX</code></td>
</tr>
<tr>
<td>get_OutputVoltage</td>
<td>IHWAuxIO</td>
<td>Reads ADC output voltages.</td>
</tr>
<tr>
<td>get_OutputVoltageMode</td>
<td>IHWAuxIO2</td>
<td>Reads mode setting for either DAC output.</td>
</tr>
<tr>
<td>get_PortCData</td>
<td>IHWAuxIO</td>
<td>Reads a 4-bit value from Port C</td>
</tr>
<tr>
<td>put_OutputVoltage</td>
<td>IHWAuxIO</td>
<td>Writes voltages to the DAC/Analog Output 1 and Output 2</td>
</tr>
<tr>
<td>put_OutputVoltageMode</td>
<td>IHWAuxIO2</td>
<td>Writes the mode setting for either DAC output.</td>
</tr>
<tr>
<td>put_PortCData</td>
<td>IHWAuxIO</td>
<td>Writes a 4-bit value to Port C</td>
</tr>
<tr>
<td>Parameter</td>
<td>Interface</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FootSwitch</td>
<td>IHWAuxIO</td>
<td><strong>Obsolete</strong> - Reads the Footswitch Input</td>
</tr>
<tr>
<td>FootswitchMode</td>
<td>IHWAuxIO3</td>
<td><strong>Obsolete</strong> - Determines the action that occurs when the footswitch is pressed.</td>
</tr>
<tr>
<td>InputVoltageEX</td>
<td>IHWAuxIO5</td>
<td>Reads the ADC input voltage</td>
</tr>
<tr>
<td>PassFailLogic</td>
<td>IHWAuxIO</td>
<td>Sets and reads the logic of the PassFail line</td>
</tr>
<tr>
<td>PassFailMode</td>
<td>IHWAuxIO</td>
<td>Sets and reads the mode of the PassFail line</td>
</tr>
<tr>
<td>PassFailPolicy</td>
<td>IHWAuxIO4</td>
<td>Sets the policy used to determine how global pass/fail is computed.</td>
</tr>
<tr>
<td>PassFailScope</td>
<td>IHWAuxIO</td>
<td>Sets and reads the scope of the PassFail line</td>
</tr>
<tr>
<td>PassFailStatus</td>
<td>IHWAuxIO4</td>
<td>Returns the most recent pass/fail status value</td>
</tr>
<tr>
<td>PortCLogic</td>
<td>HWAuxIO</td>
<td>Sets and reads the logic mode of Port C</td>
</tr>
<tr>
<td>PortCMode</td>
<td>HWAuxIO</td>
<td>Sets and reads the mode of Port C</td>
</tr>
<tr>
<td>SweepEndMode</td>
<td>HWAuxIO</td>
<td>Sets and reads the event that causes the Sweep End line to go to a false state.</td>
</tr>
</tbody>
</table>

**IHWAuxIO History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev</th>
</tr>
</thead>
<tbody>
<tr>
<td>IHWAuxIO</td>
<td>2.0</td>
</tr>
<tr>
<td>IHWAuxIO2</td>
<td>3.0</td>
</tr>
<tr>
<td>IHWAuxIO3</td>
<td>3.0</td>
</tr>
<tr>
<td>IHWAuxIO4</td>
<td>5.0</td>
</tr>
<tr>
<td>IHWAuxIO5</td>
<td>7.5</td>
</tr>
</tbody>
</table>
**HWExternalTestSetIO Object**

**Description**

Contains the methods and properties that control the rear panel External Test Set Input / Output connector

**Accessing the HWExternalTestSetIO object**

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim ExtTS As HWExternalTestSetIO
Set ExtTS = app.GetExternalTestSetIO
```

**See Also:**

- Pinout for the External Test Set Connector
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadData</td>
<td>Reads data and generates the appropriate timing signals</td>
</tr>
<tr>
<td>ReadRaw</td>
<td>Reads data, but does NOT generate appropriate timing signals</td>
</tr>
<tr>
<td>WriteData</td>
<td>Writes data and generates the appropriate timing signals</td>
</tr>
<tr>
<td>WriteRaw</td>
<td>Writes data, but does NOT generate the appropriate timing signals</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interrupt</td>
<td>Returns the state of the Interrupt line</td>
</tr>
<tr>
<td>SweepHoldOff</td>
<td>Returns the state of the Sweep Holdoff line</td>
</tr>
</tbody>
</table>

**IHWEexternalTestSetIO History**
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>HWExternalTestSetIO</td>
<td>2.0</td>
</tr>
</tbody>
</table>
HWMaterialHandlerIO Object

Description

Contains the methods and properties that control the rear panel Material Handler Input / Output connector.

Accessing the HWMaterialHandlerIO object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim MatHdlr As HWMaterialHandlerIO
Set MatHdlr = app.GetMaterialHandlerIO
```

See Also:

- Pinout for the Material HandlerIO Connector
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_Input1</td>
<td>HWMaterialHandlerIO</td>
<td>Reads a hardware latch that captures low to high transition on Input1</td>
</tr>
<tr>
<td>get_Output</td>
<td>HWMaterialHandlerIO</td>
<td>Returns the last value written to the selected output pin.</td>
</tr>
<tr>
<td>get_Port</td>
<td>HWMaterialHandlerIO</td>
<td>Returns the value from the specified &quot;readable&quot; port.</td>
</tr>
<tr>
<td>put_Output</td>
<td>HWMaterialHandlerIO</td>
<td>Writes a TTL HI or TTL Low to output pins 3 or 4.</td>
</tr>
<tr>
<td>put_Port</td>
<td>HWMaterialHandlerIO</td>
<td>Writes a value to the specified port.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IndexState</td>
<td>HWMaterialHandlerIO02</td>
</tr>
</tbody>
</table>
ReadyForTriggerState \textbf{HWMaterialHandlerIO2} Determines the control of Material Handler connector Pin 21

PassFailLogic \textbf{HWMaterialHandlerIO} Sets and reads the logic of the PassFail line

PassFailMode \textbf{HWMaterialHandlerIO} Sets and reads the mode for the PassFail line

PassFailPolicy \textbf{HWMaterialHandlerIO2} Sets the policy used to determine how global pass/fail is computed.

PassFailScope \textbf{HWMaterialHandlerIO} Sets and reads the scope for the PassFail line

PassFailStatus \textbf{HWMaterialHandlerIO2} Returns the most recent pass/fail status value.

PortLogic \textbf{HWMaterialHandlerIO} Sets and returns the logic mode of data ports A-H

PortMode \textbf{HWMaterialHandlerIO} Sets and returns whether Port C or Port D is used for writing or reading data

SweepEndMode \textbf{HWMaterialHandlerIO} Sets and reads the event that cause the Sweep End line to go to a low state.

\textbf{HWMaterialHandlerIO History}

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>HWMaterialHandlerIO</td>
<td>2.0</td>
</tr>
<tr>
<td>HWMaterialHandlerIO2</td>
<td>5.0</td>
</tr>
</tbody>
</table>
Description

These properties control the IF gain and source path settings for the PNA-X (IFConfiguration3 commands ONLY).

Accessing the IFConfiguration object

```vbnet
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim cfg as IIFConfiguration
Set cfg = chan.IFConfiguration
```

See Also

- IF Path Configuration
- SignalProcessingModuleFour Object (PNA-X ONLY)
- PulseGenerator Object (PNA-X ONLY)
- The VNA Object Model
- Pulsed Application

Methods

None

Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFFrequency</td>
<td>IFConfiguration3</td>
<td>Sets IF frequency in manual mode.</td>
</tr>
<tr>
<td>IFFrequencyMode</td>
<td>IFConfiguration3</td>
<td>Sets IF frequency mode to automatic or manual.</td>
</tr>
<tr>
<td>MaximumIFFrequency</td>
<td>IFConfiguration3</td>
<td>Returns the maximum IF frequency setting</td>
</tr>
<tr>
<td>MinimumIFFrequency</td>
<td>IFConfiguration3</td>
<td>Returns the minimum IF frequency setting</td>
</tr>
</tbody>
</table>

IFConfiguration History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IIFConfiguration</td>
<td>4.0</td>
</tr>
<tr>
<td>IIFConfiguration2</td>
<td>4.0</td>
</tr>
<tr>
<td>IIFConfiguration3</td>
<td>7.2</td>
</tr>
</tbody>
</table>
**IMixer Interface (Option S93083A/B) - Superseded**

**Note:** This object and all properties and methods are replaced with Converter Object which can be used for all converter application.

**Description**

Contains the methods and properties to setup FCA Mixer measurements.

For performing calibrations, use either the SMC Type Object or the VMC Type Object.

**Accessing the IMixer Interface**

Access the IMixer Interface through the Measurement Object. If the particular type of Measurement that was created supports IMixer, then the program determines this at run time and can access the functionality exposed by IMixer. Because the determination of IMixer support is not made until runtime, the program should handle the case where IMixer is not supported on the object.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", "analyzerName")
app.Preset

' FCA Measurements can't share the channel with standard measurements
' Because preset creates a single measurement in channel 1, we first delete the standard measurement
Dim standardMeas As IMeasurement
Set standardMeas = app.ActiveMeasurement
standardMeas.Delete

' Create a Measurement object, in this case using the IMeasurement interface
Dim meas As IMeasurement
Set meas = app.CreateCustomMeasurementEx(1, "SMC_Forward.SMC_ForwardMeas", "SC21")

' See if this measurement object supports IMixer
Dim mixer As IMixer
Set mixer = meas
```

**See Also:**

VNA Automation Interfaces

The VNA Object Model

Example: How to create and calibrate a standard SMC or VMC measurement or a fixed output SMC measurement.
<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply</td>
<td>IMixer3</td>
<td>Applies mixer settings.</td>
</tr>
<tr>
<td>Calculate</td>
<td>IMixer</td>
<td>Automatically calculate Input and Output frequencies for mixer setup.</td>
</tr>
<tr>
<td>LoadFile</td>
<td>IMixer</td>
<td>Loads a previously-configured mixer attributes file (.mfr)</td>
</tr>
<tr>
<td>SaveFile</td>
<td>IMixer</td>
<td>Saves the settings for the mixer/ converter test setup to a mixer attributes file.</td>
</tr>
<tr>
<td>SetDutPorts</td>
<td>IMixer8</td>
<td>Sets the VNA to DUT port mapping.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveXAxisRange</td>
<td>IMixer3 Sets or returns the swept parameter to display on the X-axis.</td>
</tr>
<tr>
<td>AvoidSpurs</td>
<td>IMixer Sets and returns the state of the avoid spurs feature.</td>
</tr>
<tr>
<td>EmbeddedLO</td>
<td>IMixer7 Provides measurements of mixers with an embedded LO.</td>
</tr>
<tr>
<td>DeviceInputPort</td>
<td>IMixer8 Reads the VNA port that is mapped to the DUT Input port</td>
</tr>
<tr>
<td>DeviceOutputPort</td>
<td>IMixer8 Reads the VNA port that is mapped to the DUT Output port</td>
</tr>
<tr>
<td>EnablePhase</td>
<td>IMixer12 Include phase in SMC measurements and calibrations.</td>
</tr>
<tr>
<td>IFSideband</td>
<td>IMixer Sets or returns the value of the IF sideband.</td>
</tr>
<tr>
<td>IFStartFrequency</td>
<td>IMixer Returns the start frequency of the mixer IF.</td>
</tr>
<tr>
<td>IFStopFrequency</td>
<td>IMixer Returns the stop frequency of the mixer IF.</td>
</tr>
<tr>
<td>IncludeReverseSweep</td>
<td>IMixer12 Sets or returns whether to include SC12 sweep.</td>
</tr>
<tr>
<td>InputDenominator</td>
<td>IMixer Sets or returns the denominator value of the Input Fractional Multiplier.</td>
</tr>
</tbody>
</table>
InputFixedFrequency IMixer6 Sets or returns the mixer fixed Input frequency value.

InputNumerator IMixer Sets or returns the numerator value of the Input Fractional Multiplier.

InputPower IMixer Sets or returns the value of the Input Power.

InputRangeMode IMixer6 Sets or returns the Input sweep mode.

InputStartFrequency IMixer Sets or returns the start frequency of the mixer input.

InputStopFrequency IMixer Sets or returns the stop frequency of the mixer input.

IsInputGreaterThanLO IMixer2 Specifies whether to use the Input frequency that is greater than the LO or less than the LO.

LODenominator IMixer Sets or returns the denominator value of the LO Fractional Multiplier.

LOFixedFrequency IMixer Sets or returns the fixed frequency of the specified LO.

LOName IMixer Sets or returns the LO name.

LONumerator IMixer Sets or returns the numerator value of the LO Fractional Multiplier.

LOPower IMixer Sets or returns the value of the LO Power.

LORangeMode IMixer3 Sets or returns the LO sweep mode to fixed or swept.

LOStage IMixer Returns the number of LOs (1 or 2).

LOStartFrequency IMixer3 Sets or returns the start frequency of the specified LO.

LOStopFrequency IMixer3 Sets or returns the start frequency of the specified LO.

NominalIncidentPowerState IMixer4 Toggles Nominal Incident Power ON and OFF.

NormalizePoint IMixer12 Set or return the data point used to normalize SMC phase measurements.

OutputFixedFrequency IMixer3 Sets or returns the fixed frequency of the mixer output.
OutputRangeMode  **IMixer6**  Sets or returns the Output sweep mode.

OutputSideband  **IMixer**  Sets or returns the value of the output sideband.

OutputStartFrequency  **IMixer**  Sets or returns the start frequency of the mixer output.

OutputStopFrequency  **IMixer**  Sets or returns the stop frequency of the mixer output.

**IMixer History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMixer</td>
<td>1.0</td>
</tr>
<tr>
<td>IMixer2</td>
<td>3.5</td>
</tr>
<tr>
<td>IMixer3</td>
<td>4.0</td>
</tr>
<tr>
<td>IMixer4</td>
<td>4.8</td>
</tr>
<tr>
<td>IMixer5</td>
<td>6.04</td>
</tr>
<tr>
<td>IMixer6</td>
<td>6.20</td>
</tr>
<tr>
<td>IMixer7</td>
<td>7.21</td>
</tr>
<tr>
<td>IMixer8</td>
<td>7.5/8.2</td>
</tr>
<tr>
<td>IMixer12</td>
<td>7.5/9.2</td>
</tr>
</tbody>
</table>
**IM Spectrum Object**

**Description**

Controls the IM Spectrum settings.

**Accessing the IM Spectrum object**

```vba
Dim app as AgilentPNA835x.Application
app.CreateCustomMeasurementEx(1, "IM Spectrum", "Output", 1)
Dim IMSpec
Set IMSpec = app.ActiveChannel.CustomChannelConfiguration
```

**See Also:**

- **Example Program** Create and Cal an IM Spectrum Measurement
- **About IM Spectrum Measurements**
- **SweptIMD Object**
- **VNA Automation Interfaces**
- **The VNA Object Model**

**Method**  
<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetPortMap</td>
<td>IIMSpectrum</td>
</tr>
</tbody>
</table>

**Property**  
<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoupleTonePower</td>
<td>IIMSpectrum</td>
</tr>
<tr>
<td>DeviceInputPort</td>
<td>IIMSpectrum</td>
</tr>
<tr>
<td>DeviceOutputPort</td>
<td>IIMSpectrum</td>
</tr>
<tr>
<td>DeltaFrequency</td>
<td>IIMSpectrum</td>
</tr>
<tr>
<td>EqualTonePower</td>
<td>IIMSpectrum2</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>F1Frequency</td>
<td>Frequency of the F1 tone.</td>
</tr>
<tr>
<td>F2Frequency</td>
<td>Frequency of the F2 tone.</td>
</tr>
<tr>
<td>FrequencyCenter</td>
<td>Center frequency of the main tones.</td>
</tr>
<tr>
<td>LevelingMethod</td>
<td>Set tone power leveling mode</td>
</tr>
<tr>
<td>ResolutionBW</td>
<td>Resolution Bandwidth for the measurement.</td>
</tr>
<tr>
<td>SpectrumCenterFrequency</td>
<td>Receiver Center frequency</td>
</tr>
<tr>
<td>SpectrumSpanFrequency</td>
<td>Receiver frequency span.</td>
</tr>
<tr>
<td>SpectrumStartFrequency</td>
<td>Receiver Start frequency.</td>
</tr>
<tr>
<td>SpectrumStopFrequency</td>
<td>Receiver Stop frequency.</td>
</tr>
<tr>
<td>TonePowerSetAt</td>
<td>Set power at DUT input or Output.</td>
</tr>
<tr>
<td></td>
<td><strong>Superseded with LevelingMethod</strong></td>
</tr>
<tr>
<td>TrackingChannel</td>
<td>IMD channel number to which the IM Spectrum channel is coupled.</td>
</tr>
<tr>
<td>TrackingEnable</td>
<td>Enables tracking with an IMD channel.</td>
</tr>
<tr>
<td>TrackingManualStepEnable</td>
<td>Step sweep mode for the IM Spectrum channel.</td>
</tr>
<tr>
<td>TrackingStepIndex</td>
<td>IMD data point number at which the IM spectrum measurement occurs.</td>
</tr>
<tr>
<td>SweepOrder</td>
<td>IM product to view when SweepType = NTH is specified.</td>
</tr>
<tr>
<td>SweepType</td>
<td>Type of sweep for an IMSpectrum measurement.</td>
</tr>
<tr>
<td>TonePower</td>
<td>Power level of the Main Tones.</td>
</tr>
</tbody>
</table>

### IIMSpectrum History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IIMSpectrum</td>
<td>8.35</td>
</tr>
<tr>
<td>IIMSpectrum2</td>
<td>9.40</td>
</tr>
</tbody>
</table>
IndependentPowerCalibration Object

Description

Contains the methods and properties used to perform an independent power calibration.

Accessing the IndependentPowerCalibration object

For standard S-parameter channels:

```vbnet
Dim app as AgilentPNA835x.Application
Dim CalMgr
Set CalMgr = App.GetCalManager
Dim CalAll
Set CalAll = CalMgr.CalibrateAllChannels

Dim IndependentPwrCal
Set IndependentPwrCal = CalAll.IndependentPowerCalibration
```

To calibrate an Application channel, see CreateCustomCalEx Method.

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

### Methods

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IIndependentPowerCalibration</td>
<td>Use to get a handle to a IndependentPowerCalibrationPort object.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>See History</td>
<td></td>
</tr>
<tr>
<td>IIndependentPowerCalibration</td>
<td>Queries available ports for independent power calibration.</td>
</tr>
<tr>
<td>Interface</td>
<td>Introduced with PNA Rev:</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>IndependentPowerCalibration</td>
<td>10.60</td>
</tr>
</tbody>
</table>


IndependentPowerCalibrationPort Object

Description

Contains the methods and properties used to perform an independent power calibration on a specific source port.

Accessing the IndependentPowerCalibrationPort object

```vba
Dim app as AgilentPNA835x.Application
Dim CalMgr
Set CalMgr = App.GetCalManager
Dim CalAll
Set CalAll = CalMgr.CalibrateAllChannels

Dim IndependentPwrCalPort
Set IndependentPwrCalPort = CalAll.IndependentPowerCalibrationPort
```

See Also:

- Collections in the Analyzer
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddPowerCalRange</td>
<td>IIndependentPowerCalibrationPort Adds a power cal range for a specific port &lt;n&gt;.</td>
</tr>
<tr>
<td>Reset</td>
<td>IIndependentPowerCalibrationPort Resets all ranges for the given source port &lt;n&gt;.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PowerCalRange</td>
<td>IIndependentPowerCalibrationPort</td>
<td>Returns a handle to the PowerCalRange object.</td>
</tr>
</tbody>
</table>
IndependentPowerCalibrationPort Queries how many ranges are included in the calibration for source port <n>.

### IndependentPowerCalibrationPort History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IndependentPowerCalibrationPort</td>
<td>12.90</td>
</tr>
</tbody>
</table>
InterfaceControl Object

Description
Contains the methods and properties that support Interface Control.

Accessing the InterfaceControl object

```vbnet
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim IntControl As InterfaceControl
Set IntControl = app.InterfaceControl
```

See Also:
- VNA Automation Interfaces
- The VNA Object Model
- Interface Control Feature
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
</table>
| ConfigurationFile | InterfaceControl               | Recalls an Interface Control file
| State      | InterfaceControl               | Turns Interface Control ON and OFF

InterfaceControl History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>InterfaceControl</td>
<td>5.2</td>
</tr>
</tbody>
</table>
IOConfiguration Object

Description

Contains the methods and properties that support IO Configuration.

Accessing the IOConfiguration object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim IOConfig As IOConfiguration
Set IOConfig = app.IOConfiguration
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- IO Configuration topic
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HiSLIPAddress</td>
<td>IOConfiguration</td>
</tr>
<tr>
<td>HiSLIPPort</td>
<td>IOConfiguration</td>
</tr>
<tr>
<td>InterfaceTypes</td>
<td>IOConfiguration</td>
</tr>
<tr>
<td>VisaResourceString</td>
<td>IOConfiguration</td>
</tr>
</tbody>
</table>

IOConfiguration History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IIOConfiguration</td>
<td>10.0</td>
</tr>
</tbody>
</table>
Limit Test Collection

Description

Child of the Measurement Object. A collection that provides a mechanism for iterating through the Measurement's Limit Segment objects (Limit Lines). The collection has 100 limit lines by default.

Accessing the LimitTest collection

Get a handle to an individual limit segment by specifying an item of the LimitTest collection.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim limSegs As LimitTest
Set limSegs = app.ActiveMeasurement.LimitTest
limSegs.Item(1).BeginStimulus = 1000000000
limSegs.Item(1).EndStimulus = 1000000000
limSegs.Item(1).BeginResponse = 3.5
limSegs.Item(1).EndResponse = 3.5
```

See Also:

- LimitSegment Object
- Collections in the Analyzer
- The VNA Object Model
- Limit Line Testing Example

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetTestResult</td>
<td>Retrieves the Pass/Fail results of the Limit Test (State).</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle on a limit line in the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of limit lines used in the measurement.</td>
</tr>
<tr>
<td>LineDisplay</td>
<td>Displays the limit lines on the screen.</td>
</tr>
<tr>
<td>SegmentCount</td>
<td>Returns the number of segments used in a limit test.</td>
</tr>
<tr>
<td>SoundOnFail</td>
<td>Enables a beep on Limit Test fails.</td>
</tr>
</tbody>
</table>
State

Turns ON and OFF limit testing.

LimitTest History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILimitTest</td>
<td>1.0</td>
</tr>
</tbody>
</table>
LimitSegment Object

Description

The LimitSegment object is an individual limit line.

Accessing the LimitSegment object

Get a handle to an individual limit line by using the LimitTest collection.

```vbc
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim limSegs As LimitTest
Set limSegs = app.ActiveMeasurement.LimitTest
limSegs(1).BeginResponse = 1000000000#
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

Methods

None

Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BeginResponse</td>
<td>Specifies the Y-axis value that corresponds with Begin Stimulus (X-axis) value.</td>
</tr>
<tr>
<td>BeginStimulus</td>
<td>Specifies the beginning X-axis value of the Limit Line.</td>
</tr>
<tr>
<td>EndResponse</td>
<td>Specifies the Y-axis value that corresponds with End Stimulus (X-axis) value.</td>
</tr>
<tr>
<td>EndStimulus</td>
<td>Specifies the End X-axis value of the Limit Line.</td>
</tr>
<tr>
<td>Type</td>
<td>Specifies the Limit Line type.</td>
</tr>
</tbody>
</table>

LimitSegment History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILimitSegment</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Marker Object

Description
Contains the methods and properties that control Markers.

There are two ways to control markers through COM.

1. The Measurement object has properties that apply to ALL of the markers for that measurement.
2. The properties on the Marker object override the Measurement object properties. For example, you can then override the format setting for an individual marker by specifying `mark.Format = naLogMag` on the marker object.

Marker-related commands on the Measurement Object:

- Activate Marker Method
- DeleteAllMarkers Method
- DeleteMarker Method
- GetReferenceMarker Method
- InterpolateMarkers Method
- SearchFilterBandwidth Method
- ActiveMarker Property
- Marker Format Property
- MarkerState Property
- ReferenceMarkerState Property

Important: Learn about programming the Reference marker.

Accessing the Marker object

To turn ON a marker, get a handle to the marker through the measurement object. If not already activated, this command will turn ON marker 1.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
app.ActiveMeasurement.marker(1).Format = naLinMag
```

You can also set the marker object to an object variable:
Dim m1 As Marker
Set m1 = app.ActiveMeasurement.Marker(1)
m1.Format = naMarkerFormat_LinMag

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
- All about Markers

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activate</td>
<td>IMarker</td>
<td>Makes an object the Active Object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Marker Object</td>
</tr>
<tr>
<td>BandnoiseData</td>
<td>IMarker6</td>
<td>Returns the Y-axis data from the bandnoise marker.</td>
</tr>
<tr>
<td>BandpowerData</td>
<td>IMarker6</td>
<td>Returns the Y-axis data from the bandpower marker.</td>
</tr>
<tr>
<td>SearchCompressionPoint</td>
<td>IMarker4</td>
<td>Searches the marker domain for the compression level.</td>
</tr>
<tr>
<td>SearchMax</td>
<td>IMarker</td>
<td>Searches the marker domain for the maximum value.</td>
</tr>
<tr>
<td>SearchMin</td>
<td>IMarker</td>
<td>Searches the marker domain for the minimum value.</td>
</tr>
<tr>
<td>SearchNextPeak</td>
<td>IMarker</td>
<td>Searches the marker's domain for the next largest peak value.</td>
</tr>
<tr>
<td>SearchPeakLeft</td>
<td>IMarker</td>
<td>Searches the marker's domain for the next VALID peak to the left of the marker.</td>
</tr>
<tr>
<td>SearchPeakRight</td>
<td>IMarker</td>
<td>Searches the marker's domain for the next VALID peak to the right of the marker.</td>
</tr>
<tr>
<td>SearchTarget</td>
<td>IMarker</td>
<td>Searches the marker's domain for the target value.</td>
</tr>
<tr>
<td>SearchTargetLeft</td>
<td>IMarker</td>
<td>Moving to the left of the marker position, searches the marker's domain for the target value.</td>
</tr>
<tr>
<td>SearchTargetRight</td>
<td>IMarker</td>
<td>Moving to the right of the marker position, searches the marker's domain for the target value.</td>
</tr>
<tr>
<td>SetCenter</td>
<td>IMarker</td>
<td>Changes the analyzer's center frequency to the X-axis position of the marker.</td>
</tr>
</tbody>
</table>
SetCW IMarker Changes the sweep type to CW mode and makes the CW frequency the marker's frequency.

SetCWFreq IMarker3 Changes the CW frequency to the frequency of the active marker.

SetElectricalDelay IMarker Changes the measurement's electrical delay to the marker's delay value.

SetReferenceLevel IMarker Changes the measurement's reference level to the marker's Y-axis value.

SetStart IMarker Changes the analyzer's start frequency to the X-axis position of the marker.

SetStop IMarker Changes the analyzer's stop frequency to the X-axis position of the marker.

toSA Method IMarker6 Creates an SA channel with a marker at the same CW frequency.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BandDensityACPRState IMarker9</td>
<td>Sets and reads the band ACPR density marker.</td>
</tr>
<tr>
<td>BandDensityBW IMarker8</td>
<td>Sets and reads the bandwidth of the band density marker.</td>
</tr>
<tr>
<td>BandDensityEQSPan IMarker8</td>
<td>Reads the band power density equivalent span.</td>
</tr>
<tr>
<td>BandDensityNoiseState IMarker8</td>
<td>Sets and reads the band noise density state.</td>
</tr>
<tr>
<td>BandDensityNPRState IMarker9</td>
<td>Sets and reads the band NPR density marker.</td>
</tr>
<tr>
<td>BandDensityPowerBW IMarker8</td>
<td>Sets and reads the band power density bandwidth.</td>
</tr>
<tr>
<td>BandDensityPowerState IMarker8</td>
<td>Sets and reads the band power density state.</td>
</tr>
<tr>
<td>BandDensityToneBW IMarker8</td>
<td>Sets and reads the bandwidth of the band tone density marker.</td>
</tr>
<tr>
<td>BandDensityToneSpacing IMarker8</td>
<td>Sets and reads the tone density tone spacing.</td>
</tr>
<tr>
<td>BandDensityToneState IMarker8</td>
<td>Sets and reads the band tone density state.</td>
</tr>
<tr>
<td>BandDensityValue IMarker8</td>
<td>Returns the spectral band power density.</td>
</tr>
<tr>
<td>BandNoisedBmpHz IMarker7</td>
<td>Returns the Y-axis data from the band noise marker.</td>
</tr>
<tr>
<td>BandnoiseSpan IMarker6</td>
<td>Sets and reads the frequency span of the band noise marker.</td>
</tr>
<tr>
<td>BandnoiseState IMarker6</td>
<td>Sets and reads the state of the band noise marker.</td>
</tr>
</tbody>
</table>
BandPowerdBm IMarker7 Returns the Y-axis data from the band power marker.
BandpowerSpan IMarker6 Sets and reads the frequency span of the band power marker.
BandpowerState IMarker6 Sets and reads the state of the band power marker.
Bucket Number IMarker Marker data point number.
CompressionLevel IMarker4 Set and read the marker compression level.
CompressionPin IMarker4 Reads the input power at the marker compression level.
CompressionPout IMarker4 Reads the output power at the marker compression level.
DeltaMarker IMarker Makes a marker relative to the reference marker
Distance IMarker Sets or returns distance value for time domain trace.
Format IMarker Linear, SWR, and so forth
Interpolated IMarker Turn marker interpolation ON and OFF
Number IMarker Read the number of the active marker
OccupiedBandCenter IMarker7 Read the occupied bandwidth center frequency.
OccupiedBandPercent IMarker7 Sets and reads the percentage of the band span to measure.
OccupiedBandPowerdBm IMarker7 Read the occupied bandwidth power.
OccupiedBandSpan IMarker7 Read the span of the occupied bandwidth.
OccupiedBandState IMarker7 Sets and reads the occupied bandwidth on/off state.
PeakExcursion IMarker Sets and reads the peak excursion value for the specified marker.
PeakThreshold IMarker Sets peak threshold for the specified marker.
SearchFunction IMarker Emulates the Tracking function in the marker search dialog box.
Stimulus IMarker Sets and reads the X-Axis value of the marker.
Target Value IMarker Sets the target value for the marker when doing Target Searches.
Tracking IMarker The tracking function finds the selected search function every sweep.
Type IMarker Sets and reads the marker type.
UserRange IMarker Assigns the marker to the specified User Range.
UserRangeMax IMarker Sets the stimulus stop value for the specified User Range.
UserRangeMin IMarker Sets the stimulus start value for the specified User Range.
Value IMarker Reads the Y-Axis value of the marker.

Marker History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMarker</td>
<td>1.0</td>
</tr>
<tr>
<td>IMarker2</td>
<td>4.2</td>
</tr>
<tr>
<td>IMarker3</td>
<td>8.33</td>
</tr>
<tr>
<td>IMarker4</td>
<td>8.50</td>
</tr>
<tr>
<td>IMarker5</td>
<td></td>
</tr>
<tr>
<td>IMarker6</td>
<td>10.30</td>
</tr>
<tr>
<td>IMarker7</td>
<td>12.80</td>
</tr>
<tr>
<td>IMarker8</td>
<td>13.50</td>
</tr>
<tr>
<td>IMarker9</td>
<td>13.60</td>
</tr>
</tbody>
</table>
Measurement Object

See IArrayTransfer Interface for putting and getting typed data.

See IMixer Interface (used with Option S93083A/B)

Description

The Measurement object is probably the most used object in the VNA Object Model. A measurement object represents the chain of data processing algorithms that take raw data from the channel and make it ready for display, which then becomes the scope of the Trace object.

A Measurement object is defined by it's parameter (S11, S22, A/R1, B and so forth). The measurement object is associated with a channel which drives the hardware that produces the data that feeds the measurement. The root of a measurement is the raw data. This buffer of complex paired data then flows through a number of processing blocks: error-correction, trace math, phase correction, time domain, gating, formatting. All of these are controlled through the measurement object.

The ACTIVE measurement is the measurement that will be acted upon if you make a setting from the front panel. It is the measurement whose "button" is pressed in the window with the red "active window" frame. If you create a new measurement, that measurement becomes the active measurement. Therefore, all automation methods with the word "Active" in them refer to the object associated with the Active measurement, whether that object is a Channel, Window, Trace or Limit line.

Learn about the IMeasurement2 Interface for reading stimulus properties.

Accessing the Measurement object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim meas As IMeasurement
Set meas = app.ActiveMeasurement

or

Set meas = app.Measurements(n)
```

You can access four other objects through the Measurement object: markers, limit test, transform, and gating. For example, because each measurement has its own set of markers, you can set a marker by doing this:

```vba
Dim meas as measurement
Set meas = app.ActiveMeasurement
meas.marker(1).Stimulus = 900e6
```
**IMeasurement2 Interface**

Some of the properties and methods for the IMeasurement2 Interface return stimulus values that are set using the channel object. The following is the reason these properties and methods are duplicated.

Every measurement carries with it a snapshot of the stimulus properties of the channel that were in effect when the measurement last acquired data. Therefore, it is the measurement that provides the most accurate stimulus description of its data. Any change made to the channel after the measurement was acquired renders the IChannel interface unreliable in terms of describing the measurement.

**See Also:**

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
- Superseded commands

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activate</td>
<td>IMeasurement</td>
<td>Makes a measurement the active measurement.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Marker Object</td>
</tr>
<tr>
<td>ActivateMarker</td>
<td>IMeasurement</td>
<td>Makes a marker the Active Marker.</td>
</tr>
<tr>
<td>ChangeParameter</td>
<td>IMeasurement</td>
<td>Changes the parameter of the measurement.</td>
</tr>
<tr>
<td>DataToDivisor</td>
<td>IMeasurement</td>
<td>Superseded with DoReceiverPowerCal Method</td>
</tr>
<tr>
<td>DataToMemory</td>
<td>IMeasurement</td>
<td>Stores the active measurement into memory.</td>
</tr>
<tr>
<td>Delete</td>
<td>IMeasurement</td>
<td>Deletes the measurement object.</td>
</tr>
<tr>
<td>DeleteAllMarkers</td>
<td>IMeasurement</td>
<td>Deletes all of the markers from the measurement.</td>
</tr>
<tr>
<td>DeleteMarker</td>
<td>IMeasurement</td>
<td>Deletes a marker from the active measurement.</td>
</tr>
<tr>
<td>getData</td>
<td>IMeasurement</td>
<td>Retrieves Complex data from analyzer memory.</td>
</tr>
</tbody>
</table>
GetDataBuffer  IMeasurement19  Retrieves trace data (Y data) from the modulation distortion measurement.
GetDataBufferCompact  IMeasurement19  Retrieves compact signal trace data (Y data) from the modulation distortion measurement.
getDataByString  IMeasurement  Retrieves variant data from the specified location in your choice of formats.
GetFilterStatistics  IMeasurement  Returns all four Filter Statistics.
GetReferenceMarker  IMeasurement  Returns a handle to the reference marker.
Get SnPData  IMeasurement3  Returns SnP data.
GetSnpDataWithSpecifiedPorts  IMeasurement7  Returns sNp data for the specified ports.
GetTraceStatistics  IMeasurement  Returns the Trace Statistics.
GetXAxisValues  IMeasurement2  Returns the stimulus values for the measurement.
GetXDataBuffer  IMeasurement19  Retrieves frequency tone data from the modulation distortion measurement.
GetXDataBufferCompact  IMeasurement19  Retrieves compact signal frequency tone data from the modulation distortion measurement.
InterpolateMarkers  IMeasurement  Turns All Marker Interpolation ON and OFF for the measurement.
putDataComplex  IMeasurement  Puts complex data into one of five data buffers.
putDataScalar  IMeasurement  Puts formatted variant data into the measurement results buffer.
SearchFilterBandwidth  IMeasurement  Searches the domain with the current BW target.
TraceHoldClear  IMeasurement16  Resets the currently-stored data points to the live data trace and restarts the currently-selected Trace Hold type.
WriteSnpFileWithSpecifiedPorts  IMeasurement7  Write sNp data for specified ports to a file.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ActiveMarker</strong></td>
<td>IMeasurement</td>
<td>Returns a handle to the Active Marker object.</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td><strong>ActiveXAxisRange</strong></td>
<td>IMeasurement14</td>
<td>Sets the X-axis range to display for GCX, SMC, and VMC measurements.</td>
</tr>
<tr>
<td><strong>BalancedMeasurement</strong></td>
<td>IMeasurement5</td>
<td>Sets the measurement type that is used with balanced topologies.</td>
</tr>
<tr>
<td><strong>BandwidthTarget</strong></td>
<td>IMeasurement</td>
<td>The insertion loss value at which the bandwidth of a filter is measured.</td>
</tr>
<tr>
<td><strong>BandwidthTracking</strong></td>
<td>IMeasurement</td>
<td>Turns Bandwidth Tracking function ON and OFF.</td>
</tr>
<tr>
<td><strong>CalibrationName</strong></td>
<td>IMeasurement2</td>
<td>Returns the name of the cal type.</td>
</tr>
<tr>
<td><strong>CalibrationType</strong></td>
<td>IMeasurement</td>
<td><strong>Superseded with</strong> CalibrationTypeID_property</td>
</tr>
<tr>
<td><strong>CalibrationTypeID</strong></td>
<td>IMeasurement2</td>
<td>Sets or returns the cal type for the current measurement.</td>
</tr>
<tr>
<td><strong>Center</strong></td>
<td>IMeasurement2</td>
<td>Returns the stimulus value of the center point for the measurement.</td>
</tr>
<tr>
<td><strong>channelNumber</strong></td>
<td>IIMeasurement</td>
<td>Returns the channel number.</td>
</tr>
<tr>
<td><strong>CustomMeasurementConfiguration</strong></td>
<td>IIMeasurement12</td>
<td>Provides access to custom measurement properties and methods.</td>
</tr>
<tr>
<td><strong>Domain</strong></td>
<td>IMeasurement2</td>
<td>Returns the domain (frequency, time, power) for the measurement.</td>
</tr>
<tr>
<td><strong>ElectricalDelay</strong></td>
<td>IMeasurement</td>
<td>Sets electrical delay.</td>
</tr>
<tr>
<td><strong>ElecDelayMedium</strong></td>
<td>IMeasurement2</td>
<td>Sets or returns the characteristic of the electrical delay medium.</td>
</tr>
<tr>
<td><strong>ElecDistanceDelay</strong></td>
<td>IMeasurement11</td>
<td>Sets delay in distance</td>
</tr>
<tr>
<td><strong>ElecDistanceDelayUnit</strong></td>
<td>IMeasurement11</td>
<td>Sets distance units</td>
</tr>
<tr>
<td><strong>ErrorCorrection</strong></td>
<td>IMeasurement</td>
<td>Set or get the state of error correction for the measurement.</td>
</tr>
<tr>
<td><strong>ErrorCorrectionIndicator</strong></td>
<td>IMeasurement14</td>
<td>Returns the error correction status of the measurement.</td>
</tr>
<tr>
<td><strong>FilterBW</strong></td>
<td>IMeasurement</td>
<td>Returns the results of the SearchBandwidth method.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FilterCF</td>
<td>IMeasurement</td>
<td>Returns the Center Frequency result of the SearchBandwidth method.</td>
</tr>
<tr>
<td>FilterLoss</td>
<td>IMeasurement</td>
<td>Returns the Loss value of the SearchBandwidth method.</td>
</tr>
<tr>
<td>FilterQ</td>
<td>IMeasurement</td>
<td>Returns the Q (quality factor) result of the SearchBandwidth method.</td>
</tr>
<tr>
<td>Format</td>
<td>IMeasurement</td>
<td>Sets display format.</td>
</tr>
<tr>
<td>FormatUnit</td>
<td>IMeasurement9</td>
<td>Sets units for unratioed measurements.</td>
</tr>
<tr>
<td>Gating</td>
<td>IMeasurement</td>
<td>Controls Time Domain Gating.</td>
</tr>
<tr>
<td>GroupDelayAperture</td>
<td>IMeasurement13</td>
<td>Provides access to the Group Delay Aperture settings.</td>
</tr>
<tr>
<td>InterpolateCorrection</td>
<td>IMeasurement</td>
<td>Turns ON and OFF the calculation of new error terms when stimulus values change.</td>
</tr>
<tr>
<td>InterpolateMemory</td>
<td>IMeasurement18</td>
<td>Sets or returns the state of the memory data interpolation.</td>
</tr>
<tr>
<td>InterpolateNormalization</td>
<td>IMeasurement</td>
<td><strong>Superseded with</strong> DoReceiverPowerCal Method</td>
</tr>
<tr>
<td>IsSparameter</td>
<td>IMeasurement2</td>
<td>Returns true if measurement represents an S-Parameter.</td>
</tr>
<tr>
<td>LimitTest</td>
<td>IMeasurement</td>
<td>Collection for iterating through the Limit Segment objects (Limit Lines).</td>
</tr>
<tr>
<td>LimitTestFailed</td>
<td>IMeasurement</td>
<td>Returns the results of limit testing</td>
</tr>
<tr>
<td>LoadPort</td>
<td>IMeasurement</td>
<td>Returns the load port number associated with an S-parameter reflection measurement.</td>
</tr>
<tr>
<td>LogMagnitudeOffset</td>
<td>IMeasurement</td>
<td><strong>Superseded with</strong> DoReceiverPowerCal Method</td>
</tr>
<tr>
<td>MagnitudeOffset</td>
<td>IMeasurement4</td>
<td>Offsets the magnitude of the entire data trace to a specified value.</td>
</tr>
<tr>
<td>MagnitudeSlopeOffset</td>
<td>IMeasurement4</td>
<td>Offsets the magnitude of the data trace to a value that changes linearly with frequency.</td>
</tr>
<tr>
<td>Marker</td>
<td>IMeasurement</td>
<td>Provides access to Marker settings.</td>
</tr>
<tr>
<td>Marker State</td>
<td>IMeasurement3</td>
<td>Sets or returns the ON / OFF state of a marker.</td>
</tr>
<tr>
<td>Mean</td>
<td>IMeasurement</td>
<td>Returns the mean value of the measurement.</td>
</tr>
<tr>
<td><strong>Measurement</strong></td>
<td><strong>Equation</strong></td>
<td><strong>IMeasurement</strong></td>
</tr>
<tr>
<td>----------------</td>
<td>--------------</td>
<td>------------------</td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td><strong>IMeasurement</strong></td>
<td><strong>Sets or returns the name of the measurement.</strong></td>
</tr>
<tr>
<td><strong>NWindow</strong></td>
<td><strong>IMeasurement</strong></td>
<td><strong>Controls the part of the display that contains the graticule, or what is written on the display.</strong></td>
</tr>
</tbody>
</table>

**Normalization**  
| **Number** | **IMeasurement** | Returns the number of the measurement. |       |
| **NumberOfPoints** | **IMeasurement2** | Returns the Number of Points of the measurement. |       |
| **Parameter** | **IMeasurement** | Returns the measurement Parameter. |       |
| **PeakToPeak** | **IMeasurement** | Returns the Peak to Peak value of the measurement. |       |
| **PhaseOffset** | **IMeasurement** | Sets the Phase Offset for the active channel. |       |
| **PNOP** | **IMeasurement13** | Provides access to the Power Normal Operating Point marker search object. |       |
| **PSaturation** | **IMeasurement13** | Provides access to the Power Saturation marker search object. |       |
| **ReceivePort** | **IMeasurement2** | Returns the receiver port of the measurement. |       |
| **ReferenceMarkerState** | **IMeasurement** | Turns the reference marker ON or OFF |       |
| **ShowStatistics** | **IMeasurement** | Displays and hides the measurement statistics (peak-to-peak, mean, standard deviation) on the screen. |       |
| **Smoothing** | **IMeasurement** | Turns ON and OFF data smoothing. |       |
| **SmoothingAperture** | **IMeasurement** | Specifies or returns the amount of smoothing as a ratio of the number of data points in the measurement trace. |       |
| **SourcePort** | **IMeasurement2** | Returns the source port of the measurement. |       |
| **Span** | **IMeasurement2** | Returns the stimulus span (stop - start) for the measurement. |       |
StandardDeviation IMeasurement Returns the standard deviation of the measurement.

Start IMeasurement2 Returns the stimulus value of the first point for the measurement.

StatisticsRange IMeasurement Sets the User Range number for calculating measurement statistics.

Stop IMeasurement2 Returns the stimulus value of the last point for the measurement.

Trace IMeasurement Controls scale, reference position, and reference line.

TraceDeviationType IMeasurement20 Calculates deviation from a least-squares fit line.

TraceHoldType IMeasurement16 Sets the type of trace hold to perform.

TraceMath IMeasurement Performs math operations on the measurement object and the trace stored in memory.

TraceMax IMeasurement10 Maximizes the active trace.

TraceTitle IMeasurement8 Writes and reads a trace title.

TraceTitleState IMeasurement8 Turns trace title ON and OFF

Transform IMeasurement Controls Time Domain transforms.

Uncertainty IMeasurement17 Returns a handle to the (Dynamic) Uncertainty Object.

UserRangeMax IMeasurement15 Sets the stimulus stop value for the specified User Range.

UserRangeMin IMeasurement15 Sets the stimulus start value for the specified User Range.

View IMeasurement Sets (or returns) the type of trace displayed on the screen.

WGCutoffFreq IMeasurement2 Sets or returns the value of the waveguide cut off frequency.

XAxis IMeasurement17 Sets the X-axis of the selected measurement to a DC Source.

XAxisDomain IMeasurement17 Sets and returns the X-Axis domain of the selected DIQ measurement.
**IMeasurement History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMeasurement</td>
<td>1.0</td>
</tr>
<tr>
<td>IMeasurement2</td>
<td>3.0</td>
</tr>
<tr>
<td>IMeasurement3</td>
<td>4.0</td>
</tr>
<tr>
<td>IMeasurement4</td>
<td>4.2</td>
</tr>
<tr>
<td>IMeasurement5</td>
<td>5.0</td>
</tr>
<tr>
<td>IMeasurement7</td>
<td>6.2</td>
</tr>
<tr>
<td>IMeasurement8</td>
<td>7.2</td>
</tr>
<tr>
<td>IMeasurement9</td>
<td>8.35</td>
</tr>
<tr>
<td>IMeasurement10</td>
<td>8.35</td>
</tr>
<tr>
<td>IMeasurement11</td>
<td>8.50</td>
</tr>
<tr>
<td>IMeasurement12</td>
<td>9.00</td>
</tr>
<tr>
<td>IMeasurement13</td>
<td>9.20</td>
</tr>
<tr>
<td>IMeasurement14</td>
<td>9.22</td>
</tr>
<tr>
<td>IMeasurement15</td>
<td>9.40</td>
</tr>
<tr>
<td>IMeasurement16</td>
<td>10.25</td>
</tr>
<tr>
<td>IMeasurement17</td>
<td>10.30</td>
</tr>
<tr>
<td>IMeasurement18</td>
<td>10.49</td>
</tr>
<tr>
<td>IMeasurement19</td>
<td>13.60</td>
</tr>
<tr>
<td>IMeasurement20</td>
<td>13.80</td>
</tr>
</tbody>
</table>

**IArrayTransfer Interface**

**Description**

Contains methods for putting data in and getting data out of the analyzer using typed data. This interface transfers data more efficiently than the IMeasurement Interface. However, this interface is only usable from VB6, C, and C++.

See a VB.net example using this interface.
## Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getComplex</td>
<td>Retrieves real and imaginary data from the specified buffer.</td>
</tr>
<tr>
<td>getNAComplex</td>
<td>Retrieves typed \texttt{NAComplex} data from the specified buffer.</td>
</tr>
<tr>
<td>getPairedData</td>
<td>Retrieves magnitude and phase data pairs from the specified buffer.</td>
</tr>
<tr>
<td>getScalar</td>
<td>Retrieves scalar data from the specified buffer.</td>
</tr>
<tr>
<td>putComplex</td>
<td>Puts real and imaginary data into the specified buffer.</td>
</tr>
<tr>
<td>putNAComplex</td>
<td>Puts typed \texttt{NAComplex} data into the specified buffer.</td>
</tr>
<tr>
<td>putScalar</td>
<td>Puts scalar data into the measurement result buffer.</td>
</tr>
</tbody>
</table>

## Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

## IArrayTransfer History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IArrayTransfer</td>
<td>1.0</td>
</tr>
</tbody>
</table>
MeasurementClassProperties Object

Description

These properties return properties of specific measurement classes.

Accessing the MeasurementClassProperties object

```vba
Set app = CreateObject("AgilentPNA835x.Application")
Set cap = app.Capabilities
'Access the MeasurementClassProperties Object
Set measProps = cap.MeasurementClassProperties("Swept IMD")
list=measProps.SupportedParameters
```

See Also

- MeasurementClassProperties Property
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

Methods

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>See History</td>
</tr>
</tbody>
</table>

None

Properties

| SupportedParameters | MeasurementClassProperties |

IMeasurementClassProperties History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>IMeasurementClassProperties</td>
<td>9.40</td>
</tr>
</tbody>
</table>
MeasurementEquation Object

Description

Provide commands for creating an equation.

Learn more about Equation Editor

Accessing the Equation object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim eq As MeasurementEquation
Set eq = app.ActiveMeasurement.Equation

See Also:

- VNA Automation Interfaces
- The VNA Object Model

### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetLibraryFunctions</td>
<td>IMeasurementEquation2</td>
<td>Returns the functions in an imported (loaded) DLL.</td>
</tr>
<tr>
<td>ImportLibrary</td>
<td>IMeasurementEquation2</td>
<td>Imports an Equation Editor DLL.</td>
</tr>
<tr>
<td>IsLibraryImported</td>
<td>IMeasurementEquation2</td>
<td>Returns whether a DLL has been imported into the VNA.</td>
</tr>
<tr>
<td>RemoveLibrary</td>
<td>IMeasurementEquation2</td>
<td>Removes an imported an Equation Editor DLL from the VNA.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FastProcessing</td>
<td>IMeasurementEquation3</td>
<td>Set and return equation editor trace update delay</td>
</tr>
<tr>
<td>Text</td>
<td>IMeasurementEquation</td>
<td>Sets the Equation</td>
</tr>
<tr>
<td>State</td>
<td>IMeasurementEquation</td>
<td>Sets the Equation enabled state</td>
</tr>
<tr>
<td>Valid</td>
<td>IMeasurementEquation</td>
<td>Returns whether the equation is presently valid.</td>
</tr>
</tbody>
</table>
Example Program using these commands:

```vbscript
Dim na
Dim meas
Set na = CreateObject("AgilentPNA835x.Application")
Set meas = na.ActiveMeasurement
'Make the measurement
meas.Equation.Text = "mysillyequ=sqrt(AR1_1)"
'Check to see if the equation is valid
valid_e = meas.Equation.Valid
MsgBox valid_e
'Turn on the Equation Editor
meas.Equation.State = True
```

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMeasurementEquation</td>
<td>6.03</td>
</tr>
<tr>
<td>IMeasurementEquation2</td>
<td>6.03</td>
</tr>
<tr>
<td>IMeasurementEquation3</td>
<td>13.20</td>
</tr>
</tbody>
</table>
Measurement Collection

Description

A collection object that provides a mechanism for iterating through the Application measurements.

Accessing the Measurements collection

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim measurements As Measurements
Set measurements = app.Measurements
```

See Also:

- Measurement Object
- Collections in the Analyzer
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a Measurement to the collection.</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle on a measurement in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes a measurement from the measurements collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of measurements in the analyzer.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the current Application.</td>
</tr>
</tbody>
</table>
MultiDimensionalSweep Object

Description

Controls a multi-dimensional sweep in a spectrum analyzer channel. This is also called a 3D sweep (power, phase, and frequency). Each sweep can be set to a different power level. Within each sweep is a 2D sweep of phase and frequency.

Accessing the MultiDimensionalSweep object

```vbs
set pna = CreateObject("AgilentPNA835x.Application","A-N5242A-10096")
CreateSAMeasurement
SetupMultiSweep

Sub CreateSAMeasurement
pna.Channels.RemoveChannelNumber(1)
call pna. CreateCustomMeasurementEx(1, "SpectrumAnalyzer", "B")
set chan = pna.ActiveChannel
chan.Hold
End Sub
```

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>See History</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCOrder</td>
<td>IMultiDimensionalSweep</td>
<td>Set and read the order for the specified DC source in the multi-dimensional sweep.</td>
</tr>
<tr>
<td>DCState</td>
<td>IMultiDimensionalSweep</td>
<td>Set and read the specified DC source’s ON/OFF state in the multi-dimensional sweep.</td>
</tr>
<tr>
<td>DimensionCatalog</td>
<td>IMultiDimensionalSweep</td>
<td>Read the names of source domains in the multi-dimensional sweep whose state is ON and whose dimension order is the specified dimension order.</td>
</tr>
<tr>
<td>DimensionCount</td>
<td>IMultiDimensionalSweep</td>
<td>Read the highest dimension order in the multi-dimensional sweep.</td>
</tr>
<tr>
<td>DimensionPointCount</td>
<td>IMultiDimensionalSweep</td>
<td>Set and read the point count for the specified dimension order in the multi-dimensional sweep.</td>
</tr>
</tbody>
</table>
**DimensionRepeatCount**  IMultiDimensionalSweep  Set and read the repeat count for the specified dimension order in the multi-dimensional sweep.

**SourcePortFrequencyOrder**  IMultiDimensionalSweep  Set and read the source frequency domain’s order in the multi-dimensional sweep.

**SourcePortFrequencyState**  IMultiDimensionalSweep  Set and read the source frequency domain’s ON/OFF state in the multi-dimensional sweep.

**SourcePortPhaseOrder**  IMultiDimensionalSweep  Set and read the source phase domain’s order in the multi-dimensional sweep.

**SourcePortPhaseState**  IMultiDimensionalSweep  Set and read the source phase domain’s ON/OFF state in the multi-dimensional sweep.

**SourcePortPowerOrder**  IMultiDimensionalSweep  Set and read the source power domain’s order in the multi-dimensional sweep.

**SourcePortPowerState**  IMultiDimensionalSweep  Set and read the source power domain’s ON/OFF state in the multi-dimensional sweep.

---

**MultiDimensionalSweep History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMultiDimensionalSweep</td>
<td>12.90</td>
</tr>
</tbody>
</table>

---

1153
NAWindow Object

Description

The NAWindow object controls the part of the display that contains the graticule, or what is written on
the display.

Accessing the NaWindow object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim window As NAWindow
Set window = app.NAWindows(1)
window.AutoScale

or

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", "analyzerName")
app.NAWindows(1).AutoScale

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autoscale</td>
<td>INaWindow</td>
<td>Autoscales all measurements in the window. Shared with the Trace Object</td>
</tr>
<tr>
<td>ShowMarkerReadout</td>
<td>INaWindow</td>
<td>Shows and Hides the Marker readout for the active marker in the upper-right corner of the window object.</td>
</tr>
<tr>
<td>ShowTable</td>
<td>INaWindow</td>
<td>Shows or Hides the specified table for the active measurement in the lower part of the window object.</td>
</tr>
<tr>
<td>Properties</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>ActiveTrace</strong></td>
<td>Sets a trace to the Active Trace.</td>
<td></td>
</tr>
<tr>
<td><strong>LimitTestXPosition</strong></td>
<td>Sets the X-axis position of the Limit Line Pass/Fail indicator.</td>
<td></td>
</tr>
<tr>
<td><strong>LimitTestYPosition</strong></td>
<td>Sets the Y-axis position of the Limit Line Pass/Fail indicator.</td>
<td></td>
</tr>
<tr>
<td><strong>MarkerReadout</strong></td>
<td>Sets and reads the state of the Marker readouts.</td>
<td></td>
</tr>
<tr>
<td><strong>MarkerReadoutResponsePlaces</strong></td>
<td>For the Y-axis (response), sets the number digits to display after the decimal point in marker readouts.</td>
<td></td>
</tr>
<tr>
<td><strong>MarkerReadoutSize</strong></td>
<td>Specifies the size of font used when displaying Marker readout in the selected window.</td>
<td></td>
</tr>
<tr>
<td><strong>MarkerReadoutsPerTrace</strong></td>
<td>Sets the number of marker readouts to display per trace.</td>
<td></td>
</tr>
<tr>
<td><strong>MarkerReadoutStimulusPlaces</strong></td>
<td>For the X-axis (stimulus), sets the number digits to display after the decimal point in marker readouts.</td>
<td></td>
</tr>
<tr>
<td><strong>MarkerReadoutXPosition</strong></td>
<td>Sets the X-axis position of marker readouts.</td>
<td></td>
</tr>
<tr>
<td><strong>MarkerReadoutYPosition</strong></td>
<td>Sets the Y-axis position of marker readouts.</td>
<td></td>
</tr>
<tr>
<td><strong>MarkerSymbol</strong></td>
<td>Sets the symbol to display for marker position.</td>
<td></td>
</tr>
<tr>
<td><strong>MarkerSymbolsAboveTrace</strong></td>
<td>Specifies whether or not to force marker symbols to be displayed above the trace</td>
<td></td>
</tr>
<tr>
<td><strong>OneMarkerReadoutPerTrace</strong></td>
<td><strong>Superseded with</strong> MarkerReadoutsPerTrace Property</td>
<td></td>
</tr>
<tr>
<td><strong>ScaleCouplingMethod</strong></td>
<td>Sets and returns the method of scale coupling.</td>
<td></td>
</tr>
<tr>
<td><strong>ScaleCouplingState</strong></td>
<td>Enables and disables scale coupling for the specified window.</td>
<td></td>
</tr>
<tr>
<td><strong>Title</strong></td>
<td>Writes or reads a custom title for the window.</td>
<td></td>
</tr>
<tr>
<td><strong>TitleState</strong></td>
<td>Turns ON and OFF the window title.</td>
<td></td>
</tr>
<tr>
<td><strong>Traces</strong></td>
<td>Collection for getting a handle to a trace or iterating through the traces in a window.</td>
<td></td>
</tr>
<tr>
<td><strong>WindowNumber</strong></td>
<td>Reads the number of the active window.</td>
<td></td>
</tr>
<tr>
<td><strong>WindowState</strong></td>
<td>Maximizes or minimizes a window.</td>
<td></td>
</tr>
</tbody>
</table>
## INaWindow History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>INaWindow</td>
<td>1.0</td>
</tr>
<tr>
<td>INaWindow2</td>
<td>9.0</td>
</tr>
<tr>
<td>INaWindow3</td>
<td>9.30</td>
</tr>
<tr>
<td>INaWindow4</td>
<td>10.30</td>
</tr>
</tbody>
</table>
NAWindows Collection

Description

A collection object that provides a mechanism for iterating through the Application windows.

Accessing the NaWindows collection

```vbnet
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim windows As NAWindows
Set windows = app.NAWindows
```

See Also:

- NAWindow Object
- Collections in the Analyzer
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a window to the NAWindows collection.</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle to a window in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes a window from the NAWindows collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of windows on the analyzer.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the current Application.</td>
</tr>
</tbody>
</table>
NoiseCal Object

Description

Controls the noise figure calibration settings for amplifiers and converters. These commands supplement the standard calibration commands on the GuidedCalibration Object.

Accessing the NoiseCal object

```vba
Dim app as AgilentPNA835x.Application
Set noisecal = pna.GetCalmanager.CreateCustomCalEx(channelNum)
Set noiseCalExtension = noisecal.CustomCalConfiguration
noiseCalExtension.NoiseSourceCold = 300
```

See Also

- Examples:
  - Create and Cal a Noise Figure Measurement
  - Create and Cal an NFX Measurement
- NoiseFigure Object
- About Noise Figure Measurements
- About NFX Measurements
- VNA Automation Interfaces
- The VNA Object Model
<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoOrientTuner</td>
<td>INoiseCal2</td>
<td>Sets the state of auto orientation for a noise tuner during Noise Figure for NFX.</td>
</tr>
<tr>
<td>CalMethod</td>
<td>INoiseCal</td>
<td>Sets and returns the method for performing calibration on a noise channel.</td>
</tr>
<tr>
<td>EnableLOPowerCal</td>
<td>INoiseCal2</td>
<td>Enables and disables LO power calibration for NFX.</td>
</tr>
<tr>
<td>ENRFile</td>
<td>INoiseCal</td>
<td>Sets and returns the name of the ENR file associated with the noise source.</td>
</tr>
<tr>
<td>ForceDeEmbedENRAdapter</td>
<td>INoiseCal2</td>
<td>Sets and reads the state of ENR adapter de-embedding.</td>
</tr>
<tr>
<td>ForceDeEmbedSensorAdapter</td>
<td>INoiseCal2</td>
<td>Sets and reads the state of noise source adapter de-embedding.</td>
</tr>
<tr>
<td>ForceDeEmbedThruAdapter</td>
<td>INoiseCal4</td>
<td>Sets and reads the state of Thru adapter de-embedding.</td>
</tr>
<tr>
<td>NoiseSourceCalKitType</td>
<td>INoiseCal</td>
<td>Sets and reads the Cal Kit type used to perform a cal at the adapter which is used to connect the noise source (if required.)</td>
</tr>
<tr>
<td>NoiseSourceCold</td>
<td>INoiseCal</td>
<td>Sets and returns the current temperature at the noise source.</td>
</tr>
<tr>
<td>NoiseSourceConnectorType</td>
<td>INoiseCal</td>
<td>Sets and reads the connector type of the noise source used during the cal.</td>
</tr>
<tr>
<td>RcvCharMethod</td>
<td>INoiseCal3</td>
<td>Set and read the method used to characterize the noise receivers.</td>
</tr>
</tbody>
</table>

**NoiseConfiguration History**
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>INoiseCal</td>
<td>8.0</td>
</tr>
<tr>
<td>INoiseCal2</td>
<td>9.10</td>
</tr>
<tr>
<td>INoiseCal3</td>
<td>9.70</td>
</tr>
<tr>
<td>INoiseCal4</td>
<td>12.90</td>
</tr>
</tbody>
</table>
NoiseFigure Object

Description

Controls the Noise Figure application settings.

Accessing the NoiseFigure object

```vba
Dim app as AgilentPNA835x.Application
app.CreateCustomMeasurementEx(1, "NoiseFigure", "NF", 1)

Dim NoiseFig
Set NoiseFig = app.ActiveChannel.CustomChannelConfiguration
```

See Also:

- Example programs
  - Create and Cal a NoiseFigure Measurement
  - Create and Cal an NFX Measurement
- About Noise Figure Measurements
- Noise Figure Calibration Object
- app.NoiseSourceState (ON and OFF)
- ENRFile Object
- VNA Automation Interfaces
- The VNA Object Model
<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetSnPData</td>
<td>INoiseFigure7</td>
<td>Reads noise parameter snp data.</td>
</tr>
<tr>
<td>SetPortMap</td>
<td>INoiseFigure6</td>
<td>Maps DUT ports to PNA-X ports (Opt 028)</td>
</tr>
<tr>
<td>WriteSnPData</td>
<td>INoiseFigure7</td>
<td>Reads noise parameter data to snp file.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AmbientTemperature</td>
<td>INoiseFigure</td>
<td>Sets the air temperature at which the measurement is being performed.</td>
</tr>
<tr>
<td>DeviceInputPort</td>
<td>INoiseFigure6</td>
<td>Read DUT input port map</td>
</tr>
<tr>
<td>DeviceOutputPort</td>
<td>INoiseFigure6</td>
<td>Read DUT output port map</td>
</tr>
<tr>
<td>ImpedanceStates</td>
<td>INoiseFigure</td>
<td>Sets the number of impedance states to use during calibrated measurements.</td>
</tr>
<tr>
<td>NarrowBand</td>
<td>INoiseFigure8</td>
<td>Enable narrowband compensation</td>
</tr>
<tr>
<td>NoiseAverageFactor</td>
<td>INoiseFigure</td>
<td>Set averaging of noise receiver.</td>
</tr>
<tr>
<td>NoiseAverageState</td>
<td>INoiseFigure</td>
<td>Turn noise averaging ON and OFF</td>
</tr>
<tr>
<td>NoiseBandwidth</td>
<td>INoiseFigure</td>
<td>Set bandwidth of noise receiver.</td>
</tr>
<tr>
<td>NoiseGain</td>
<td>INoiseFigure</td>
<td>Set gain state of noise receiver.</td>
</tr>
<tr>
<td>NoiseGainCTCheck</td>
<td>INoiseFigure3</td>
<td>Turns noise threshold checking ON and OFF.</td>
</tr>
<tr>
<td>NoiseReceiver</td>
<td>INoiseFigure5</td>
<td>Sets and returns the receiver to use for noise measurements.</td>
</tr>
<tr>
<td>NoiseReceiverSweepTime</td>
<td>INoiseFigure3</td>
<td>Returns an estimate of sweep time.</td>
</tr>
<tr>
<td>NoiseTuner</td>
<td>INoiseFigure</td>
<td>Sets and returns the noise tuner identifier,</td>
</tr>
<tr>
<td>NoiseTunerIn</td>
<td>INoiseFigure</td>
<td>Sets and returns the port identifier of the ECal noise tuner Input</td>
</tr>
<tr>
<td>NoiseTunerOut</td>
<td>INoiseFigure</td>
<td>Sets and returns the port identifier of the ECal noise tuner Output</td>
</tr>
<tr>
<td>SourcePullForSParameters</td>
<td>INoiseFigure4</td>
<td>Set and read the use of source pull technique to compute S22 on Noise Figure on Converters.</td>
</tr>
</tbody>
</table>
### NoiseFigure History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>INoiseFigure</td>
<td>8.0</td>
</tr>
<tr>
<td>INoiseFigure3</td>
<td>9.0</td>
</tr>
<tr>
<td>INoiseFigure4</td>
<td>9.1</td>
</tr>
<tr>
<td>INoiseFigure5</td>
<td>9.2</td>
</tr>
<tr>
<td>INoiseFigure6</td>
<td>9.22</td>
</tr>
<tr>
<td>INoiseFigure7</td>
<td>9.80</td>
</tr>
<tr>
<td>INoiseFigure8</td>
<td>10.15</td>
</tr>
</tbody>
</table>
PathConfiguration Object

Description

Provides access to the path configuration currently active on the channel object.

To load, store, or delete a configuration, see ConfigurationManager Object.

Accessing the PathConfiguration object in VB

```vbnet
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim chan as Channel
Set chan = app.ActiveChannel

Dim pathConfig As PathConfiguration
Set pathConfig = chan.PathConfiguration
```

Accessing the PathConfiguration object in C#

```csharp
Type pnaType = Type.GetTypeFromProgID("AgilentPNA835x.Application", "PNA-NAME-HERE");

AgilentPNA835x.Application pna =
    (AgilentPNA835x.Application)Activator.CreateInstance(pnaType);

AgilentPNA835x.Channel chan = (AgilentPNA835x.Channel)pna.ActiveChannel;

AgilentPNA835x.PathConfiguration path =
    (AgilentPNA835x.PathConfiguration)chan.get_PathConfiguration();
```

Note:

To learn how to make configuration (element) settings, see this Path Configuration Example

Also see this list of configurable elements and settings.

See Also:

- PathConfigurationManager Object
- PathElement Object
- Path Configurator UI
## VNA Automation Interfaces

### The VNA Object Model

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CopyFrom</td>
<td>IPathConfiguration2</td>
<td>Copy the mechanical switch settings and attenuator settings from the specified channel to the active channel.</td>
</tr>
<tr>
<td>Store</td>
<td>IPathConfiguration</td>
<td>Saves the current configuration to the specified name.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DescriptiveText</td>
<td>IPathConfiguration</td>
<td>Write and read descriptive text associated with the configuration.</td>
</tr>
<tr>
<td>Elements</td>
<td>IPathConfiguration</td>
<td>Collection of Elements that can be configured (switches and so forth). See the list of elements and settings.</td>
</tr>
<tr>
<td>Element</td>
<td>IPathConfiguration</td>
<td>Returns a handle to a IPathElement object.</td>
</tr>
<tr>
<td>Name</td>
<td>IPathConfiguration</td>
<td>Returns the name of the current configuration.</td>
</tr>
<tr>
<td>Parent</td>
<td>IPathConfiguration</td>
<td>Returns a pointer to the parent COM object (Channel).</td>
</tr>
</tbody>
</table>

### IPathConfiguration History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPathConfiguration</td>
<td>7.2</td>
</tr>
<tr>
<td>IPathConfiguration2</td>
<td>9.4</td>
</tr>
</tbody>
</table>
PathConfigurationManager Object

Description

These commands allow configurations to be stored, loaded, or deleted on the VNA.

To make path configuration settings, see PathConfiguration Object and the PathElement Object

Accessing the PathConfigurationManager object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim pathConfig As PathConfigurationManager
Set pathConfig = app.PathConfigurationManager

Note:

To learn how to make configuration (element) settings, see this Path Configuration Example

Also see this list of configurable elements and settings.

See Also:

- Path Configuration Example
- Path Configurator
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeleteConfiguration</td>
<td>IPathConfigurationManager</td>
<td>Deletes the specified configuration from the VNA.</td>
</tr>
<tr>
<td>LoadConfiguration</td>
<td>IPathConfigurationManager</td>
<td>Loads the named configuration.</td>
</tr>
<tr>
<td>StoreConfiguration</td>
<td>IPathConfigurationManager</td>
<td>Saves the path configuration</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configurations</td>
<td>IPathConfigurationManager</td>
<td>Returns a list of configuration names stored in the VNA.</td>
</tr>
<tr>
<td>Parent</td>
<td>IPathConfigurationManager</td>
<td>Returns a handle to the Application object.</td>
</tr>
</tbody>
</table>

### IPathConfigurationManager History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPathConfigurationManager</td>
<td>7.2</td>
</tr>
</tbody>
</table>
PathElement Object

Description

Provides access to the settings for the PathElement object.

Accessing the PathElement object in VB

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim chan as Channel
Set chan = app.ActiveChannel

Dim pathConfig As PathConfiguration
Set pathConfig = chan.PathConfiguration

Dim element as PathElement
Set element = pathConfig.PathElement("Src1")

Accessing the PathElement object in C#

Type pnaType = Type.GetTypeFromProgID("AgilentPNA835x.Application", "PNA-NAME-HERE");

AgilentPNA835x.Application pna =
    (AgilentPNA835x.Application)Activator.CreateInstance(pnaType);

AgilentPNA835x.Channel chan = (AgilentPNA835x.Channel)pna.ActiveChannel;

AgilentPNA835x.PathConfiguration path =
    (AgilentPNA835x.PathConfiguration)chan.get_PathConfiguration();

path.get_Element("Port1RefMxr").Value = "External";

Note:

To learn how to make configuration (element) settings, see this Path Configuration Example
Also see this list of configurable elements and settings.

See Also:

- Path Configurator
- PathConfigurationManager Object
PathConfiguration Object

VNA Automation Interfaces

The VNA Object Model

Example Programs

### Methods

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>See History</td>
</tr>
</tbody>
</table>

None

### Properties

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>See History</td>
</tr>
</tbody>
</table>

| Name       | IPathElement | Returns the name of the element. |
| Parent     | IPathElement | Returns a pointer to the Parent Object (PathConfiguration) |
| Value      | IPathElement | Read / Write get the current setting for the element. |
| Values     | IPathElement | Returns all valid settings for the element. |

### IPathElement History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPathElement</td>
<td>7.2</td>
</tr>
</tbody>
</table>
PhaseControl Object

Description

Contains the properties for configuring Phase Sweep (Opt S93088A/B) in the VNA.

Accessing the PhaseControl object

```
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim phase as PhaseControl
Set phase = chan.PhaseControl
```

See Also:

- About Phase Control
- Set Phase Sweep using SweepType Property
- VNA Automation Interfaces
- The VNA Object Model
- Example Program

Methods

None

Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CouplePhasePortSettings</td>
<td>IPhaseControl</td>
<td>Set and return whether to couple phase control settings.</td>
</tr>
<tr>
<td>FixedPhase</td>
<td>IPhaseControl</td>
<td>Set and return the fixed phase value.</td>
</tr>
<tr>
<td>FixedRatioedPower</td>
<td>IPhaseControl</td>
<td>Set and return the fixed ratioed power value.</td>
</tr>
<tr>
<td>PhaseControlMode</td>
<td>IPhaseControl</td>
<td>Set and return the ON/Off state of phase control.</td>
</tr>
<tr>
<td>PhaseCorrectionData</td>
<td>IPhaseControl</td>
<td>Set and return an array of phase offsets.</td>
</tr>
<tr>
<td>PhaseCorrectionEnabled</td>
<td>IPhaseControl</td>
<td>Set and return whether to use the phase correction offset array.</td>
</tr>
</tbody>
</table>
PhaseIterationNumber      IPhaseControl  Set and return max number of leveling sweeps
PhaseParameter           IPhaseControl  Set and return the ratioed receivers (parameter) to use for phase control.
PhaseParameterModes      IPhaseControl  Returns the available phase control modes for the specified port.
PhaseReferencePort       IPhaseControl  Sets and returns the reference port for the Phase Control measurement.
PhaseTolerance           IPhaseControl  Set and return tolerance value for leveling sweeps
RatioedPowerCorrectionData IPhaseControl  Set and return ratioed power offset data
RatioedPowerCorrectionEnabled IPhaseControl  Set and return whether to use the ratioed power offset array
StartPhase                IPhaseControl  Set and return the start value of phase sweep.
StartRatioedPower         IPhaseControl  Set and return the start ratioed power value.
StopPhase                 IPhaseControl  Set and return the stop value of phase sweep.
StopRatioedPower          IPhaseControl  Set and return the stop ratioed power value.

IPhaseControl History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPhaseControl</td>
<td>9.33</td>
</tr>
</tbody>
</table>
PhaseReferenceCalibration Object

Description

Use this interface to perform a Phase Reference Calibration. This calibration is performed as a 'Tier 1' cal. After it is performed and saved, it can then be recalled into an SMC+Phase Calibration.

- It is NOT necessary to create an SMC measurement before performing a remote Phase Reference Cal. It is necessary when performed from the user interface.
- Port selection is made remotely by selecting connectors and Cal Kits for the ports to be included in the SOLT calibration.

Accessing the PhaseReferenceCalibration Object

```vba
Dim app as AgilentPNA835x.Application
Dim CalMgr
Set CalMgr = app.GetCalManager
Dim phaseRefCal
Set phaseRefCal = CalMgr.PhaseReferenceCalibration
' Then get a handle to the GuidedCal object
Dim guidedCal
Set guidedCal = phaseRefCal.GuidedCalibration
```

See Also:

- Example program
- Learn about Calibrate All Channels.
- VNA Automation Interfaces
- The VNA Object Model
- Superseded commands

(Bold Methods or Properties provide access to a child object)

In the following table IPhaseReferenceCalibration is abbreviated to IPhaseRefCal
<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetConnectedPhaseReferences</td>
<td>IPhaseRefCal</td>
<td>Reads the ID strings of the phase references that are currently connected to the VNA USB</td>
</tr>
<tr>
<td>Reset</td>
<td>IPhaseRefCal</td>
<td>Resets all properties associated with the Cal All session to their default values.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalKitType</td>
<td>IPhaseRefCal2</td>
<td>Sets and returns the Cal Kit to be used during the S-parameter portion of a Phase Reference calibration.</td>
</tr>
<tr>
<td>CalSet</td>
<td>IPhaseRefCal</td>
<td>Sets and reads the Cal Set name into which the calibration will be saved.</td>
</tr>
<tr>
<td>GuidedCalibration</td>
<td>IPhaseRefCal</td>
<td>Provides access to the GuidedCal object. Use this to perform the Calibration.</td>
</tr>
<tr>
<td>IncludePort</td>
<td>IPhaseRefCal2</td>
<td>Sets and returns the enable state for the specified port.</td>
</tr>
<tr>
<td>IncludeUnknownMixer</td>
<td>IPhaseRefCal2</td>
<td>Sets and returns the state of Unknown Mixer calibration.</td>
</tr>
<tr>
<td>PhaseReference</td>
<td>IPhaseRefCal</td>
<td>Sets and returns the Phase Reference ID to be used for the Phase Reference calibration.</td>
</tr>
<tr>
<td>SourceAttenuator</td>
<td>IPhaseRefCal</td>
<td>Sets and returns the Source Attenuator setting.</td>
</tr>
<tr>
<td>StartFrequency</td>
<td>IPhaseRefCal</td>
<td>Sets and returns the phase reference cal start frequency.</td>
</tr>
<tr>
<td>StopFrequency</td>
<td>IPhaseRefCal</td>
<td>Sets and returns the phase reference cal stop frequency.</td>
</tr>
<tr>
<td>UnknownMixerInputPower</td>
<td>IPhaseRefCal2</td>
<td>Sets and returns the input power level to the unknown mixer.</td>
</tr>
</tbody>
</table>
UnknownMixerLOFrequency

Sets and returns the LO Frequency of the unknown mixer.

UnknownMixerLOPower

Sets and returns the LO power level to the unknown mixer.

IPhaseReferenceCalibration History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPhaseReferenceCalibration</td>
<td>9.70</td>
</tr>
</tbody>
</table>
PNOP Object

Description

Contains the methods and properties that initiate and return Power Normal Operating Point markers.

Accessing the PNOP Object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")

Dim pnop As PNOP
Set pnop = app.ActiveMeasurement.PNOP
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- PNOP Markers
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SearchPowerNormalOperatingPoint</td>
<td>IPNOP</td>
<td>Initiates a PNOP search</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BackOff</td>
<td>IPNOP Sets and returns the backoff value.</td>
</tr>
<tr>
<td>BackOffGain</td>
<td>IPNOP Returns the BackOffGain result.</td>
</tr>
<tr>
<td>BackOffPIn</td>
<td>IPNOP Returns the BackOffPIn result</td>
</tr>
<tr>
<td>BackOffPOut</td>
<td>IPNOP Returns the BackOffPOut result</td>
</tr>
<tr>
<td>Compression</td>
<td>IPNOP Returns the Compression result</td>
</tr>
<tr>
<td>CompressionMax</td>
<td>IPNOP Returns the Compression Max result</td>
</tr>
<tr>
<td>Gain</td>
<td>IPNOP Returns the Gain result</td>
</tr>
<tr>
<td>GainMax</td>
<td>IPNOP Returns the Gain Max result</td>
</tr>
<tr>
<td>Pin</td>
<td>IPNOP Returns the Pin result</td>
</tr>
<tr>
<td>PinOffset</td>
<td>IPNOP Sets and returns the PinOffset value</td>
</tr>
</tbody>
</table>
PMaxIn  IPNOP  Returns the PMaxIn result
PMaxOut IPNOP  Returns the PMaxOut result
POut  IPNOP  Returns the P Out result

IPNOP History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPNOP</td>
<td>A.09.20</td>
</tr>
</tbody>
</table>
PortExtension Object  Superseded

ALL methods and properties on the PortExtension Object are Superseded with the Fixturing Object.

Description

Contains the methods and properties that control Port Extensions.

Accessing a PortExtension object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim PortExt As PortExtension
Set PortExt = app.PortExtension

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
- Superseded commands

Methods

None

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input A</td>
<td>Sets the Input A extension value.</td>
</tr>
<tr>
<td>Input B</td>
<td>Sets the Input B extension value.</td>
</tr>
<tr>
<td>Input C</td>
<td>Sets the Input C extension value.</td>
</tr>
<tr>
<td>Port 1</td>
<td>Sets the Port 1 extension value.</td>
</tr>
<tr>
<td>Port 2</td>
<td>Sets the Port 2 extension value.</td>
</tr>
<tr>
<td>Port 3</td>
<td>Sets the Port 3 extension value.</td>
</tr>
<tr>
<td>State</td>
<td>Turns Port Extensions ON and OFF.</td>
</tr>
</tbody>
</table>
# IPort Extension History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPort Extension</td>
<td>1.0</td>
</tr>
</tbody>
</table>

1178
Port Object

Description

Provides access to the properties and methods that are used to assign a cable to the port and reset noise data.

Accessing the Port object

Get a handle to the Port object through the Ports Collection.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim oPorts as UncertaintyManager
Set oPorts = app.UncertaintyManager.Ports
Dim oPort1 as Ports(1)
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
- Superseded commands

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>See History</td>
</tr>
<tr>
<td>ResetNoise</td>
<td>IUncertaintyPort</td>
<td>Resets (clears) the characterized noise data for the VNA port.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cable</td>
<td>IUncertaintyPort</td>
<td>Set and return the cable name that is assigned to the port.</td>
</tr>
<tr>
<td>Number</td>
<td>IUncertaintyPort</td>
<td>Returns the VNA port number.</td>
</tr>
</tbody>
</table>

IUncertaintyPort History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IUncertaintyPort</td>
<td>10.40</td>
</tr>
</tbody>
</table>
Ports Collection

Description

A collection object that provides a mechanism for iterating through the Uncertainty Manager Port objects.

Accessing the Ports Collection

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim oPorts as UncertaintyManager
Set oPorts = app.UncertaintyManager.Ports

Dim oPort1 as Ports(1)

See Also:

- Port Object
- UncertaintyManager Object
- Collections in the Analyzer
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CopyNoiseToAllPorts</td>
<td>Copies the characterized noise data associated with the specified port, to all the other ports.</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle to a Port object in the collection.</td>
</tr>
<tr>
<td>ResetNoiseForAllPorts</td>
<td>Clears the characterized noise data for ALL VNA port objects in the Ports collection.</td>
</tr>
<tr>
<td>SelectCableForAllPorts</td>
<td>Selects the name of the cable to be associated with all the ports in the Ports collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of objects in the collection.</td>
</tr>
<tr>
<td>Interface</td>
<td>Introduced with VNA Rev:</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>IUncertaintyPorts</td>
<td>10.40</td>
</tr>
</tbody>
</table>
Power Range Object

Description

These methods and properties provide access to data sheet specified and typical, max and min power levels (in dBm). Max power refers to the maximum leveled source power at the specified port. Min power is calculated by subtracting the power sweep range from the max leveled power. This information is stored by frequency band in a power specification file. This object provides access to the file’s contents and provides an interface to configure the port number and RF signal path of interest.

Power data is available as either the most restrictive value across a range of frequencies (when `RangeGetMaxPower` and `RangeGetMinPower` are used) or for discrete CW frequencies (when `DiscreteGetMaxPower` and `DiscreteGetMinPower` are used).

No measurement of instrument-specific dynamic range is performed; all power levels are equivalent to power data published in device data sheets. Power levels are valid only for measurement configurations where the front panel jumpers are in their standard positions, as originally shipped. Internal source attenuation and any calibrated external path loss/gain due to cables, fixtures, switches or booster amplifiers are not included in the reported min/max leveled power values. It remains the users’ responsibility to transform the reported factory power range data to a value corresponding to the specific calibration plane of their setups.

The power range data files contain both specified min/max leveled power values and the corresponding "typical" values. Some paths, that are not part of the specifications of the instrument may only have typical data. Only the “Specified” power range data is guaranteed for an instrument with an up-to-date calibration certificate.

Accessing the Power Range object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyserName>)
Dim pwrRange As PowerRange
Set pwrRange = app.Capabilities.PowerRange
```

See Also:

- ALC
- Source Unleveled
- Capabilities Object
- The PNA Object Model
### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DiscreteGetMaxPower</td>
<td>IPowreRange</td>
<td>Returns a single max leveled power value (in dBm) indicating the most restrictive maximum for all discrete maximum powers (the minimum of all max leveled powers).</td>
</tr>
<tr>
<td>DiscreteGetMaxPowerArray</td>
<td>IPowreRange</td>
<td>Returns an array of max leveled power values (in dBm), where each element corresponds to the maximum leveled power possible for CW stimulus at the corresponding frequency set by the DiscreteFrequencies property.</td>
</tr>
<tr>
<td>DiscreteGetMinPower</td>
<td>IPowreRange</td>
<td>Returns a single minimum power value (in dBm) indicating the most restrictive minimum for all discrete minimum powers (the maximum of all minimum powers).</td>
</tr>
<tr>
<td>DiscreteGetMinPowerArray</td>
<td>IPowreRange</td>
<td>Returns an array of minimum power values (in dBm), where each element corresponds to the minimum power possible for CW stimulus at the corresponding frequency set by the DiscreteFrequencies property.</td>
</tr>
<tr>
<td>RangeGetMaxPower</td>
<td>IPowreRange</td>
<td>Set minimum of all max leveled power values from RangeStartFrequency to RangeStopFrequency (inclusive).</td>
</tr>
<tr>
<td>RangeGetMinPower</td>
<td>IPowreRange</td>
<td>Set maximum of all min power values from RangeStartFrequency to RangeStopFrequency (inclusive).</td>
</tr>
<tr>
<td>Reset</td>
<td>IPowreRange</td>
<td>Resets all PowerRange properties to default values, as if the instrument had been preset.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DiscreteFrequencies</td>
<td>IPowreRange</td>
<td>An array of frequencies (in Hertz); each element of the array corresponds to a power returned by DiscreteGetMaxPowerArray or DiscreteGetMinPowerArray.</td>
</tr>
<tr>
<td>PathElement</td>
<td>IPowreRange</td>
<td>Given the name of a path element, returns a handle to the PathElement interface corresponding to the given name.</td>
</tr>
</tbody>
</table>
PathElements    IPowreRange    Returns an array with the names of all RF path elements that may be configured.
PortNumber    IPowreRange    Set port number for power data.
PowerRangeType    IPowreRange    Select warranted or typical power performance.
RangeStartFrequency    IPowreRange    Set lower bound of the frequency range (in Hertz).
RangeStopFrequency    IPowreRange    Set upper bound of the frequency range (in Hertz).

ICapabilities History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerRange</td>
<td>12.70</td>
</tr>
</tbody>
</table>

1185
PowerCalRange Object

Description

Contains the methods and properties used to set start and stop frequency, and number of points for an independent power calibration.

Accessing the IndependentPowerCalibration object

For standard S-parameter channels:

```vba
Dim app as AgilentPNA835x.Application
Dim CalMgr
Set CalMgr = App.GetCalManager
Dim CalAll
Set CalAll = CalMgr.CalibrateAllChannels

Dim IndependentPwrCal
Set IndependentPwrCal = CalAll.IndependentPowerCalibration
```

To calibrate an Application channel, see CreateCustomCalEx Method.

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

Methods

None

Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NumberOfPoints</td>
<td>IPowerCalRange</td>
<td>Sets and gets the number of points for range &lt;m&gt; for source port&lt;n&gt;.</td>
</tr>
<tr>
<td>StartFrequency</td>
<td>IPowerCalRange</td>
<td>Sets and gets the start frequency for range &lt;m&gt; for source port&lt;n&gt;.</td>
</tr>
</tbody>
</table>

See History
StopFrequency

IPowerCalRange

Sets and gets the stop frequency for range <m> for source port<n>.

IndependentPowerCalibration History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerCalRange</td>
<td>10.60</td>
</tr>
</tbody>
</table>
**PowerLossSegment Object**

**Description**

Contains the properties describing a segment of the power loss table used in source power calibration.

You can get a handle to one of these segments through the `segments.Item` Method of the PowerLossSegments collection.

**Accessing the PowerLossSegment object**

You can get a handle to one of these segments through `PowerLossSegments.Item(n)`

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", < analyzerName >)

Dim PwrLossSeg As PowerLossSegment
Set PwrLossSeg = app.SourcePowerCalibrator.PowerLossSegments(1)
```

**See Also:**

- About Source Power Cal
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

**Methods**

None

**Properties**

<table>
<thead>
<tr>
<th>Description</th>
<th>Frequency</th>
<th>Loss</th>
<th>SegmentNumber</th>
</tr>
</thead>
<tbody>
<tr>
<td>The frequency (Hz) associated with this segment.</td>
<td></td>
<td>The loss value (dB) associated with this segment.</td>
<td>Returns the number of this segment</td>
</tr>
</tbody>
</table>

IPowerLossSegment History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerLossSegment</td>
<td>2.0</td>
</tr>
</tbody>
</table>
PowerLossSegments Collection

Description

A collection object that provides a mechanism for iterating through the segments of the power loss table used in source power calibration. The power loss table can contain up to 9999 segments.

Accessing the PowerLossSegments collection

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim PwrLossSegs As PowerLossSegments
Set PwrLossSegs = app.SourcePowerCalibrator.PowerLossSegments

See Also:

- PowerLossSegment Object
- Collections in the Analyzer
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a PowerLossSegment object to the collection.</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle to a PowerLossSegment object in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes an object from the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of objects in the collection.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the Parent object (SourcePowerCalibrator) of this collection.</td>
</tr>
</tbody>
</table>
PowerLossSegmentPMAR Object

Description

Contains the properties describing a segment of the power loss table used with a Power Meter as Receiver.

Accessing the PowerLossSegmentPMAR object

You can get a handle to one of these segments through PowerLossSegmentsPMAR.Item(n)

See Also

- Example: Create a PMAR Device and Measurement
- Configure an External Device
- About PMAR
- ExternalDevice Object
- The VNA Object Model

Methods

None

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>The frequency (Hz) associated with this segment.</td>
</tr>
<tr>
<td>Loss</td>
<td>The loss value (dB) associated with this segment.</td>
</tr>
<tr>
<td>SegmentNumber</td>
<td>Returns the number of this segment</td>
</tr>
</tbody>
</table>

IPowerLossSegment History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerLossSegment</td>
<td>9.0</td>
</tr>
</tbody>
</table>
PowerLossSegmentsPMAR Collection

Description

A collection object that provides a mechanism for iterating through the segments of the power loss table used with a Power Meter as Receiver. The power loss table can contain up to 9999 segments.

Accessing the PowerLossSegmentsPMAR collection

- Example: Create a PMAR Device and Measurement

See Also

- Configure an External Device
- About PMAR
- ExternalDevice Object
- The VNA Object Model

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a PowerLossSegmentPMAR object to the collection.</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle to a PowerLossSegmentPMAR object in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes a PowerLossSegmentPMAR object from the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of PowerLossSegmentPMAR objects in the collection.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the Parent object of this collection.</td>
</tr>
</tbody>
</table>

IPowerLossSegments History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerLossSegments</td>
<td>9.0</td>
</tr>
</tbody>
</table>
PowerMeterInterface Object

Description

Contains the properties used to select a power meter and sensor to be used for a source power calibration.

Note: This object replaces the PowerMeterGPIBAddress Property.

Accessing the PowerMeterInterface object

Get a handle to a power meter object using the PowerMeterInterfaces collection.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim pwrMtrInterfaces As PowerMeterInterfaces
Set pwrMtrInterfaces = app.SourcePowerCalibrator.PowerMeterInterfaces
If pwrMtrInterfaces.Count > 0 Then
    Dim pwrMtrInterface As PowerMeterInterface
    Set pwrMtrInterface = pwrMtrInterfaces(1)
    pwrMtrInterface.Path = naUSB
    pwrMtrInterface.Locator = "Agilent Technologies,U2000A,MY12345678"
End If
```

See Also:

- Source Power Calibration
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
### Methods

None

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Specifies the interface to use: GPIB, USB, LAN</td>
</tr>
<tr>
<td>Locator</td>
<td>Specifies the location (address) of the power meter/sensor.</td>
</tr>
</tbody>
</table>

### IPowerMeterInterface History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerMeterInterface</td>
<td>7.50</td>
</tr>
</tbody>
</table>
PowerMeterInterfaces Collection

Description

A collection object that provides a mechanism for accessing the PowerMeterInterface objects.

The collection size is limited to one PowerMeterInterface object. By default, that PowerMeterInterface object refers to GPIB, and to the GPIB address that is currently set for the power meter on that VNA.

The power meter is specified by using the Path property.

Accessing the PowerMeterInterfaces collection

Get a handle to a power meter object using the PowerMeterInterfaces collection.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim pwrMtrInterfaces As PowerMeterInterfaces
Set pwrMtrInterfaces = app.SourcePowerCalibrator.PowerMeterInterfaces
If pwrMtrInterfaces.Count > 0 Then
    Dim pwrMtrInterface As PowerMeterInterface
    Set pwrMtrInterface = pwrMtrInterfaces(1)
    pwrMtrInterface.Path = naUSB
    pwrMtrInterface.Locator = "Agilent Technologies,U2000A,MY12345678"
End If
```

See Also:

- Source Power Calibration
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
## Methods

| Item   | Use to get a handle to a `PowerMeterInterface` object in the collection. |

## Properties

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Count</strong></td>
</tr>
<tr>
<td><strong>Parent</strong></td>
</tr>
</tbody>
</table>

### `IPowerMeterInterfaces` History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>IPowerMeterInterfaces</code></td>
<td>7.50</td>
</tr>
</tbody>
</table>
**PowerSensor Object**

**Description**

Each power sensor connected to the power meter associated with Source Power Calibration will have a PowerSensor object created to represent it. These PowerSensor objects reside in the `PowerSensors` collection within the `SourcePowerCalibrator` object. You cannot directly create PowerSensor objects, but can only retrieve existing ones from the `PowerSensors` collection.

The `PowerSensorCalFactorSegment` object is also accessed through the PowerSensor object. These are accessed through the `CalFactorSegments` collection in the PowerSensor object.

**Accessing a PowerSensor object**

```vbnet
Dim pna As AgilentPNA835x.Application
Set pna = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim powerCalibrator as SourcePowerCalibrator
Dim powerSensor as PowerSensor
Dim calFactorSegment as PowerSensorCalFactorSegment

Set powerCalibrator = pna.SourcePowerCalibrator

' Specify GPIB address of the power meter.
powerCalibrator.PowerMeterGPIBAddress = 13

' Each time the PowerSensors collection is accessed, the power meter is queried to
determine which channels have sensors attached. The collection is updated
accordingly.
If powerCalibrator.PowerSensors.Count > 0 Then
  ' If channel B of the meter has a sensor attached but channel A does not, then
  ' element 1 of the
  ' collection is sensor B. Whenever channel A has a sensor, sensor A will be element
  ' 1.
  Set powerSensor = powerCalibrator.PowerSensors(1)
  ' Insert one new PowerSensorCalFactorSegment at the beginning of the collection
  ' (index 1).
  powerSensor.CalFactorSegments.Add(1)
  ' Assign our variable to refer to that object.
  Set calFactorSegment = powerSensor.CalFactorSegments(1)

  ' Set property values for that object.
  calFactorSegment.Frequency = 300000
  ' frequency in Hz
  calFactorSegment.CalFactor = 98
  ' cal factor in percent
```
See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

**Methods or Properties provide access to a child object**

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CalFactorSegments</strong></td>
<td>IPowerSensor</td>
<td>Collection for iterating through the segments of a power sensor cal factor table.</td>
</tr>
<tr>
<td>LimitFrequency</td>
<td>IPowerSensor2</td>
<td>Enable or disable the use of the power meter min and max frequencies.</td>
</tr>
<tr>
<td>MaximumFrequency</td>
<td>IPowerSensor</td>
<td>Maximum usable frequency (Hz) specified for this power sensor.</td>
</tr>
<tr>
<td>MinimumFrequency</td>
<td>IPowerSensor</td>
<td>Minimum usable frequency (Hz) specified for this power sensor.</td>
</tr>
<tr>
<td><strong>PowerLossSegments</strong></td>
<td>IPowerSensor2</td>
<td>Collection for iterating through the segments of the power loss table used in source power calibration.</td>
</tr>
<tr>
<td>PowerMeterChannel</td>
<td>IPowerSensor</td>
<td>Identifies which power sensor this object corresponds to (or which channel of the power meter the sensor is connected to).</td>
</tr>
<tr>
<td><strong>PowerSensorUncertainty</strong></td>
<td>IPowerSensor4</td>
<td>Provides access to the PowerSensorUncertainty object.</td>
</tr>
<tr>
<td>ReadingsPerPoint</td>
<td>IPowerSensor2</td>
<td>Allows for settling of the power sensor READINGS.</td>
</tr>
<tr>
<td>ReadingsTolerance</td>
<td>IPowerSensor2</td>
<td>Allows for settling of the power sensor READINGS.</td>
</tr>
<tr>
<td>ReferenceCalFactor</td>
<td>IPowerSensor</td>
<td>Reference cal factor (%) associated with this power sensor.</td>
</tr>
<tr>
<td>SensorIndex</td>
<td>IPowerSensor2</td>
<td>Sets the power sensor channel (1 or 2) to be used.</td>
</tr>
</tbody>
</table>
**UseInternalCalFactors**  
IPowerSensor3  Provides access to the PowerSensorAsReceiver object.

**UsePowerLossSegments**  
IPowerSensor2  Specifies if subsequent power readings will use of the loss table.

### IPowerSensor History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerSensor</td>
<td>2.0</td>
</tr>
<tr>
<td>IPowerSensor2</td>
<td>13.50</td>
</tr>
<tr>
<td>IPowerSensor3</td>
<td>13.50</td>
</tr>
<tr>
<td>IPowerSensor4</td>
<td>13.50</td>
</tr>
</tbody>
</table>
**PowerSensors Collection**

**Description**

A collection object that provides a mechanism for iterating through the PowerSensor objects which are connected to the power meter. Each time this collection object is accessed, the power meter is queried to determine how many sensors are connected to it. The collection size and order of objects is then adjusted accordingly before the requested method or property operation is performed. The power meter is specified by using the `PowerMeterGPIBAddress` property of the `SourcePowerCalibrator` object.

**Accessing the PowerSensors Collection**

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim PwrSensors As PowerSensors
Set PwrSensors = app.SourcePowerCalibrator.PowerSensors
```

**See Also:**

- [PowerSensor Object](#)
- [Collections in the Analyzer](#)
- [The VNA Object Model](#)
- [Example Programs](#)

**Methods**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Item</strong> Use to get a handle to a PowerSensor object in the collection.</td>
</tr>
</tbody>
</table>

**Properties**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Count</strong> Returns the number of objects in the collection.</td>
</tr>
<tr>
<td><strong>Parent</strong> Returns a handle to the Parent object (SourcePowerCalibrator) of this collection.</td>
</tr>
</tbody>
</table>
PowerSensorCalFactorSegment Object

Description

Contains the properties describing a segment of a power sensor cal factor table.

Accessing the PowerSensorCalFactorSegment object

You can get a handle to one of these segments through CalFactorSegments.Item(n)

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim calFactSeg As CalFactorSegments
Set calFactSeg = app.SourcePowerCalibrator.PowerSensors(1).CalFactorSegments(1)
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

Methods

None

Properties | Description
---|---
Frequency | The frequency (Hz) associated with this segment.
  | Shared with the PowerLossSegment Object
CalFactor | The cal factor (%) associated with this segment.
SegmentNumber | Returns the number of this segment.
  | Shared with the PowerLossSegment Object

IPowerSensorCalFactorSegment History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerSensorCalFactorSegment</td>
<td>2.0</td>
</tr>
</tbody>
</table>
PowerSensorCalFactorSegmentPMAR Object

Description
Contains the properties describing a segment of a power sensor cal factor table that is used with Power Meter as Receiver.

Accessing the PowerSensorCalFactorSegment PMAR object
You can get a handle to one of these segments through CalFactorSegmentsPMAR.Item(n).

See Also

- **Example:** Create a PMAR Device and Measurement
- Configure an External Device
- ExternalDevice Object
- The VNA Object Model

Methods
None

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>The frequency (Hz) associated with this segment.</td>
</tr>
<tr>
<td>CalFactor</td>
<td>The cal factor (%) associated with this segment.</td>
</tr>
<tr>
<td>SegmentNumber</td>
<td>Returns the number of this segment.</td>
</tr>
</tbody>
</table>

IPowerSensorCalFactorSegment History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerSensorCalFactorSegment</td>
<td>9.0</td>
</tr>
</tbody>
</table>
PowerSensorAsReceiver Object

Description

Provides the settings for configuring a Power Meter to be used as a VNA Receiver (PMAR).

Accessing a PowerSensorAsReceiver object

This object is accessed through ExternalDevice.ExtendedProperties. When an external device is added to the ExternalDevices collection, and the DeviceType property is set to Power Meter, then ExtendedProperties is used to get a handle to this object.

```vba
externalDevices.Add "NewPMAR"

dim newExternalDevice
Set newExternalDevice = externalDevices.Item("NewPMAR")
newExternalDevice.DeviceType = "Power Meter"

dim PMAR
Set PMAR = newExternalDevice.ExtendedProperties
PMAR.ReadingsPerPoint = 10
```

See Also:

- **Example**: Create a PMAR Device and Measurement
- Configure an External Device
- ExternalDevice Object
- The VNA Object Model

(Bold Methods or Properties provide access to a child object)
### Methods

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CalFactorSegments</strong></td>
</tr>
<tr>
<td><strong>LimitFrequency</strong></td>
</tr>
<tr>
<td><strong>MaximumFrequency</strong></td>
</tr>
<tr>
<td><strong>MinimumFrequency</strong></td>
</tr>
<tr>
<td><strong>PowerLossSegments</strong></td>
</tr>
<tr>
<td><strong>PowerMeterChannel</strong></td>
</tr>
<tr>
<td><strong>ReadingsPerPoint</strong></td>
</tr>
<tr>
<td><strong>ReadingsTolerance</strong></td>
</tr>
<tr>
<td><strong>ReferenceCalFactor</strong></td>
</tr>
<tr>
<td><strong>SensorIndex</strong></td>
</tr>
<tr>
<td><strong>UseInternalCalFactors</strong></td>
</tr>
<tr>
<td><strong>UsePowerLossSegments</strong></td>
</tr>
</tbody>
</table>

### IPowerSensorAsReceiver History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerSensorAsReceiver</td>
<td>9.0</td>
</tr>
</tbody>
</table>
PowerSensorUncertainty Object

Description

The PowerSensorUncertainty object allows you to set up power uncertainty on a power meter.

Accessing the PowerSensorUncertainty Object

You can obtain a handle to a PowerSensorUncertainty object through the PowerSensor object.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim PwrSensUncert As PowerSensorUncertainty
```

See Also

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

### Methods

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PowerForBestAccuracy</td>
<td>IPowerSensorUncertainty</td>
</tr>
<tr>
<td>PowerMtrReadingUncertainty</td>
<td>IPowerSensorUncertainty</td>
</tr>
<tr>
<td>UncertaintyFile</td>
<td>IPowerSensorUncertainty</td>
</tr>
<tr>
<td>UncertaintyModel</td>
<td>IPowerSensorUncertainty</td>
</tr>
<tr>
<td>UncertaintyModelCatalog</td>
<td>IPowerSensorUncertainty</td>
</tr>
</tbody>
</table>
IPowerSensor History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerSensorUncertainty</td>
<td>13.50</td>
</tr>
</tbody>
</table>

Preferences Object

Description

Sets the preferences for the behavior of several properties.

Accessing the Preferences object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim pref As Preferences
Set pref = app.Preferences
```

See Also:

- VNA Preferences
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

(Bold Methods or Properties provide access to a child object)

Methods

- **RestoreDefaults**

  IPreferences9

  Restores preference settings to their factory defaults.

Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuxTriggerScopeIsGlobal</td>
<td>IPreferences5</td>
<td>Sets the External Trigger OUT behavior to have either Global or Channel scope.</td>
</tr>
<tr>
<td>BandwidthSearch</td>
<td>IPreferences18</td>
<td>Sets the bandwidth search preference to start a bandwidth or notch search in either peak or marker mode.</td>
</tr>
<tr>
<td>CitiContents</td>
<td>IPreferences</td>
<td>Specifies the contents of subsequent citifile saves. <strong>Superseded</strong> with SaveData</td>
</tr>
</tbody>
</table>

See History
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CitiFormat</strong></td>
<td>Specifies the format of subsequent citifile saves. <strong>Superseded</strong> with <strong>SaveData</strong></td>
</tr>
<tr>
<td><strong>ConfirmPreset</strong></td>
<td>IPreferences18  Set and return preset confirmation.</td>
</tr>
<tr>
<td><strong>DisplayColors</strong></td>
<td>IPreferences10  Provides access to the ComColors Object.</td>
</tr>
<tr>
<td><strong>EnableSourceUnleveledEvents</strong></td>
<td>IPreferences6   Specifies whether or not to report Source Unleveled errors as system events.</td>
</tr>
<tr>
<td><strong>ExternalDeviceDeActivatePolicy</strong></td>
<td>IPreferences10  External Devices remain activated or are de-activated when the VNA is Preset or when a Instrument State is recalled.</td>
</tr>
<tr>
<td><strong>FrequencyOffsetRangeForCalComputations</strong></td>
<td>IPreferences10  Specifies the FOM frequency range to use when performing calibration.</td>
</tr>
<tr>
<td><strong>IMDECalExtrapolation</strong></td>
<td>IPreferences14  Allow ECal beyond stop frequency for IMD apps.</td>
</tr>
<tr>
<td><strong>InterpolateMemoryIsDefault</strong></td>
<td>IPreferences17  Sets and reads the state of the memory data interpolation default preference.</td>
</tr>
<tr>
<td><strong>LegacyGroupDelayApertureMath</strong></td>
<td>IPreferences19  Sets the group delay aperture to use the legacy computation method.</td>
</tr>
<tr>
<td><strong>MarkCoupControlsMkrState</strong></td>
<td>IPreferences15  Coupled Marker controls marker state.</td>
</tr>
<tr>
<td><strong>MarkCoupMethPresetIsChan</strong></td>
<td>IPreferences15  Coupled Marker state at Preset.</td>
</tr>
<tr>
<td><strong>MarkCoupPresetIsOn</strong></td>
<td>IPreferences15  Coupled Marker Method at Preset.</td>
</tr>
<tr>
<td><strong>OffsetReceiverAttenuator</strong></td>
<td>IPreferences6   Mathematically offset the test port receiver.</td>
</tr>
<tr>
<td><strong>OffsetSourceAttenuator</strong></td>
<td>IPreferences6   Mathematically offset the reference receiver.</td>
</tr>
<tr>
<td><strong>Port1NoiseTunerSwitchPresetsToExternal</strong></td>
<td>IPreferences8   Sets default setting for Noise Figure switch.</td>
</tr>
<tr>
<td><strong>PowerOnDuringRetraceMode</strong></td>
<td>IPreferences4   Specify whether to turn RF power ON or OFF during a retrace for single-band frequency or segment sweeps ONLY.</td>
</tr>
<tr>
<td><strong>PowerSweepRetracePowerMode</strong></td>
<td>IPreferences3   At the end of a power sweep, specifies whether to maintain source power at the start or stop power level.</td>
</tr>
<tr>
<td>Preference</td>
<td>Set</td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>PreferInternalTriggerOnChannelSingle</td>
<td>IPreferences2</td>
</tr>
<tr>
<td>PreferInternalTriggerOnUnguidedCal</td>
<td>IPreferences2</td>
</tr>
<tr>
<td>PrefSourcePowerCalFromCalset</td>
<td>IPreferences</td>
</tr>
<tr>
<td>PresetPowerState</td>
<td>IPreferences11</td>
</tr>
<tr>
<td>PrintColors</td>
<td>IPreferences10</td>
</tr>
<tr>
<td>RecallSoftkeysMostRecent</td>
<td>IPreferences13</td>
</tr>
<tr>
<td>RemoteCalStoragePreference</td>
<td>IPreferences7</td>
</tr>
<tr>
<td>ReportReceiverOverload</td>
<td>IPreferences12</td>
</tr>
<tr>
<td>RFOffOnReceiverOverload</td>
<td>IPreferences12</td>
</tr>
<tr>
<td>ShowKeysToolbarAtPowerOn</td>
<td>IPreferences16</td>
</tr>
<tr>
<td>ShowQuickStartOnPreset</td>
<td>IPreferences13</td>
</tr>
<tr>
<td>SingleMarkerSearch</td>
<td>IPreferences18</td>
</tr>
<tr>
<td>SnPFormat</td>
<td>IPreferences</td>
</tr>
<tr>
<td>TreatMkr10AsReference</td>
<td>IPreferences15</td>
</tr>
<tr>
<td>TwoPointGroupDelayAperture</td>
<td>IPreferences11</td>
</tr>
<tr>
<td>Interface</td>
<td>Introduced with VNA Rev:</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>IPreferences</td>
<td>4.0</td>
</tr>
<tr>
<td>IPreferences2</td>
<td>6.0</td>
</tr>
<tr>
<td>IPreferences3</td>
<td>7.2</td>
</tr>
<tr>
<td>IPreferences4</td>
<td>6.04</td>
</tr>
<tr>
<td>IPreferences5</td>
<td>7.10</td>
</tr>
<tr>
<td>IPreferences6</td>
<td>7.20</td>
</tr>
<tr>
<td>IPreferences7</td>
<td>7.21</td>
</tr>
<tr>
<td>IPreferences8</td>
<td>8.0</td>
</tr>
<tr>
<td>IPreferences9</td>
<td>8.2</td>
</tr>
<tr>
<td>IPreferences10</td>
<td>9.0</td>
</tr>
<tr>
<td>IPreferences11</td>
<td>9.20</td>
</tr>
<tr>
<td>IPreferences12</td>
<td>9.30</td>
</tr>
<tr>
<td>IPreferences13</td>
<td>9.50</td>
</tr>
<tr>
<td>IPreferences14</td>
<td>10.0</td>
</tr>
<tr>
<td>IPreferences15</td>
<td>10.40</td>
</tr>
<tr>
<td>IPreferences16</td>
<td>10.49</td>
</tr>
<tr>
<td>IPreferences17</td>
<td>10.49</td>
</tr>
<tr>
<td>IPreferences18</td>
<td>12.70</td>
</tr>
<tr>
<td>IPreferences19</td>
<td>13.80</td>
</tr>
</tbody>
</table>
### PSaturation Object

#### Description

Contains the methods and properties that initiate a Power Saturation marker search and returns PSAT data.

#### Accessing the PSaturation Object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")
Dim PSat As PSaturation
Set PSat = app.ActiveMeasurement.PSaturation
```

#### See Also:

- VNA Automation Interfaces
- The VNA Object Model
- PSaturation Markers
- Example Programs

#### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SearchPowerSaturation</td>
<td>IPSaturation</td>
<td>Initiates a Power Saturation marker search.</td>
</tr>
</tbody>
</table>

#### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CompressionMax</td>
<td>IPSaturation</td>
<td>Returns the Compression Max result of a PSat marker search.</td>
</tr>
<tr>
<td>CompressionSaturation</td>
<td>IPSaturation</td>
<td>Returns the Compression Saturation result of a PSat marker search.</td>
</tr>
<tr>
<td>GainLinear</td>
<td>IPSaturation</td>
<td>Returns the Linear Gain result of a PSat marker search.</td>
</tr>
<tr>
<td>GainMax</td>
<td>IPSaturation</td>
<td>Returns the GainMax result of a PSAT marker search.</td>
</tr>
<tr>
<td>GainSaturation</td>
<td>IPSaturation</td>
<td>Returns the GainSaturation result of a PSAT marker search.</td>
</tr>
<tr>
<td>Pin</td>
<td>IPSaturation</td>
<td>Returns the Pin result of a PSAT marker search.</td>
</tr>
<tr>
<td>PMaxBackOff</td>
<td>IPSaturation</td>
<td>Sets and returns the backoff value used to calculate various PSAT parameters.</td>
</tr>
</tbody>
</table>
PMaxIn IPSaturation Returns the PMaxIn result of a PSAT marker search.
PMaxOut IPSaturation Returns the PMaxOut result of a PSAT marker search.
POut Property IPSaturation Returns the POut result of a PSAT marker search.

IPSaturation History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPSaturation</td>
<td>A.09.20</td>
</tr>
</tbody>
</table>
PulseGenerator Object

Description

Contains the properties for configuring the five internal pulse generators in the PNA-X.

Learn more about the PNA-X Pulse Generators.

Accessing the PulseGenerator object

```vbnet
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim pulse as PulseGenerator
Set pulse = chan.PulseGenerator
```

Each pulse generator is specified in the Pulse Generator properties.

External Pulse Generators

Beginning with A.09.50, External Pulse Generators can be used with the Integrated Pulse Application.

Use `chan.PulseGeneratorNames` and `chan.PulseGeneratorID` to refer to the external pulse generator when setting properties on this (PulseGenerator) Object.

Pulse Definitions

- D = Delay; the time before each pulse begins
- W = Width; the time the pulse is ON
- P = Period; one complete pulse cycle
- W/P = Duty Cycle; the ratio of pulse ON/OFF

**Important:** If \( D + W \) is greater than \( P \), then undefined VNA behavior results. There is NO error message or warning.
### Methods

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>See History</td>
<td></td>
</tr>
<tr>
<td>Delay</td>
<td>IPulsedGenerator</td>
</tr>
<tr>
<td>DelayIncrement</td>
<td>IPulsedGenerator</td>
</tr>
<tr>
<td>EnableOffsetDelays</td>
<td>IPulsedGenerator5</td>
</tr>
<tr>
<td>ADCDelay</td>
<td>IPulsedGenerator5</td>
</tr>
<tr>
<td>Invert</td>
<td>IPulsedGenerator4</td>
</tr>
<tr>
<td>ModulatorDelay</td>
<td>IPulsedGenerator5</td>
</tr>
<tr>
<td>Period</td>
<td>IPulsedGenerator</td>
</tr>
<tr>
<td>Pulse4OutAsADCActivity</td>
<td>IPulsedGenerator5</td>
</tr>
<tr>
<td>PulseTimingDevice</td>
<td>IPulsedGenerator6</td>
</tr>
<tr>
<td>State</td>
<td>IPulsedGenerator</td>
</tr>
<tr>
<td>SubPointTrigger</td>
<td>IPulsedGenerator2</td>
</tr>
</tbody>
</table>
**TriggerInPolarity**  
IPulsedGenerator3  
Sets the polarity of trigger to which the internal pulse generators will respond when being externally triggered.

**TriggerInType**  
IPulsedGenerator3  
Sets the type of trigger to which the internal pulse generators will respond when being externally triggered.

**Width**  
IPulsedGenerator  
Sets the pulse width for the specified pulse generator.

### IPulseGenerator History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPulseGenerator</td>
<td>7.2</td>
</tr>
<tr>
<td>IPulseGenerator2</td>
<td>8.55.09</td>
</tr>
<tr>
<td>IPulseGenerator3</td>
<td>9.10</td>
</tr>
<tr>
<td>IPulseGenerator4</td>
<td>9.33</td>
</tr>
<tr>
<td>IPulseGenerator5</td>
<td>13.50</td>
</tr>
<tr>
<td>IPulseGenerator6</td>
<td>13.60</td>
</tr>
</tbody>
</table>
PulseMeasControl Object

Description

Contains the properties for configuring pulse measurements in the PNA-X.

Some of these settings require Opt H08 / S93026A/B.

Learn about Integrated Pulse Application.

Accessing the PulseMeasControl object

```vba
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim pulse as PulseMeasControl
Set pulse = chan.PulseMeasControl
```

The following shows how to make two new settings introduced with A.09.50:

- **Primary Pulse Trigger** - to select an external pulse generator as the primary trigger, use `IPathElement.Value` and using “PulseTrigInput” = “Internal” or “External”. However, there is no way to tell which external pulse generator is selected.

- To select an external pulse generator for a receiver, use `IPathElement.Value` commands. For example, to select an external pulse generator for RCVR A, send "IFGateA","RearPanel".

See Also:

- IF Path Block Diagram.
- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoCWSweepTime</td>
<td>IPulseMeasControl2</td>
<td>Sets the state of automatic CW sweep time in Pulse Profile mode.</td>
</tr>
<tr>
<td>AutoDetection</td>
<td>IPulseMeasControl</td>
<td>Automatically or manually set pulse mode (Narrowband or Wideband)</td>
</tr>
<tr>
<td>AutoIFBandWidth</td>
<td>IPulseMeasControl</td>
<td>Autoselect the IFBW</td>
</tr>
<tr>
<td>AutoIFGain</td>
<td>IPulseMeasControl</td>
<td>For future use.</td>
</tr>
<tr>
<td>AutoOptimizePRF</td>
<td>IPulseMeasControl</td>
<td>Auto-optimize pulse clock period</td>
</tr>
<tr>
<td>AutoPulseTiming</td>
<td>IPulseMeasControl</td>
<td>Autoselect Width and Delay</td>
</tr>
<tr>
<td>AutoSelectPulseGen</td>
<td>IPulseMeasControl</td>
<td>Autoselect Pulse Generators</td>
</tr>
<tr>
<td>PrimaryFrequency</td>
<td>IPulseMeasControl2</td>
<td>Sets the pulse repetition frequency (PRF) for ALL internal pulse generators.</td>
</tr>
<tr>
<td>PrimaryPeriod</td>
<td>IPulseMeasControl2</td>
<td>Sets the period for ALL internal pulse generators.</td>
</tr>
<tr>
<td>PrimaryWidth</td>
<td>IPulseMeasControl2</td>
<td>Sets the pulse width for ALL internal pulse generators.</td>
</tr>
<tr>
<td>Parent</td>
<td>IPulseMeasControl</td>
<td>Returns the channel object</td>
</tr>
<tr>
<td>PulseMeasMode</td>
<td>IPulseMeasControl</td>
<td>Select Pulse Measurement selection</td>
</tr>
<tr>
<td>PulseProfileStart</td>
<td>IPulseMeasControl3</td>
<td>Sets and returns the start time of the profile pulse.</td>
</tr>
<tr>
<td>PulseProfileStop</td>
<td>IPulseMeasControl3</td>
<td>Sets and returns the stop time of the profile pulse.</td>
</tr>
<tr>
<td>SoftwareGateState</td>
<td>IPulseMeasControl2</td>
<td>This setting is used for troubleshooting purposes.</td>
</tr>
<tr>
<td>WideBandDetectionState</td>
<td>IPulseMeasControl</td>
<td>Select Narrowband or Wideband pulse detection.</td>
</tr>
</tbody>
</table>

**IPulseMeasControl History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPulseMeasControl</td>
<td>9.2</td>
</tr>
<tr>
<td>IPulseMeasControl2</td>
<td>9.40</td>
</tr>
<tr>
<td>IPulseMeasControl3</td>
<td>10.45</td>
</tr>
</tbody>
</table>
RxLevelingConfiguration Object

Description

Contains the properties for configuring Receiver Leveling in the VNA.

Accessing the RxLevelingConfiguration object

```
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim RxLevel as RxLevelingConfiguration
Set RxLevel = chan.RxLevelingConfiguration
```

See Also:

- About Receiver Leveling
- VNA Automation Interfaces
- The VNA Object Model
- Example Program

Methods

None

Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FastMode</td>
<td>IRxLevelingConfiguration</td>
<td>Select separate IFBW for leveling sweeps</td>
</tr>
<tr>
<td>FrequencyType</td>
<td>IRxLevelingConfiguration3</td>
<td>Frequency range to use for receiver leveling.</td>
</tr>
<tr>
<td>IterationNumber</td>
<td>IRxLevelingConfiguration</td>
<td>Max number of leveling sweeps</td>
</tr>
<tr>
<td>LastLevelingAsSPC</td>
<td>IRxLevelingConfiguration2</td>
<td>Turn ON Source Power Cal using latest correction data</td>
</tr>
<tr>
<td>LevelingIFBW</td>
<td>IRxLevelingConfiguration</td>
<td>IFBW for leveling sweeps</td>
</tr>
<tr>
<td>PowerMax</td>
<td>IRxLevelingConfiguration</td>
<td>Max power for safe mode</td>
</tr>
</tbody>
</table>

See History
PowerMin  IRxLevelingConfiguration  Min power for safe mode
PowerOffset  IRxLevelingConfiguration  Offset power for external components
PowerStep  IRxLevelingConfiguration  Power step for safe mode
ReferenceReceiver  IRxLevelingConfiguration  Select a receiver
SafeMode  IRxLevelingConfiguration  Enable safe mode
State  IRxLevelingConfiguration  Enable receiver leveling
Tolerance  IRxLevelingConfiguration  Tolerance value for leveling sweeps

**ReceiverLevelingConfiguration History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRxLevelingConfiguration</td>
<td>8.5</td>
</tr>
<tr>
<td>IRxLevelingConfiguration2</td>
<td>9.30</td>
</tr>
<tr>
<td>IRxLevelingConfiguration3</td>
<td>9.50</td>
</tr>
</tbody>
</table>
SCPIStringParser Object

Description

Provides the ability to send a SCPI command from within the COM command. The two commands differ in the following ways:

**Execute** - will not return an error unless the Execute command itself fails, which is unlikely. Otherwise, you are required to read the SCPI error queue for errors that were caused by the SCPI command. The Execute command operates with minimal interference between you, the programmer, and the SCPI parser. It does not presume how you want to handle errors: handle by ignore, handle by reading the status byte, etc. This command was defined because automation engines like VB throw runtime errors when a COM method returns a failed HRESULT.

**Parse** - parses the input command, and then reads the SCPI error queue until the queue is empty. If the queue contains errors, Parse returns a failed HRESULT (E_NA_BAD_SCPI_EXECUTE). It then creates an IErrorInfo object and bundles the error numbers and descriptions into the error object. This object is available so that you can detect the failed HRESULT and interrogate the errorInfo object for more details.

The SCPIStringParser Methods can NOT be used with:

- SCPI Status Reporting. However, the *OPC? will work.
- Transferring Binary (block) data with commands such as: MMEM:TRAN or CALC:DATA with Format:Data set to Real32 or Real64

Accessing the ScpiStringParser object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim SCPI As IScpiStringParser
Set SCPI = app.ScpiStringParser
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
- See an example of how to return error information when using the Parse method.
### Methods

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parse</td>
<td>ISCPIStringParser</td>
<td>Provides the ability to send a SCPI command from within the COM command.</td>
</tr>
<tr>
<td>Execute</td>
<td>ISCPIStringParser2</td>
<td>Does not convert scpi errors. Use :SYST:ERR?</td>
</tr>
</tbody>
</table>

### Properties

None

### History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISCPIStringParser</td>
<td>1.0</td>
</tr>
<tr>
<td>ISCPIStringParser2</td>
<td>3.0</td>
</tr>
</tbody>
</table>
Segment Object

Description
Contains the methods and properties that affect a sweep segment.

Note: All of these properties are shared with at least one of the following objects: Channel, Cal Set, PowerSensorCalFactorSegment, or PowerLossSegment.

Accessing a Segment object and setting Segment Properties
You can get a handle to a sweep segment through the segments collection.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim segs As ISegments
Set segs = app.ActiveChannel.Segments

segs.Add(1)
segs(1).NumberOfPoints = 30
```

See Also:
- VNA Automation Interfaces
- The VNA Object Model
- Segment Sweep
- Example Programs

Methods
None

Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>centerFrequency</td>
<td>ISegment</td>
<td>Sets or returns the center frequency of the segment.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Channel Object</td>
</tr>
<tr>
<td>Delay</td>
<td>ISegment3</td>
<td>Sets or returns the delay value.</td>
</tr>
</tbody>
</table>
DwellTime  ISegment  Dwell time value.
Shared with the Channel Object

FrequencySpan  ISegment  Sets or returns the frequency span of the segment.
Shared with the Channel Object

IFBandwidth  ISegment  Sets or returns the IF Bandwidth of the segment.
Shared with the Channel Object and Cal Set object.

NoiseFigureBW  ISegment4  Sets or returns the noise figure bandwidth.

NumberOfPoints  ISegment  Sets or returns the Number of Points of the segment.
Shared with the Channel Object

PortIFBandwidth  ISegment3  Sets or returns the port IF bandwidth.

SADATAThreshold  ISegment5  Sets or returns the SA data threshold.

SAMTReference  ISegment4  Sets or returns the SA multitone reference.

SAVectorAverage  ISegment4  Sets or returns the SA vector average.

SAVideoBandwidth  ISegment5  Sets or returns the SA video bandwidth.

SegmentNumber  ISegment  Returns the number of the current segment.

ShiftLO  ISegment3  Enable or disable shift LO.

StartFrequency  ISegment  Sets or returns the start frequency of the segment.
Shared with the Channel Object

State  ISegment  Turns On or OFF a segment.

StopFrequency  ISegment  Sets or returns the stop frequency of the segment.
Shared with the Channel Object

SweepMode  ISegment3  Sets or returns the sweep mode.

SweepTime  ISegment2  Sets or returns the sweep time of the segment.
Shared with the Channel Object

TestPortPower  ISegment  Sets or returns the RF power level of the segment.
Shared with the Channel Object
### ISegment History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISegment</td>
<td>1.0</td>
</tr>
<tr>
<td>ISegment2</td>
<td>7.1</td>
</tr>
<tr>
<td>ISegment3</td>
<td>12.85</td>
</tr>
<tr>
<td>ISegment4</td>
<td>13.50</td>
</tr>
<tr>
<td>ISegment5</td>
<td>13.50</td>
</tr>
</tbody>
</table>
Segments Collection

Description

The segment collection provides a mechanism for iterating through the sweep segments of a channel. Sweep segments are a potentially faster method of sweeping the analyzer through only the frequencies of interest. Learn more about Segment Sweep.

Accessing the Segments collection

There are two paths to the Segments Collection:

1. From the Channel object
2. From the FOMRange object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim segs As ISegments
Set segs = app.ActiveChannel.Segments

or

Set segs = app.ActiveChannel.FOM.FOMRange(1).Segments
```

See Also:

- Segment Object to learn how to set the properties for individual segments.
- Collections in the Analyzer
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>ISegments</td>
<td>Adds an item to either the Segments collection.</td>
</tr>
<tr>
<td>ExportCSVfile</td>
<td>ISegments6</td>
<td>Export CSV file</td>
</tr>
<tr>
<td>GetAllSegments</td>
<td>ISegments5</td>
<td>Downloads a segment table from the VNA</td>
</tr>
<tr>
<td>ImportCSVfile</td>
<td>ISegments6</td>
<td>Import CSV file</td>
</tr>
<tr>
<td>Properties</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Item ISegments</td>
<td>Use to get a handle to a segment in the collection.</td>
<td></td>
</tr>
<tr>
<td>Remove ISegments</td>
<td>Removes an item from a collection of objects.</td>
<td></td>
</tr>
<tr>
<td>SetAllSegments ISegments2</td>
<td>Uploads a segment table to the VNA.</td>
<td></td>
</tr>
</tbody>
</table>

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowArbitrarySegments ISegments3</td>
<td>Enables the setup of arbitrary segment sweep</td>
</tr>
<tr>
<td>Count ISegments</td>
<td>Returns the number of items in a collection of objects.</td>
</tr>
<tr>
<td>DelayOption ISegments6</td>
<td>Enables delay.</td>
</tr>
<tr>
<td>IFBandwidthOption ISegments</td>
<td>Enables the IFBandwidth to be set on individual sweep segments.</td>
</tr>
<tr>
<td>NoiseFigureBWOption ISegments7</td>
<td>Enables noise figure bandwidth setting.</td>
</tr>
<tr>
<td>Parent ISegments</td>
<td>Returns a handle to the current naNetworkAnalyzer application.</td>
</tr>
<tr>
<td>PortIFBandwidthOption ISegments6</td>
<td>Enables port IF bandwidth.</td>
</tr>
<tr>
<td>SADataThresholdOption ISegments8</td>
<td>Specifies whether SA Data Threshold can be set independently for each segment.</td>
</tr>
<tr>
<td>SAMTReferenceFreqOption ISegments7</td>
<td>Specifies whether SA Reference Tone can be set independently for each segment.</td>
</tr>
<tr>
<td>SAVectorAverageOption ISegments7</td>
<td>Specifies whether SA Vector Averaging can be set independently for each segment.</td>
</tr>
<tr>
<td>SAVideoAverageOption ISegments8</td>
<td>Specifies whether SA Video Bandwidth can be set independently for each segment.</td>
</tr>
<tr>
<td>ShiftLOOption ISegments6</td>
<td>Sets or returns the Shift LO state of each segment in the segment sweep table.</td>
</tr>
<tr>
<td>SourcePowerOption ISegments</td>
<td>Enables setting the Source Power for a segment.</td>
</tr>
<tr>
<td>SweepModeOption ISegments6</td>
<td>Enables analog or stepped sweep mode.</td>
</tr>
<tr>
<td>SweepTimeOption ISegments4</td>
<td>Enables the Sweep time or Dwell time to be set independently on sweep segments.</td>
</tr>
</tbody>
</table>

**ISegments History**

1229
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISegments</td>
<td>1.0</td>
</tr>
<tr>
<td>ISegments2</td>
<td>3.5</td>
</tr>
<tr>
<td>ISegments3</td>
<td>4.2</td>
</tr>
<tr>
<td>ISegments4</td>
<td>7.1</td>
</tr>
<tr>
<td>ISegments5</td>
<td>8.6</td>
</tr>
<tr>
<td>ISegments6</td>
<td>12.85</td>
</tr>
<tr>
<td>ISegments7</td>
<td>13.50</td>
</tr>
<tr>
<td>ISegments8</td>
<td>13.50</td>
</tr>
</tbody>
</table>
SignalProcessingModuleFour Object

Description

Contains the properties for configuring the DSP (digital filters) in the PNA-X.

See the entire IF Path Block diagram.

DSP Version 5 Notes

Programs that control these settings, or state files that are saved, will yield different results when run or recalled on VNAs with DSP 4 versions versus DSP 5 Versions.

Stage 2 settings are IGNORED with DSP 5 Versions.

Learn more about DSP Versions

Accessing the SignalProcessingModuleFour object

Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim digFilter as SignalProcessingModuleFour
Set digFilter = chan.SignalProcessingModuleFour

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- About PNA-X Pulse Capabilities
- Example Programs

Methods

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
<tr>
<td>Properties</td>
</tr>
<tr>
<td>-----------------------------</td>
</tr>
<tr>
<td>ADCCaptureMode</td>
</tr>
<tr>
<td>FilterErrors</td>
</tr>
<tr>
<td>FilterMode</td>
</tr>
<tr>
<td>Stage1Coefficients</td>
</tr>
<tr>
<td>Stage1Frequency</td>
</tr>
<tr>
<td>Stage1MaximumCoefficient</td>
</tr>
<tr>
<td>Stage1MaximumCoefficientCount</td>
</tr>
<tr>
<td>Stage1MaximumCoefficientSum</td>
</tr>
<tr>
<td>Stage1MinimumCoefficientCount</td>
</tr>
<tr>
<td>Stage2Coefficients</td>
</tr>
<tr>
<td>Stage2MaximumCoefficient</td>
</tr>
<tr>
<td>Stage2MaximumCoefficientCount</td>
</tr>
<tr>
<td>Stage2MaximumCoefficientSum</td>
</tr>
<tr>
<td>Stage2MinimumCoefficientCount</td>
</tr>
<tr>
<td>Stage3Coefficients</td>
</tr>
<tr>
<td>Stage3FilterType</td>
</tr>
<tr>
<td>Stage3FilterTypes</td>
</tr>
<tr>
<td>Stage3MaximumCoefficient</td>
</tr>
</tbody>
</table>
Stage3MaximumCoefficientCount ISignalProcessingModuleFour2 Returns the maximum number of Stage3coefficients.

Stage3MinimumCoefficient ISignalProcessingModuleFour2 Returns the minimum value of any single stage3coefficient.

Stage3MinimumCoefficientCount ISignalProcessingModuleFour2 Returns the minimum number of Stage3coefficients.

Stage3Parameter ISignalProcessingModuleFour Sets and returns the parameter value of the current filter type.

Stage3ParameterMaximum ISignalProcessingModuleFour Returns maximum parameter value for the current filter type.

Stage3ParameterMinimum ISignalProcessingModuleFour Returns minimum parameter value for the current filter type.

Stage3Parameters ISignalProcessingModuleFour Returns the names of parameters for the current filter type.

**ISignalProcessingModuleFour History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISignalProcessingModuleFour2</td>
<td>9.90</td>
</tr>
<tr>
<td>ISignalProcessingModuleFour</td>
<td>7.2</td>
</tr>
</tbody>
</table>
SMCType Object

Description

Contains the methods and properties to perform an Scalar Measurement Calibration for the Frequency Converter Application (option S93083A/B).

Accessing the SMCType object

See an example which creates and calibrates an SMC measurement.

You can also do the following:

```vba
Set app = CreateObject("AgilentPNA835x.Application")
Set CalMgr = app.GetCalManager
Set guidedCal = CalMgr.CreateCustomCalEx(1)
Set SMC = guidedCal.CustomCalConfiguration
SMC.ConnectorType(1) = "APC 3.5 male"
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcquireStep</td>
<td>ISMCType</td>
<td>Acquire the measurement data for the specified step in the calibration process.</td>
</tr>
<tr>
<td>GenerateErrorTerms</td>
<td>ISMCType</td>
<td>Generates the error terms for the calibration.</td>
</tr>
<tr>
<td>GenerateSteps</td>
<td>ISMCType</td>
<td>Returns the number of steps required to complete the calibration.</td>
</tr>
<tr>
<td>GetStepDescription</td>
<td>ISMCType</td>
<td>Returns the description of the specified step calibration process.</td>
</tr>
<tr>
<td>ImportDataSet</td>
<td>ISMCType4</td>
<td>Imports separate power meter data for SMC cal.</td>
</tr>
<tr>
<td>Properties</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Initialize</td>
<td>Begins a calibration.</td>
<td></td>
</tr>
<tr>
<td>AutoOrient</td>
<td>Sets ECAL module automatic orientation ON or OFF.</td>
<td></td>
</tr>
<tr>
<td>CalibrationPort</td>
<td>Sets or returns the calibration source port for the calibration.</td>
<td></td>
</tr>
<tr>
<td>CalKitType</td>
<td>Sets and returns a calibration kit type for calibration.</td>
<td></td>
</tr>
<tr>
<td>CompatibleCalKits</td>
<td>Returns a list of cal kits that are compatible with the connector type for the specified port.</td>
<td></td>
</tr>
<tr>
<td>ConnectorType</td>
<td>Sets or queries the connector type for the specified port.</td>
<td></td>
</tr>
<tr>
<td>Do2PortEcal</td>
<td><strong>Superseded</strong> - Replaced by CalKitType Specify ECAL or Mechanical calibration.</td>
<td></td>
</tr>
<tr>
<td>EcalCharacterization</td>
<td><strong>Superseded</strong> - Replaced by CalKitType Specifies the characterization data within an ECal module to be used for the calibration.</td>
<td></td>
</tr>
<tr>
<td>EcalOrientation</td>
<td>Specifies which port of the ECal module is connected to which port of the VNA when the AutoOrient property = False.</td>
<td></td>
</tr>
<tr>
<td>EnableLOPowerCal</td>
<td>Enable LO Power Cal</td>
<td></td>
</tr>
<tr>
<td>FixedDelay</td>
<td>Set and return the known delay through the calibration mixer.</td>
<td></td>
</tr>
<tr>
<td>MixerCharacterizationFile</td>
<td>Set the filename of the S2P file used to characterize the calibration mixer.</td>
<td></td>
</tr>
<tr>
<td>NetworkFilename</td>
<td>Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement.</td>
<td></td>
</tr>
<tr>
<td>NetworkMode</td>
<td>Embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement.</td>
<td></td>
</tr>
<tr>
<td>OmitIsolation</td>
<td><strong>Superseded</strong> - Replaced by SetIsolationPaths and GetIsolationPaths Sets and returns whether Isolation portion of the calibration will be performed or not.</td>
<td></td>
</tr>
<tr>
<td>SeparatePowerCal</td>
<td>Use a Thru standard or to use two power sensor connections during the SMC power cal</td>
<td></td>
</tr>
</tbody>
</table>
ThruCalMethod | ISMCType | **Superseded** - Replaced by PathThruMethod Property

Sets and returns the method for performing the thru portion of the calibration.

ValidConnectorTypes | ISMCType | Returns a list of connector types for which there are calibration kits.

**ISMCType History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISMCType</td>
<td>3.5</td>
</tr>
<tr>
<td>ISMCType2</td>
<td>6.0</td>
</tr>
<tr>
<td>ISMCType4</td>
<td>9.0</td>
</tr>
</tbody>
</table>
SourcePowerCalibrator Object

Description

This object is a child of the Application object and is a vehicle for performing source power calibrations.

Accessing the SourcePowerCalibrator Object

```vbs
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim ispc As ISourcePowerCalibrator
Set ispc = app.SourcePowerCalibrator
```

Note: If you see this error:

*The target NA stimulus channel has not been set. With the SourcePowerCalibrator object, you must first call SetCalInfoEx with the channel number. SetCalInfoEx must not be made against a temporary copy of the SourcePowerCalibrator object.*

Or this error:

*Channel not found.*

It is either because you set the wrong channel number in the SetCalInfoEx call or because the SourcePowerCalibrator object is unique in that you MUST use the SAME instance of the SourcePowerCalibrator object for every property that is set. Do this by naming/setting a variable as in this vbs example:

```vbs
set sourcepowercalibrator = app.sourcepowercalibrator
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

Note: Interface ISourcePowerCalibrator is abbreviated as ISPC in the following table.

(Bold Methods or Properties provide access to a child object)
### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AbortPowerAcquisition</td>
<td>ISPC</td>
<td>Aborts a source power cal acquisition sweep that is currently in progress.</td>
</tr>
<tr>
<td>AcquirePowerReadings</td>
<td>ISPC</td>
<td><strong>Superseded with</strong> AcquirePowerReadingsEx</td>
</tr>
<tr>
<td>AcquirePowerReadingsEx</td>
<td>ISPC4</td>
<td>Initiates a source power cal acquisition.</td>
</tr>
<tr>
<td>ApplyPowerCorrectionValues</td>
<td>ISPC</td>
<td><strong>Superseded with</strong> ApplyPowerCorrectionValuesEx</td>
</tr>
<tr>
<td>ApplyPowerCorrectionValuesEx</td>
<td>ISPC5</td>
<td>Applies correction values after completing a source power cal acquisition sweep. Optionally perform a calibration of the reference receiver used in the source power cal.</td>
</tr>
<tr>
<td>CheckPower</td>
<td>ISPC2</td>
<td>Measures power at a specific frequency. Used to test power level before and/or after applying a source power calibration.</td>
</tr>
<tr>
<td>LaunchPowerMeterSettingsDialog</td>
<td>ISPC2</td>
<td>Launches the Power Meter Settings dialog on the VNA.</td>
</tr>
<tr>
<td>SetCalInfo</td>
<td>ISPC</td>
<td><strong>Superseded with</strong> SetCalInfoEx Method</td>
</tr>
<tr>
<td>SetCalInfo2</td>
<td>ISPC3</td>
<td><strong>Superseded with</strong> SetCalInfoEx Method</td>
</tr>
<tr>
<td>SetCalInfoEx Method</td>
<td>ISPC4</td>
<td>Specifies the channel and source port to be used for the source power calibration.</td>
</tr>
<tr>
<td>SetPowerAcquisitionDevice</td>
<td>ISPC3</td>
<td>Sets the power sensor channel (A or B) to be used. This method is ONLY necessary when performing an SMC calibration.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalPower</td>
<td>ISPC</td>
<td>Specifies the power level that is expected at the desired reference plane.</td>
</tr>
<tr>
<td>IterationsTolerance</td>
<td>ISPC3</td>
<td>Sets the maximum desired deviation from the sum of the test port power and the offset value.</td>
</tr>
<tr>
<td>LastCalPassedTolerance</td>
<td>ISPC7</td>
<td>Returns pass / fail status of the user-specified tolerance limits on the target cal power.</td>
</tr>
<tr>
<td>MaximumIterationsPerPoint</td>
<td>ISPC3</td>
<td>Specifies maximum number of readings to take at each data point for iterating the source power.</td>
</tr>
</tbody>
</table>
**PowerAcquisitionDevice** ISPC2 Specifies the power sensor channel (A or B) that is currently selected for use at a specific frequency.

**PowerLossSegments (collection)** ISPC2 Collection for iterating through the segments of the power loss table used in source power calibration.

**PowerMeterGPIBAddress** ISPC Specifies the GPIB address of the power meter.

**PowerMeterInterfaces** ISPC6 Collection for getting a handle to the available power meters.

**PowerSensors (collection)** ISPC2 Collection for iterating through the PowerSensor objects which are connected to the power meter for a source power cal.

**ReadingsPerPoint** ISPC Specifies the maximum power readings for power meter settling.

**ReadingsTolerance** ISPC3 Power meter settling tolerance value.

**USBPowerMeterCatalog** ISPC6 Returns a list of USB power meters that are connected to the VNA.

**UsePowerLossSegments** ISPC Specifies if subsequent calls to the AcquirePowerReadings method will make use of the loss table (PowerLossSegments).

**UsePowerSensorFrequencyLimits** ISPC Specifies if subsequent calls to the AcquirePowerReadings method will make use of power sensor frequency checking capability.

---

**ISourcePowerCalibrator History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISourcePowerCalibrator</td>
<td>2.0</td>
</tr>
<tr>
<td>ISourcePowerCalibrator2</td>
<td>3.5</td>
</tr>
<tr>
<td>ISourcePowerCalibrator3</td>
<td>4.0</td>
</tr>
<tr>
<td>ISourcePowerCalibrator4</td>
<td>6.2</td>
</tr>
<tr>
<td>ISourcePowerCalibrator5</td>
<td>7.2</td>
</tr>
<tr>
<td>ISourcePowerCalibrator6</td>
<td>7.5</td>
</tr>
<tr>
<td>ISourcePowerCalibrator7</td>
<td>9.2</td>
</tr>
</tbody>
</table>
SpectrumAnalyzer Object

Description

Controls the Spectrum Analyzer Application settings.

Accessing the SpectrumAnalyzer object

```
Dim app as AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")
app.CreateCustomMeasurementEx (1, "SpectrumAnalyzer", "A", 1)
Dim SA
Set SA = app.ActiveChannel.CustomChannelConfiguration
```

SA Application - SA Setup tab

- ResBWList
- ResolutionBW
- ResolutionBWMode
- ResolutionBWMin
- ResolutionBWMax
- VideoBW
- VideoBWMode
- VideoBWMin
- VideoBWMax
- DetectorFunction
- EnableDetectorBypass
- VideoAveragingType
- VideoAveragingCount
Source Setup tab

- SourceSweepType
- SourceSweepType2
- SourceStartFrequency
- SourceStopFrequency
- SourceCWFrequency
- SourcePowerPointCount
- SourcePowerRepeatCount
- SourceStartPower
- SourceStopPower
- SourcePower
- SourceSweepFirstDimension
- SourcePointCount
- SourceRepeatCount

Coherence Setup tab

- Coherence (Provides access to SpectrumAnalyzerCoherence Object)
- MultiToneImageRejectEnable (Part of SpectrumAnalyzerCoherence Object)
- MultiToneImageRejectDataDisplay (Part of SpectrumAnalyzerCoherence Object)
- MultiToneImageRejectSpacing (Part of SpectrumAnalyzerCoherence Object)
- MultiToneImageRejectPeriod (Part of SpectrumAnalyzerCoherence Object)
- MultiToneImageRejectReference (Part of SpectrumAnalyzerCoherence Object)
- MultiToneImageRejectHarmonic (Part of SpectrumAnalyzerCoherence Object)
- VectorAverageEnable (Part of SpectrumAnalyzerCoherence Object)
- VectorAverageMax (Part of SpectrumAnalyzerCoherence Object)
- VectorAverageValue (Part of SpectrumAnalyzerCoherence Object)
Advanced Setup tab

BandwidthShape
ImageRejectMethod
ImageRejectStrength
ResolutionBWVideoBWRatio
SpanResolutionBWRatio
SearchOccupiedBWMinFreq
DCSourceSweepState
DCSourcePointCount
DCSourceSweepFirstDimension
FrequencyAutoTune

IF Setup tab
DFT (Provides access to SpectrumAnalyzerDFT Object)

ADCFilter

EnableADCFilterAuto

AutoBandwidth (Part of SpectrumAnalyzerDFT Object)

BandwidthNarrowMin (Part of SpectrumAnalyzerDFT Object)

BandwidthNarrowMax (Part of SpectrumAnalyzerDFT Object)

BandwidthWideMin (Part of SpectrumAnalyzerDFT Object)

BandwidthWideMax (Part of SpectrumAnalyzerDFT Object)

Trigger Setup tab

TriggerADCLevelState

TriggerADCLevelValue

TriggerPeriodicCounterState

TriggerPeriodicCounterValue
Processing Setup tab

- Type (Part of SpectrumAnalyzerDFT Object)
  - EnableImageRejectTraces
  - AcquisitionTime
  - LOCount
  - SpanBinsCount
  - Resolution (Part of SpectrumAnalyzerDFT Object)
  - RecordSize (Part of SpectrumAnalyzerDFT Object)
  - ForceADCRecordSize

ADC & LO Setup tab

- ADCSampleRate
  - EnableADCSampleRateAuto
  - ADCEnableFIRFor25Mhz
  - EnableADCdither
  - ForceADCRecordSize
  - ADCRecordSizeMax
  - ADCRecordSizeMin
  - ADCRecordSize
  - EnableForceADCRecordSize
  - ADCStacking
  - ADCStackingMax
  - ADCStackingState
  - ADCMultRecSize
  - ADCMultRecPeriod
  - ADCMultRecState
  - EnableRandomizedLO
  - EnableForceLOToFrequency
  - ForceLOToFrequency
Data Setup tab

<table>
<thead>
<tr>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataFormat</td>
</tr>
<tr>
<td>DataFirstRFBin</td>
</tr>
<tr>
<td>ExportReceiverList</td>
</tr>
<tr>
<td>ExportReceiverSetList</td>
</tr>
<tr>
<td>ExportReceiverCount</td>
</tr>
<tr>
<td>DataBinCount</td>
</tr>
<tr>
<td>DataByteSize</td>
</tr>
<tr>
<td>DataBytesPerBin</td>
</tr>
<tr>
<td>DataByteSizeLSB</td>
</tr>
<tr>
<td>DataByteSizeMSB</td>
</tr>
<tr>
<td>BinaryFileEnabled</td>
</tr>
<tr>
<td>DataExportMarkersEnabled (Part of SpectrumAnalyzerDFT Object)</td>
</tr>
<tr>
<td>DataExportWindowingFactor (Part of SpectrumAnalyzerDFT Object)</td>
</tr>
<tr>
<td>TextFileEnabled</td>
</tr>
<tr>
<td>FileVerboseEnabled</td>
</tr>
<tr>
<td>FileEraseEachSweep</td>
</tr>
<tr>
<td>FilePrefix</td>
</tr>
<tr>
<td>FIFOEnabled</td>
</tr>
<tr>
<td>MemShareEnabled</td>
</tr>
<tr>
<td>MemShareName</td>
</tr>
<tr>
<td>DataLevelThreshold (Part of SpectrumAnalyzerDFT Object)</td>
</tr>
<tr>
<td>DataLevelThresholdEnabled (Part of SpectrumAnalyzerDFT Object)</td>
</tr>
</tbody>
</table>

See Also:

- **Example Program**: Create a Spectrum Analyzer measurement
- About the Spectrum Analyzer Application
- VNA Automation Interfaces
- The VNA Object Model

Other SA commands
SA Marker Settings
Sets and reads the bandwidth of the band density marker.
Returns the band density level in dBm/Hz from the band density marker.
Marker to SA
Read Band Power
Read/Set Band Power Span
Read/Set Band Power State
Sets and reads the state of the band density noise marker.
Sets and reads the bandwidth of the band power density marker.
Sets and reads the state of the band power density marker.
Sets and reads the bandwidth of the band tone density marker.
Sets and reads the state of the band tone density marker.
Sets and reads the spacing of the band tone density marker.
Sets and reads the frequency span used by Power Density to normalize the power.
Read occupied bandwidth center frequency
Set and read occupied bandwidth percentage of span

BandDensityBW
BandDensityValue
toSA
BandpowerData
BandpowerSpan
BandpowerState
BandDensityNoiseState
BandDensityPowerBW
BandDensityPowerState
BandDensityToneBW
BandDensityToneState
BandDensityToneSpacing
BandDensityEQSPan
OccupiedBandCenter
OccupiedBandPercent
Read the occupied bandwidth power.
Read occupied bandwidth span
Set occupied bandwidth state

OccupiedBandPowerdBm
OccupiedBandSpan
OccupiedBandState
SpectrumAnalyzerCoherence

Description

Controls the Spectrum Analyzer Application Coherence and modulation distortion settings.

Accessing the SpectrumAnalyzerCoherence object

```cpp
CComPtr<ISpectrumAnalyzer4> SA;
CComPtr<ICoherenceSA> sacoherence;
SA->GetCoherence(&sacoherence);
```

Method

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MultiToneDataDisplay</td>
<td>ICoherenceSA</td>
<td>Sets and returns the data display mode.</td>
</tr>
<tr>
<td>MultiToneEnable</td>
<td>ICoherenceSA</td>
<td>Enables/disables multitone image rejection.</td>
</tr>
<tr>
<td>MultiToneHarmonicRejection</td>
<td>ICoherenceSA</td>
<td>Sets and returns the multitone harmonic rejection.</td>
</tr>
<tr>
<td>MultiToneNyquistProtection</td>
<td>ICoherenceSA2</td>
<td>Sets and returns the Nyquist protection level.</td>
</tr>
<tr>
<td>MultiTonePeriod</td>
<td>ICoherenceSA</td>
<td>Sets and returns the multitone image rejection period.</td>
</tr>
<tr>
<td>MultiToneReference</td>
<td>ICoherenceSA</td>
<td>Sets and returns the multitone image rejection offset frequency.</td>
</tr>
<tr>
<td>MultiToneSettingsValid</td>
<td>ICoherenceSA2</td>
<td>Returns the current multitone settings and determine if they are valid or not.</td>
</tr>
<tr>
<td>MultiToneSpacing</td>
<td>ICoherenceSA</td>
<td>Sets and returns the tone spacing of the multitone signal.</td>
</tr>
<tr>
<td>PhaseDisplayMinLevel</td>
<td>ICoherenceSA2</td>
<td>Sets and returns the phase display minimum level.</td>
</tr>
<tr>
<td>PhaseProcessState</td>
<td>ICoherenceSA2</td>
<td>Enables/disables phase computing.</td>
</tr>
<tr>
<td>VectorAverageEnable</td>
<td>ICoherenceSA</td>
<td>Sets and returns the ON/OFF state of the vector averaging.</td>
</tr>
</tbody>
</table>
VectorAverageValue  ICoherenceSA  Sets and returns the vector averaging value.
VectorAverageValueMax  ICoherenceSA  Returns the current maximum available vector averaging value.

SpectrumAnalyzerCoherence History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICoherenceSA</td>
<td>12.80</td>
</tr>
<tr>
<td>ICoherenceSA2</td>
<td>12.90</td>
</tr>
</tbody>
</table>
SpectrumAnalyzerDFT Object

Description

Controls the Spectrum Analyzer Application Discrete Fourier Transform (DFT) settings.

Accessing the SpectrumAnalyzer object

```cpp
CComPtr<ISpectrumAnalyzer4> SA;
CComPtr<ISpectrumAnalyzerDFT> sadft;
SA->GetDFT(&sadft);
```

Method | Interface | Description
---|---|---
See History

None

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoBandwidth</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Sets default values for DFT bandwidth.</td>
</tr>
<tr>
<td>BandwidthNarrowMax</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Sets maximum value for narrow DFT bandwidth.</td>
</tr>
<tr>
<td>BandwidthNarrowMin</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Sets minimum value for narrow DFT bandwidth.</td>
</tr>
<tr>
<td>BandwidthWideMax</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Sets maximum value for wide DFT bandwidth.</td>
</tr>
<tr>
<td>BandwidthWideMin</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Sets minimum value for wide DFT bandwidth.</td>
</tr>
<tr>
<td>BinaryFileEnabled</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Enables binary file output.</td>
</tr>
<tr>
<td>DataBinCount</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Returns the number of DFT points processed across the total RF span.</td>
</tr>
<tr>
<td>DataByteSize</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Returns the byte size.</td>
</tr>
<tr>
<td>DataByteSizeLOW</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Returns the low part of the byte.</td>
</tr>
<tr>
<td>DataByteSizeHIGH</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Returns the high part of the byte.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DataBytesPerBin</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Returns the number of bytes per bin.</td>
</tr>
<tr>
<td>DataExportMarkersEnabled</td>
<td>ISpectrumAnalyzerDFT2</td>
<td>Enables/disables adding marker data to the text file (*.txt) output.</td>
</tr>
<tr>
<td>DataExportWindowingFactor</td>
<td>ISpectrumAnalyzerDFT2</td>
<td>Returns the windowing factor for band power computation.</td>
</tr>
<tr>
<td>DataFirstRFBin</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Returns the frequency of the first RF bin.</td>
</tr>
<tr>
<td>DataFormat</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Sets and returns the data format.</td>
</tr>
<tr>
<td>DataLevelThreshold</td>
<td>ISpectrumAnalyzerDFT2</td>
<td>Sets and returns the threshold value (dBm).</td>
</tr>
<tr>
<td>DataLevelThresholdEnabled</td>
<td>ISpectrumAnalyzerDFT2</td>
<td>Enables/disables data level threshold mode.</td>
</tr>
<tr>
<td>ExportReceiverCount</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Returns the number of exported receivers.</td>
</tr>
<tr>
<td>ExportReceiverList</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Returns the currently exported receiver list.</td>
</tr>
<tr>
<td>ExportReceiverSetList</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Sets and returns the list of exported receivers.</td>
</tr>
<tr>
<td>FIFOEnabled</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Enables exporting data to the FIFO (First-IN, First-OUT) data buffer.</td>
</tr>
<tr>
<td>FileEraseEachSweep</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Enables output data files to be erased after each sweep.</td>
</tr>
<tr>
<td>FilePrefix</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Sets and returns the file name prefix and path of the data file.</td>
</tr>
<tr>
<td>FileVerboseEnabled</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Enables output of data and frequency.</td>
</tr>
<tr>
<td>MemShareEnabled</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Enables data to be output to shared memory.</td>
</tr>
<tr>
<td>MemShareName</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Assigns a specified name to the shared data.</td>
</tr>
<tr>
<td>RecordSize</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Returns the current DFT record size.</td>
</tr>
<tr>
<td>Resolution</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Returns the DFT resolution.</td>
</tr>
<tr>
<td>TextFileEnabled</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Enables text file output.</td>
</tr>
<tr>
<td>Type</td>
<td>ISpectrumAnalyzerDFT</td>
<td>Sets the DFT record size type.</td>
</tr>
</tbody>
</table>

---

**SpectrumAnalyzerDFT History**
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISpectrumAnalyzerDFT</td>
<td>12.70 and 12.80</td>
</tr>
<tr>
<td>ISpectrumAnalyzerDFT2</td>
<td>13.25</td>
</tr>
</tbody>
</table>
SweptIMD Object

Description

Controls the Swept IMD Application settings.

See Properties to set for each sweep type.

Accessing the SweptIMD object

```vba
Dim app as AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")
app.CreateCustomMeasurementEx(1, "SweptIMD", "PwrMain", 1)
Dim IMD
Set IMD = app.ActiveChannel.CustomChannelConfiguration
```

See Also:

- **Example Program** Create and Cal a Swept IMD Measurement
- Use std channel commands to set source and receiver attenuation.
- SweptIMDCal Object
- About the Swept IMD Application
- VNA Automation Interfaces
- The VNA Object Model

### Method

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetPortMap</td>
<td>ISweptIMD</td>
<td>Sets port mapping</td>
</tr>
</tbody>
</table>

### Property

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CompositeNormalizationMode</td>
<td>ISweptIMD</td>
<td>Sets and returns the method by which CTB and CSO calculations are performed.</td>
</tr>
<tr>
<td>CompositeNormalizedCSOPower</td>
<td>ISweptIMD</td>
<td>Sets and returns the CSO Power.</td>
</tr>
<tr>
<td>Property</td>
<td>ISweptIMD/ISweptIMD2/ISweptIMD4</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>---------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CompositeNormalizedCTBPower</td>
<td>ISweptIMD</td>
<td>Sets and returns the CTB Power.</td>
</tr>
<tr>
<td>CoupleTonePower</td>
<td>ISweptIMD</td>
<td>ON</td>
</tr>
<tr>
<td>CSONumDistortionProducts</td>
<td>ISweptIMD</td>
<td>Sets the “N = number of distortion products” value for the calculation of the CSO parameter.</td>
</tr>
<tr>
<td>CSOOffset Property</td>
<td>ISweptIMD</td>
<td>Sets and returns the offset that is applied to CSO measurements.</td>
</tr>
<tr>
<td>CTBOffset Property</td>
<td>ISweptIMD</td>
<td>Sets and returns the offset that is applied to CTB measurements.</td>
</tr>
<tr>
<td>CTBXMODNumCarriers</td>
<td>ISweptIMD</td>
<td>Sets the “N = Total number of carriers” value used in the calculation of the XMOD and CTB parameter.</td>
</tr>
<tr>
<td>DeltaFrequency</td>
<td>ISweptIMD</td>
<td>Fixed tone spacing value.</td>
</tr>
<tr>
<td>DeltaFrequencyStart</td>
<td>ISweptIMD</td>
<td>Start spacing of the main tones.</td>
</tr>
<tr>
<td>DeltaFrequencyStop</td>
<td>ISweptIMD</td>
<td>Stop spacing of the main tones.</td>
</tr>
<tr>
<td>DeviceInputPort</td>
<td>ISweptIMD</td>
<td>Reads input port map</td>
</tr>
<tr>
<td>DeviceOutputPort</td>
<td>ISweptIMD</td>
<td>Reads output port map</td>
</tr>
<tr>
<td>EqualTonePower</td>
<td>ISweptIMD2</td>
<td>Superseded with LevelingMethod Set Equal Tone Power state.</td>
</tr>
<tr>
<td>F1Frequency</td>
<td>ISweptIMD</td>
<td>Frequency of the F1 tone.</td>
</tr>
<tr>
<td>F2Frequency</td>
<td>ISweptIMD</td>
<td>Frequency of the F1 tone.</td>
</tr>
<tr>
<td>FrequencyCenter</td>
<td>ISweptIMD</td>
<td>Center frequency of the main tones.</td>
</tr>
<tr>
<td>FrequencyCenterCenter</td>
<td>ISweptIMD</td>
<td>Sweep center frequency when sweeping the main tones.</td>
</tr>
<tr>
<td>FrequencyCenterSpan</td>
<td>ISweptIMD</td>
<td>Frequency span when sweeping the main tones.</td>
</tr>
<tr>
<td>FrequencyCenterStart</td>
<td>ISweptIMD</td>
<td>Start frequency when sweeping the main tones.</td>
</tr>
<tr>
<td>FrequencyCenterStop</td>
<td>ISweptIMD</td>
<td>Stop frequency when sweeping the main tones.</td>
</tr>
<tr>
<td>Has2ndOrderTrace</td>
<td>ISweptIMD4</td>
<td>Reads the state of whether or not 2nd order parameters are measured.</td>
</tr>
<tr>
<td>HighestOrderProduct</td>
<td>ISweptIMD</td>
<td>Reads the highest product that can be measured by SweptIMD.</td>
</tr>
<tr>
<td>HighestOrderProductInUse</td>
<td>ISweptIMD4</td>
<td>Reads the highest product measured by SweptIMD.</td>
</tr>
</tbody>
</table>
**IMToneIFBandwidth**  
ISweptIMD  
IF Bandwidth for measurement of the intermodulation products.

**LevelingMethod**  
ISweptIMD3  
Set power leveling mode

**MainToneIFBandwidth**  
ISweptIMD  
IF Bandwidth for measurement of the Main tones.

**SweepType**  
ISweptIMD  
Type of sweep for a Swept IMD measurement.

**TonePower**  
ISweptIMD  
Power level of the Main Tones.

**TonePowerSetAt**  
ISweptIMD2  
*Superseded* with LevelingMethod  
Set power level at DUT Input or Output

**TonePowerStart**  
ISweptIMD  
Start power level of the Main tones.

**TonePowerStop**  
ISweptIMD  
Stop power level of the Main tones.

**See Also**

- RoleDevice Property to use an external source for f2.

The following commands are relevant for each IMD sweep type:

**0 - naIMDToneCW Sweep**

- F1Frequency Property
- F2Frequency Property
- FrequencyCenter Property
- DeltaFrequency Property
- TonePower (1,2) Property

**1 - naIMDTonePowerSweep**

- F1Frequency Property
- F2Frequency Property
- FrequencyCenter Property
- DeltaFrequency Property
- TonePowerStart Property
- TonePowerStop Property
2 - naIMDToneCenterFreqSweep

- FrequencyCenterStart Property
- FrequencyCenterStop Property
- FrequencyCenterCenter Property
- FrequencyCenterSpan Property
- DeltaFrequency Property
- TonePower (1,2) Property

3 - naIMDDeltaFrequencySweep

- DeltaFrequencyStart Property
- DeltaFrequencyStop Property
- FrequencyCenter Property
- TonePower (1,2) Property

4 - naIMDToneSegmentSweep  (Not available for IMDx)

- DeltaFrequency Property
- TonePower (1,2) Property
- Also use standard segment sweep commands

5 - naLOPowerSweep  (IMDx ONLY)

- converter.LOStartPower Property
- converter.LOStopPower Property

ISweptIMD History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISweptIMD</td>
<td>8.33</td>
</tr>
<tr>
<td>ISweptIMD2</td>
<td>9.40</td>
</tr>
<tr>
<td>ISweptIMD3</td>
<td>9.80</td>
</tr>
<tr>
<td>ISweptIMD4</td>
<td>12.80</td>
</tr>
</tbody>
</table>
SweptIMDCal Object

Description

Sets properties that are unique to a Swept IMD Cal (opt S93087A/B).

Use the Guided Calibration commands for the remaining commands to perform a Swept IMD Cal

Accessing the SweptIMDCal object

```vba
Dim app as AgilentPNA835x.Application
Set IMDcal = pna.GetCalmanager.CreateCustomCalEx(channelNum)
Set IMDCalExtension = IMDcal.CustomCalConfiguration
IMDCalExtension.PowerLevel = 5
```

See Also:

- Example Program Create and Cal a Swept IMD Measurement
- SweptIMD Object
- About Swept IMD measurements
- The VNA Object Model
- VNA Automation Interfaces

### Method

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>See History</td>
</tr>
</tbody>
</table>

None

### Property

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalibrationFrequencies</td>
<td>ISweptIMDCal</td>
<td>Perform the source power cal at the center frequencies or at all main tone frequencies.</td>
</tr>
<tr>
<td>CalMethod</td>
<td>ISweptIMDCal</td>
<td>Sets the method by which the match-correction portion of an IMD cal is performed</td>
</tr>
<tr>
<td>EnableLOPowerCal</td>
<td>ISweptIMDCal2</td>
<td>Enable or disable an LO power cal with an IMDx calibration.</td>
</tr>
</tbody>
</table>
Include 2nd Order Product

Include the second order products in the calibration.

Max Product

Sets and returns the maximum intermod product frequencies to be calibrated.

Power Level

Set and read the power level of the source power cal.

Power Sensor Cal Kit Type

Set and read the cal kit to be used for port 1 adapter compensation.

Power Sensor Connector Type

*Superseded* Use GuidedCal.PowerSensorConnectorType

ISweptIMD History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISweptIMDCal</td>
<td>8.33</td>
</tr>
<tr>
<td>ISweptIMDCal2</td>
<td>8.55</td>
</tr>
</tbody>
</table>
TestsetControl Object

Description

A TestsetControl object is used to control one of the supported test sets. Only one external test set can be controlled by the VNA at any time. The Testset Control object appears as an item in the ExternalTestsets collection, which in turn is a property of the main application object.

If the specified test set is not connected to the VNA or is not ON, then setting Enabled = True will return an error. All other properties can be set even if the test set is not connected.

Note: The ONLY way to load a test set configuration file is by sending the testsets.Add method. There is no method to query the test set type. See an example program.

Accessing a TestsetControl object

The ExternalTestsets collection is a property of the main Application Object. You can obtain a handle to a testset object by specifying an item in the collection.

Visual Basic Example

```vba
Dim pna
Dim testsets As ExternalTestsets
Dim tset1 As TestsetControl
Set pna = CreateObject("AgilentPNA835x.Application")
Set testsets = pna.ExternalTestsets
Set tset1 = testsets(1)
' make COM calls on tset1 object
End Sub
```

See Also:

- E5091A Testset Object
- About External Testset Control
- External Testset Control Example
- ExternalTestsets Collection
- The VNA Object Model
## Methods

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(See history)</td>
</tr>
</tbody>
</table>

None

## Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ControlLines</td>
<td>ITestsetControl</td>
<td>Sets the control lines of the specified Test set.</td>
</tr>
<tr>
<td>Enabled</td>
<td>ITestsetControl</td>
<td>Enables and disables (ON/OFF) the port mapping and control line output of the specified test set.</td>
</tr>
<tr>
<td>ID</td>
<td>ITestsetControl</td>
<td>Returns the test set ID number.</td>
</tr>
<tr>
<td>Label</td>
<td>ITestsetControl</td>
<td>Returns the label on a given channel for the specified test set.</td>
</tr>
<tr>
<td>NumberOfPorts</td>
<td>ITestsetControl</td>
<td>Reads the number of ports that are on the specified test set.</td>
</tr>
<tr>
<td>OutputPorts</td>
<td>ITestsetControl</td>
<td>Sets or returns the port mappings for ALL ports.</td>
</tr>
<tr>
<td>PortCatalog</td>
<td>ITestsetControl</td>
<td>Returns the selections available for a given logical port.</td>
</tr>
<tr>
<td>SelectPort</td>
<td>ITestsetControl</td>
<td>Sets and returns the logical port value.</td>
</tr>
<tr>
<td>ShowProperties</td>
<td>ITestsetControl</td>
<td>Turns status bar display of test set properties on or off.</td>
</tr>
<tr>
<td>Type</td>
<td>ITestsetControl</td>
<td>Returns the test set model.</td>
</tr>
</tbody>
</table>

## ExternalTestset History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITestsetControl</td>
<td>6.0</td>
</tr>
</tbody>
</table>
Trace Object

Description

The Trace object controls how the measurement data is displayed. You can control scale, reference position, and value from the Trace Object.

Accessing a Trace object

There are several ways to get a handle to a trace.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim trace As Trace

Then you can do any of the following:

Set trace = app.NAWindows(1).traces(1)
set trace = app.NAWindows.item(1).ActiveTrace
set trace = app.ActiveNAWindow.traces.item(1)
set trace = app.ActiveNAWindow.ActiveTrace
Set trace = app.Measurements(1).trace
Set trace = app.ActiveMeasurement.trace
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Traces, Channels, and Windows on the VNA
- Example Programs
<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autoscale</td>
<td>ITrace</td>
<td>Autoscales the trace or all of the traces in the selected window.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the NAWindow Object</td>
</tr>
<tr>
<td>Move</td>
<td>ITrace3</td>
<td>Moves a trace from one window to another.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measurement</td>
<td>Returns a measurement handle from trace object.</td>
</tr>
<tr>
<td>Name</td>
<td>Sets or returns the trace name.</td>
</tr>
<tr>
<td>ReferencePosition</td>
<td>Sets or returns the Reference Position of the active trace.</td>
</tr>
<tr>
<td>ReferenceValue</td>
<td>Sets or returns the value of the Y-axis Reference Level of the active trace.</td>
</tr>
<tr>
<td>YScale</td>
<td>Sets or returns the Y-axis Per-Division value of the active trace.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITrace</td>
<td>1.0</td>
</tr>
<tr>
<td>ITrace2</td>
<td>9.40</td>
</tr>
<tr>
<td>ITrace3</td>
<td>9.50</td>
</tr>
</tbody>
</table>
**Traces Collection**

**Description**

Child of the **Application** Object. A collection that provides a mechanism for getting a handle to a trace or iterating through the traces in a window.

**Accessing the Traces collection**

Get a handle to the traces collection through the NaWindows collection. The following example sets the variable `trcs` to the collection of traces in window 1 of the NaWindows collection.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim trcs As traces
Set trcs = app.NAWindows(1).traces
```

**See Also:**

- **Trace Object**
- **Collections in the Analyzer**
- **The VNA Object Model**
- **Example Programs**

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>Use to get a handle to a trace</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of traces in the collection.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the current Application.</td>
</tr>
</tbody>
</table>
Transform Object

Description
Contains the methods and properties that control Time Domain transforms.

Accessing the Transform Object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim trans As Transform
Set trans = app.ActiveMeasurement.Transform
```

See Also:
- VNA Automation Interfaces
- The VNA Object Model
- Time Domain Topics
- Example Programs

**Note:** Sweep Type must be set to Linear before setting Time Domain Transform (state) ON.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetFrequencyLowPass</td>
<td>ITransform</td>
<td>Sets low frequencies for low pass.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alignment</td>
<td>ITransform3 Sets or returns the alignment of the time domain measurement.</td>
</tr>
<tr>
<td>Center</td>
<td>ITransform Sets or returns the Center time.</td>
</tr>
<tr>
<td></td>
<td>Shared with the Gating Object</td>
</tr>
<tr>
<td>CoupledParameters</td>
<td>ITransform2 Select Transform parameters to couple</td>
</tr>
<tr>
<td>DistanceMarkerMode</td>
<td>ITransform2 Sets the measurement type in order to determine the correct</td>
</tr>
<tr>
<td></td>
<td>marker distance.</td>
</tr>
<tr>
<td>DistanceMarkerUnit</td>
<td>ITransform2 Sets the unit of measure for the display of marker distance</td>
</tr>
<tr>
<td></td>
<td>values.</td>
</tr>
</tbody>
</table>
**ImpulseWidth**  
*ITransform*  
Sets or returns the Impulse Width of Time Domain transform windows.

**KaiserBeta**  
*ITransform*  
Sets or returns the Kaiser Beta of Time Domain transform windows.

**Mode**  
*ITransform*  
Sets the type of transform.

**Span**  
*ITransform*  
Sets or returns the Span time.  
Shared with the Gating Object

**Start**  
*ITransform*  
Sets or returns the Start time.  
Shared with the Gating Object

**State**  
*ITransform*  
Turns an Object ON and OFF.

**StepRiseTime**  
*ITransform*  
Sets or returns the Rise time of the stimulus in Low Pass Step Mode.

**Stop**  
*ITransform*  
Sets or returns the Stop time.  
Shared with the Gating Object

**ITransform History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITransform</td>
<td>1.0</td>
</tr>
<tr>
<td>ITransform2</td>
<td>4.2</td>
</tr>
<tr>
<td>ITransform3</td>
<td>12.70</td>
</tr>
</tbody>
</table>
### TriggerSetup Object

**Description**

These properties setup Global triggering that effects the entire VNA application.

**Accessing the TriggerSetup object**

```plaintext
Dim app as AgilentPNA835x.Application
Dim trigSetup as ITriggerSetup
Set trigSetup = app.TriggerSetup
```

**See Also:**

- VNA Automation Interfaces
- The VNA Object Model
- Triggering in the VNA
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcceptTriggerBefore Armed</td>
<td>ITriggerSetup2 Allows a trigger signal to be remembered and then used when the VNA becomes armed (ready to be triggered).</td>
</tr>
<tr>
<td>ExternalTriggerConnectionBehavior</td>
<td>ITriggerSetup Configures the external triggering signal for the VNA</td>
</tr>
<tr>
<td>ReadyForTriggerPolarity</td>
<td>ITriggerSetup3 Specifies the polarity of Ready for Trigger output.</td>
</tr>
<tr>
<td>ReadyForTriggerStatus</td>
<td>ITriggerSetup4 Checks if the PNA is ready for a specific or any hardware trigger</td>
</tr>
</tbody>
</table>
Scope

ITriggerSetup

Determines whether a trigger signal affects a single channel or all channels in the VNA.

Source

ITriggerSetup

Sets or returns the source of triggering in the VNA.

TriggerOutputEnabled

ITriggerSetup2

Superseded

Enables the VNA to send trigger signals out the rear-panel TRIGGER OUT connector.

ITriggerSetup History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITriggerSetup</td>
<td>4.0</td>
</tr>
<tr>
<td>ITriggerSetup2</td>
<td>4.2</td>
</tr>
<tr>
<td>ITriggerSetup3</td>
<td>7.50.2 and 8.2</td>
</tr>
<tr>
<td>ITriggerSetup4</td>
<td>12.80</td>
</tr>
</tbody>
</table>
Uncertainty Object

Description

Provides access to the properties and methods that are used to display uncertainties for the measurement (Option S93015A/B).

Limitations

- Calibrations can be performed for ONLY ONE channel at a time.
- Putting Error Term data into Uncertainty Cal Sets using remote commands is NOT supported.

Accessing the Uncertainty object

Get a handle to the Uncertainty object.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim oUncert as Uncertainty
Set oUncert = app.ActiveMeasurement.Uncertainty
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
- Superseded commands

(Bold Methods or Properties provide access to a child object)
<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WriteUncertaintyFile</td>
<td>IUncertainty</td>
<td>Saves uncertainty data for the specified ports in three different formats.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CableRepeatabilityUncertainty</td>
<td>IUncertainty</td>
<td>Sets and returns whether to include the cable/connection repeatability contribution for the measurement trace.</td>
</tr>
<tr>
<td>CoverageFactor</td>
<td>IUncertainty</td>
<td>Sets and returns the coverage factor value to apply to the displayed uncertainty for the measurement trace.</td>
</tr>
<tr>
<td>DisplayType</td>
<td>IUncertainty</td>
<td>Sets and returns the display type for uncertainties for the measurement trace</td>
</tr>
<tr>
<td>ErrorTermUncertainty</td>
<td>IUncertainty</td>
<td>Sets and returns whether to include the uncertainties associated with the correction error terms in the uncertainty values for the measurement trace.</td>
</tr>
<tr>
<td>MeasurementNoiseUncertainty</td>
<td>IUncertainty</td>
<td>Sets and returns whether the noise contribution is currently included in the uncertainty values for the measurement trace.</td>
</tr>
</tbody>
</table>

**IUncertainty History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IUncertainty</td>
<td>10.40</td>
</tr>
</tbody>
</table>
**UncertaintyManager Object**

**Description**

Provides access to the properties and methods that are used to characterize Dynamic Uncertainties (Option S93015A/B).

**Accessing the UncertaintyManager object**

Get a handle to a the UncertaintyManager through the Application Object.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim uMgr as UncertaintyManager
Set uMgr = app.UncertaintyManager
```

**See Also:**

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
- **Superseded** commands

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recall</td>
<td>UncertaintyManager</td>
<td>Loads an uncertainty ‘workspace’ (*.ml4) file.</td>
</tr>
<tr>
<td>Save</td>
<td>UncertaintyManager</td>
<td>Saves an uncertainty ‘workspace’ (*.ml4) file.</td>
</tr>
</tbody>
</table>

**Properties**

- **CableRepeatabilityEnabled**
  - UncertaintyManager
  - Allows cable repeatability data to contribute to the uncertainty of a calibration.
Cables
UncertaintyManager Provides access to the collection of characterized cables used in Dynamic Uncertainty calibrations.

Characterizer
UncertaintyManager Provides access to the Methods and Properties used to perform Uncertainty characterizations.

MaximumUncertaintyPoints
UncertaintyManager Sets and returns maximum number of points for which uncertainties are to be computed.

PortNoiseEnabled
UncertaintyManager Allows noise data to contribute to the uncertainty of a calibration.

Ports
UncertaintyManager Provides access to the collection of VNA ports used in Dynamic Uncertainty calibrations.

StandardDefinitionsEnabled
UncertaintyManager Allows cal standard data to contribute to the uncertainty of a calibration.

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IUncertaintyManager</td>
<td>10.40</td>
</tr>
</tbody>
</table>
VISAPassThrough Object

Description

VISA (Virtual Instrument Software Architecture) is used to communicate with most instrumentation buses including the following:

- GPIB
- USB
- Serial
- Ethernet

Accessing the VISAPassThrough object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim vPass As VISAPassThrough
Set vPass = app.VISAPassThrough

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Close</td>
<td>IVISAPassthrough  Closes the remote VISA session</td>
</tr>
<tr>
<td>Find</td>
<td>IVISAPassthrough  Returns a comma separated list of either visa address strings or aliases</td>
</tr>
<tr>
<td>GetVISATimeout</td>
<td>IVISAPassthrough  Returns the timeout value (in milliseconds) for subsequent VISA pass-through commands</td>
</tr>
<tr>
<td>Open</td>
<td>IVISAPassthrough  Initiates a VISA pass-through session</td>
</tr>
<tr>
<td>ReadBinary</td>
<td>IVISAPassthrough  Reads data from the VISA pass-through device as a Safe Array of variants.</td>
</tr>
<tr>
<td>ReadBinaryCompact</td>
<td>IVISAPassthrough  Reads binary data in a more compact form of Safe Array.</td>
</tr>
<tr>
<td>ReadString</td>
<td>IVISAPassthrough  Reads string data from the VISA pass-through device.</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Reset</td>
<td>IVISAPassthrough Closes all currently open remote VISA sessions</td>
</tr>
<tr>
<td>SetVISATimeout</td>
<td>IVISAPassthrough Sends the timeout value (in milliseconds) for subsequent VISA pass-through commands</td>
</tr>
<tr>
<td>WriteBinary</td>
<td>IVISAPassthrough Write data to the VISA pass-through device - without header.</td>
</tr>
<tr>
<td>WriteString</td>
<td>IVISAPassthrough Write string data to the VISA pass-through device.</td>
</tr>
</tbody>
</table>
VMC Type Object

Description

Contains the methods and properties to perform a Vector Measurement Calibration for the Frequency Converter Application (option S93083A/B).

Accessing the VMCType object

See an example which creates and calibrates a VMC measurement.

You can also do the following:

```vba
Set app = CreateObject("AgilentPNA835x.Application")
Set CalMgr = app.GetCalManager
Set guidedCal = CalMgr.CreateCustomCalEx(1)
Set VMC = guidedCal.CustomCalConfiguration
VMC.ConnectorType(1) = "APC 3.5 male"
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcquireStep</td>
<td>IVMCType</td>
<td>Acquire the measurement data for the specified step in the calibration process.</td>
</tr>
<tr>
<td>GenerateErrorTerms</td>
<td>IVMCType</td>
<td>Generates the error terms for the calibration.</td>
</tr>
<tr>
<td>GenerateSteps</td>
<td>IVMCType</td>
<td>Returns the number of steps required to complete the calibration.</td>
</tr>
<tr>
<td>GetStepDescription</td>
<td>IVMCType</td>
<td>Returns the description of the specified step in the calibration process.</td>
</tr>
<tr>
<td>Initialize</td>
<td>IVMCType</td>
<td>Begins a calibration.</td>
</tr>
</tbody>
</table>

See History
<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoOrient</td>
<td>Sets ECAL module automatic orientation ON or OFF.</td>
</tr>
<tr>
<td>CalKitType</td>
<td>Sets and returns a calibration kit type for calibration.</td>
</tr>
<tr>
<td>CharacterizeMixerOnly</td>
<td>Sets and returns whether to perform a mixer characterization ONLY or full 2-port calibration.</td>
</tr>
<tr>
<td>CharFileName</td>
<td>Specifies the .S2P mixer characterization file name.</td>
</tr>
<tr>
<td>CharMixerReverse</td>
<td>Specifies the direction in which to characterize the calibration mixer.</td>
</tr>
<tr>
<td>CompatibleCalKits</td>
<td>Returns a list of cal kits that are compatible with the connector type for the specified port.</td>
</tr>
<tr>
<td>ConnectorType</td>
<td>Sets or queries the connector type for the specified port.</td>
</tr>
<tr>
<td>Do1PortEcal</td>
<td>Specify ECAL or Mechanical calibration for the mixer characterization portion of a VMC calibration.</td>
</tr>
<tr>
<td>Do2PortEcal</td>
<td>Specify ECAL or Mechanical calibration for the 2-port calibration portion of a VMC calibration.</td>
</tr>
<tr>
<td>EcalCharacterization</td>
<td>Specifies the characterization data within an ECal module to be used for the calibration.</td>
</tr>
<tr>
<td>EcalOrientation1Port</td>
<td>For Mixer Characterization ONLY - Specifies which port of the ECal module is connected to which port of the VNA</td>
</tr>
<tr>
<td>EcalOrientation2Port</td>
<td>For full 2-port VMC cal - Specifies which port of the ECal module is connected to which port of the VNA</td>
</tr>
<tr>
<td>EnableLOPowerCal</td>
<td>Perform LO Power Cal</td>
</tr>
<tr>
<td>LoadCharFromFile</td>
<td>Specifies and loads a mixer characterization (S2P) file.</td>
</tr>
<tr>
<td>NetworkFilename</td>
<td>Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement.</td>
</tr>
<tr>
<td>NetworkMode</td>
<td>Embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement.</td>
</tr>
<tr>
<td>OmitIsolation</td>
<td>Superseded - Replaced by SetIsolationPaths and</td>
</tr>
</tbody>
</table>
GetIsolationPaths

Sets and returns whether Isolation portion of the calibration will be performed or not.

ThruCalMethod

Sets and returns the method for performing the thru portion of the calibration.

ValidConnectorTypes

Returns a list of connector types for which there are calibration kits.

IVMCType History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IVMCType</td>
<td>3.5</td>
</tr>
<tr>
<td>IVMCType2</td>
<td>3.53</td>
</tr>
<tr>
<td>IVMCType3</td>
<td>6.0</td>
</tr>
<tr>
<td>IVMCType4</td>
<td>9.1</td>
</tr>
</tbody>
</table>
VSA Object

Description

Controls the Vector Signal Analyzer Application settings.

Accessing the VSA object

```vba
Dim app as AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")
app.CreateCustomMeasurementEx 2 ,"VSA", "Port 2(B)"
Dim VSA
Set VSA = app.ActiveChannel.CustomChannelConfiguration
```

VSA Application - VSA Setup tab

- `chan.StartFrequency`
- `chan.StopFrequency`
- `vsa.CenterFrequency`
- `chan.FrequencySpan`
- `FrequencySpanFull`
- `CenterFrequencyStepSize`
- `CenterFrequencyStepSizeMode`
- `Receiver Attenuator Property`
- `vsa.VSACOnnection`
Source Setup tab

IF Setup tab

SourceStartFrequency
SourceStopFrequency
SourceCWFrequency
SourceSweepType
SourcePointCount
SASweepRepeatCount
chan.TestPortPower

Element

vsa.ADCFilter
vsa.ADCSampleRate
vsa.ADCDither
Advanced Settings tab

See Also:

- **Example Program**: Create a VSA measurement
- About the Vector Signal Analyzer Application
- VNA Automation Interfaces
- The VNA Object Model
## AbortSweepCmd Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the <strong>Abort Sweep command</strong> for an external DC Source and an external DC Meter.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>extDC.AbortSweepCmd = value</code></td>
</tr>
</tbody>
</table>
| **Variable** | (Type) - Description  
  *extDC* A `ExternalDCDevice` (object)  
  *value* (String) The SCPI command used to abort or reset the DC Source or DC Meter. |
| **Return Type** | String |
| **Default** | " " Empty String |
| **Examples** | `extDC.AbortSweepCmd = "ABORt" 'Write`  
  `value = extDC.AbortSweepCmd 'Read` |
| **C++ Syntax** | HRESULT get_AbortSweepCmd( BSTR* cmd);  
  HRESULT put_AbortSweepCmd( BSTR cmd); |
| **Interface** | IExternalDCDevice3 |
## AbsoluteExtractionToneLevel Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the absolute tone power level.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>HotS22.AbsoluteExtractionToneLevel = level</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>HotS22</code></td>
<td>A <code>ActiveParametersApp (object)</code></td>
</tr>
<tr>
<td><code>level</code></td>
<td>(Double) - Absolute tone power level</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>-5 dBm</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>HotS22.AbsoluteExtractionToneLevel = -5 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>level = HotS22.AbsoluteExtractionToneLevel 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_AbsoluteExtractionToneLevel(double* level)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_AbsoluteExtractionToneLevel(double level)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IActiveChannelSettings</td>
</tr>
</tbody>
</table>
About Trigger

AcceptTriggerBeforeArmed Property

**Description**
Determines what happens to an EDGE trigger signal if it occurs before the VNA is ready to be triggered. (LEVEL trigger signals are always ignored.) For more information, see External triggering.

**VB Syntax**
```
trigsetup.AcceptTriggerBeforeArmed = boolean
```

**Variable**
```
trigsetup - A TriggerSetup2 (object)
```

**Boolean**
Choose from:

- **False** - A trigger signal is ignored if it occurs before the VNA is ready to be triggered.
- **True** - A trigger signal is remembered and then used when the VNA becomes armed (ready to be triggered). The VNA remembers only one trigger signal.

**Return Type**
Boolean

**Default**
False

**Examples**
```
trigsetup.AcceptTriggerBeforeArmed = True 'Write
```

**C++ Syntax**
```
HRESULT get_AcceptTriggerBeforeArmed( BOOL *pVal);
HRESULT put_AcceptTriggerBeforeArmed( BOOL newVal);
```

**Interface**
ITriggerSetup2
### AcquisitionDirection Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the direction of each part of a 2-port calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>cal.AcquisitionDirection = value</code></td>
</tr>
<tr>
<td>Variable (Type)</td>
<td><code>cal</code> A Calibrator (object)</td>
</tr>
<tr>
<td></td>
<td><code>value</code> (enum NADirection) - Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <code>naForward</code> - measures the forward direction</td>
</tr>
<tr>
<td></td>
<td>1 - <code>naReverse</code> - measures the reverse direction</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long Integer</td>
</tr>
<tr>
<td>Default</td>
<td><code>naForward</code></td>
</tr>
<tr>
<td>Examples</td>
<td><code>cal.AcquisitionDirection = naForward</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT AcquisitionDirection(tagNADirection dir);</td>
</tr>
<tr>
<td>Interface</td>
<td>ICalibrator</td>
</tr>
</tbody>
</table>
# About Compression Mode

## AcquisitionMode Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the method by which gain compression data is acquired.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>gca.AcquisitionMode = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>gca</code></td>
<td>A GainCompression <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(NAGCAAcquisitionMode)</em> Choose from:</td>
</tr>
<tr>
<td></td>
<td>- <code>naSmartSweep (0)</code> Iterate quickly to find compression point</td>
</tr>
<tr>
<td></td>
<td>- <code>naSweepPowerAtEachFreq2D (1)</code> Sweep power at each frequency</td>
</tr>
<tr>
<td></td>
<td>- <code>naSweepFreqAtEachPower2D (2)</code> Sweep frequency at each power level</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum</td>
</tr>
<tr>
<td>Default</td>
<td><code>naSmartSweep</code></td>
</tr>
<tr>
<td>Examples</td>
<td><code>gca.AcquisitionMode = naSmartSweep</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>acqMode = gca.AcquisitionMode</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_AcquisitionMode(tagNAGCAAcquisitionMode* mode)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_AcquisitionMode(tagNAGCAAcquisitionMode mode)</td>
</tr>
<tr>
<td>Interface</td>
<td>IGainCompression</td>
</tr>
</tbody>
</table>
# AcquisitionTime Property

**Description**
Returns the LO acquisition time which is the ADC Record Size x ADC Sampling Frequency (10 nsec or 40 nsec) x (1 + Stacking) x (Video Average.Coefficient).

**VB Syntax**
```vbnet
value = sa.AcquisitionTime
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sa</code></td>
<td>A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Variable to store the returned number of LOs.</td>
</tr>
</tbody>
</table>

Learn about these settings.

**Return Type**
Double

**Default**
N/A

**Examples**
```vbnet
value = sa.AcquisitionTime 'Read
```

See an example program.

**C++ Syntax**
```cpp
HRESULT get_AcquisitionTime(double* val);
```

**Interface**
ISpectrumAnalyzer
ActiveCalKit Property

Description

Returns a handle to the Active CalKit object. The active cal kit is the kit selected for use in **Unguided** calibrations.

You can either (1) use the handle directly to access CalKit properties and methods, or (2) set a variable to the CalKit object. The variable retains a handle to the original object if another CalKit becomes active.

**VB Syntax**

1) `app.ActiveCalKit.<setting>`
   
   or
   
   2) `Set cKit = app.ActiveCalKit`

**Variable (Type) - Description**

- `app` An **Application** (object)
- `<setting>` A CalKit property (or method) and arguments
- `cKit` (object) - A CalKit object

**Return Type** CalKit object

**Default** None

**Examples**

```
Public cKit as calKit
Set cKit = app.ActiveCalKit 'read
```

**C++ Syntax**

```
HRESULT get_ActiveCalKit (ICalkit * kit)
```

**Interface** IApplication
### ActiveChannel Property

**Description**
Returns a handle to the Active Channel object. You can either (1) use the handle directly to access channel properties and methods, or (2) set a variable to the channel object. The variable retains a handle to the original channel if another channel becomes active.

**VB Syntax**
1. `app.ActiveChannel.<setting>`
   or
2. `Set chan = app.ActiveChannel`

**Variable**
- `chan` - A Channel *(object)*
- `app` - An Application *(object)*
- `<setting>` - A channel property (or method) and arguments

**Return Type** Channel object

**Default** Not applicable

**Examples**
1. `app.ActiveChannel.Averaging = 1`
2. `Public chan as Channel
   Set chan = app.ActiveChannel`

**C++ Syntax**
`HRESULT get_ActiveChannel(IChannel* *pVal)`

**Interface** IApplication
Active (ExtDev) Property

Description
Set and return the state of activation of the device. When `extDev.IOEnable = True`, and this command is set to True, the VNA will attempt communication with the external device. An error is returned if communication cannot be verified.

**Note:** Send this command AFTER sending other External Device settings to avoid communicating with the device before it has been fully configured.

See Also: ExternalDeviceDeActivatePolicy Property - Determines whether External Devices remain activated or are de-activated when the VNA is Preset or when a Instrument State is recalled.

**VB Syntax**
```
extDevices.Active = value
```

**Variable (Type) - Description**

*extDevices* An ExternalDevice (object)

*value* (Boolean) Choose from:

- **True** - Device is active.
- **False** - Device is NOT active.

Return Type
Boolean

Default
False - When configured using the front panel user interface, the device is ON (activated) by default.

**Examples**
```
extDevices.Active = True 'Write
bool = extDevices.Active 'Read
```

See example program to configure PMAR device
See example program to configure External Source

**C++ Syntax**
```
HRESULT get_Active( VARIANT_BOOL* value);

HRESULT put_Active( VARIANT_BOOL newVal);
```

**Interface**
IExternalDevices
ActiveMarker Property

Description
Returns a handle to the Active Marker object. You can either (1) use the handle directly to access Marker properties and methods, or (2) set a variable to the Marker object. The variable retains a handle to the original object if another Marker becomes active.

VB Syntax
1) `meas.ActiveMarker.<setting>`
or
2) Set `mark = meas.ActiveMarker`

Variable (Type) - Description
meas (object) - An Measurement object
<setting> A marker property (or method) and arguments
mark (object) - A marker object

Return Type
marker object

Default
None

Examples
`Public mark as marker`
`Set mark = meas.ActiveMarker`

C++ Syntax
`HRESULT get_ActiveMarker(IMarker** marker)`

Interface
IMeasurement
**ActiveMeasurement Property**

**Description**
Returns a handle to the Active Measurement object. You can either (1) use the handle directly to access measurement properties and methods, or (2) set a variable to the measurement object. The variable retains a handle to the original measurement.

**VB Syntax**
1) `app.ActiveMeasurement.<setting>`
2) `Set meas = app.ActiveMeasurement`

**Variable (Type) - Description**
- `meas` A Measurement (object)
- `app` An Application (object)
- `<setting>` A measurement property (or method) and arguments

**Return Type**
Measurement object

**Default**
None

**Examples**
1) `app.ActiveMeasurement.Averaging = 1`
2) `Public meas as Measurement
   Set meas = app.ActiveMeasurement`

**C++ Syntax**
`HRESULT get_ActiveMeasurement(IMeasurement **ppMeas)`

**Interface**
IApplication
ActiveNAWindow Property

Description
Returns a handle to the ActiveWindow object. You can either (1) use the handle directly to access window properties and methods, or (2) set a variable to the window object. The variable retains a handle to the original window if another window becomes active.

VB Syntax
1) `app.ActiveNAWindow.<setting>
or
2) Set `win = app.ActiveNAWindow`

Variable (Type) - Description
- `win` A NAWindow (object)
- `app` An Application (object)
- `<setting>` A NAWindow property (or method) and arguments

Return Type
A NAWindow object

Default
Not applicable

Examples
```vbnet
Public win as NAWindow
Set win = app.ActiveWindow
```

C++ Syntax
```cpp
HRESULT get_ActiveNAWindow(INAWindow **ppWindow)
```

Interface
IApplication
**ActiveTrace Property**

**Description**
Returns a handle to the Active Trace object. You can either (1) use the handle directly to access trace properties and methods, or (2) set a variable to the trace object. The variable retains a handle to the original trace if another trace becomes active.

**VB Syntax**
1) `win.ActiveTrace.<setting>`
   or
2) Set `trce = win.ActiveTrace`

**Variable (Type) - Description**
- `trce`: A Trace (object)
- `win`: An NAWindow (object)

**<setting>:** A trace property (or method) and arguments

**Return Type:** An NAWindow object

**Default:** None

**Examples**
1) `win.ActiveTrace.Autoscale`
2) Public `trce as Trace`
   `Set trce = Application.ActiveNAWindow.ActiveTrace`

**C++ Syntax**
`HRESULT get_ActiveTrace(ITrace* *pVal)`

**Interface**
INAWindow
# ActiveBackground Property

**Description**: Set and return the background color of the active window for the VNA display or hardcopy print.

**VB Syntax**

```vbnet
colors.ActiveBackground = value
```

**Variable**

- **Type**: `ComColors` (object)
- **Value**: `Long Integer` - RGB color of the ActiveBackground pen.

Convert the three RGB colors to an integer as follows:

```
RGB = R + (G * 2^8) + (B * 2^16)
```

To find the three RGB values from the Display Colors dialog, click **Change Color**, then **Define Custom Color**.

**Return Type**: Long

**Default**: Display = 0,0,24

**Examples**

```
R = 10
G = 10
B = 10
RGB = R + (G * 2^8) + (B * 2^16)
```

```vbnet
colors.ActiveBackground = RGB 'Write
color = colors.ActiveBackground 'Read
```

**C++ Syntax**

```csharp
HRESULT get_ActiveBackground(long* pVal);

HRESULT put_ActiveBackground(long newVal);
```

**Interface**: IColors2
### ActiveLabels Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return the labels and grid frame colors in the active window for the VNA display or hardcopy print. (Active labels, Grid frame)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>colors.ActiveLabels = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>colors</strong> <em>(Type) - Description</em></td>
</tr>
<tr>
<td></td>
<td><strong>(Object) - A ComColors</strong></td>
</tr>
<tr>
<td></td>
<td><strong>value</strong> <em>(Long Integer) - RGB color of the ActiveLabels pen.</em></td>
</tr>
<tr>
<td></td>
<td>Convert the three RGB colors to an integer as follows:</td>
</tr>
<tr>
<td></td>
<td><code>RGB = R + (G * 2^8) + (B * 2^16)</code></td>
</tr>
<tr>
<td></td>
<td>To find the three RGB values from the Display Colors dialog, click <strong>Change Color</strong>, then <strong>Define Custom Color</strong>.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Display = 175,175,175</td>
</tr>
<tr>
<td></td>
<td>Print = 0,0,0 (Black)</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>R = 10</code></td>
</tr>
<tr>
<td></td>
<td><code>G = 10</code></td>
</tr>
<tr>
<td></td>
<td><code>B = 10</code></td>
</tr>
<tr>
<td></td>
<td><code>RGB = R + (G * 2^8) + (B * 2^16)</code></td>
</tr>
<tr>
<td></td>
<td><code>colors.ActiveLabels = RGB 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>color = colors.ActiveLabels 'Read</code></td>
</tr>
</tbody>
</table>

### C++ Syntax

- `HRESULT get_ActiveLabels(long* pVal);`
- `HRESULT put_ActiveLabels(long newVal);`

### Interface

- `IColors`
### ActiveXAxisRange Property

**Description**  
For FCA and GCX measurements, sets the swept parameter to display on the X-axis. This command does not change the default setting for new traces.

Use `Converter.ActiveXAxisRange` to change all existing traces and make the setting the default setting for new traces.

This command is NOT used for NFX, IMDX, and IMSX measurements. Use `Converter.ActiveXAxisRange`.

**VB Syntax**

```vbnet
mixer.ActiveXAxisRange = value
```

**Variable**  
*(Type)* - Description

- `mixer` A Mixer *(object)*
- `value` *(Enum as MixerStimulusRange)* - Parameter to display on the X-axis.

**Return Type**  
Enum

**Default**  
OUTPUT

**Examples**

```vbnet
mixer.ActiveXAxisRange = 1 'Write
variable = mixer.ActiveXAxisRange 'Read
```

**C++ Syntax**

```cpp
HRESULT get_ActiveXAxisRange(tagMixerStimulusRange *Val)

HRESULT put_ActiveXAxisRange(tagMixerStimulusRange newVal)
```

**Interface**  
IMixer3
### ActiveXAxisRange Property

**Description**
Sets the swept frequency range to display on the X-axis for all existing traces and sets the default for all future traces.

**VB Syntax**

```vbnet
obj.ActiveXAxisRange = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>obj</code></td>
<td>A Converter Object (for IMDX and NFX measurements) or A Measurement Object (for GCX, SMC, and VMC measurements)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Enum as ConverterStimulusRange)</td>
</tr>
</tbody>
</table>

**Swept stimulus range to display on the X-axis. Choose from:**

- **0 - naInputRange** - Input frequency range
- **1 - naLO1Range** - LO 1 frequency range
- **2 - naLO2Range** - LO 2 frequency range
- **3 - naOutputRange** - Output frequency range
- **4 - naPerMeasurementRange** - reserved for future use.

If the specified frequency range is not swept, the default swept range is used.

**Return Type**
Enum

**Default**
Search is performed in the following order until a swept range is found:

1. OUTPUT
2. INPUT (If the OUTPUT is fixed)
3. Number of Points (If ALL ranges are fixed)

**Examples**

```vbnet
conv.ActiveXAxisRange = naInputRange 'Write
```

```vbnet
variable = meas.ActiveXAxisRange 'Read
```

**C++ Syntax**

```c++
HRESULT get_ActiveXAxisRange(tagConverterStimulusRange range *Val)
```

```c++
HRESULT put_ActiveXAxisRange(tagConverterStimulusRange range newVal)
```
Interface  IConverter

IMeasurement14
**ADCCaptureMode Property**

**Description**
Sets and returns the ADC capture mode modeled as a 2-pole switch in the diagram on the SignalProcessingModuleFour page. The switch either bypasses or routes the IF through the 3-stage digital filter.

**VB Syntax**
```vbnet
spm4.ADCCaptureMode = value
```

**Variable**
*spm4* - Description

- **spm4**: A SignalProcessingModuleFour *(object)*
- **value**: *(Enum as NASTates)* Capture mode.

- **naOFF (0)** - The digital filters are used to process IF information. The filters can be configured automatically or manually using FilterMode Property.

- **naON (1)** - The digital filters are bypassed and the raw ADC readings are taken directly. With DSP 4 versions, a maximum of 4096 data points per sweep can be acquired.

With DSP 5 versions, the VNA maximum data points per sweep can be acquired.

Learn more about DSP Versions.

**Return Type**
Enum

**Default**
OFF

**Examples**
```vbnet
spm4.ADCCaptureMode = 0 'Write
mode = spm4.ADCCaptureMode 'Read
```

**C++ Syntax**
```cpp
HRESULT get_ADCCaptureMode(tagNASTates* pCaptureMode);
HRESULT put_ADCCaptureMode(tagNASTates pCaptureMode);
```

**Interface**
ISignalProcessingModuleFour
### ADCDelay Property

**Description**  
Returns the ADC delay for pulse measurements.

**VB Syntax**  
`value = pulse.ADCDelay`

**Variable** (Type) - Description

- **pulse**  
  A PulseGenerator (object)

- **value**  
  (Double) - Variable to store the returned ADC delay.

**Return Type**  
Double

**Default**  
Not Applicable

**Examples**  
`value = pulse.ADCDelay`  
Read

**C++ Syntax**  
`HRESULT get_ADCDelay(double* delay);`

**Interface**  
IPulseGenerator5
**ADCEnableFIRFor25Mhz Property**

**Description**
Set and read the FIR filter for 25 MHz decimation.

**VB Syntax**
\[ sa.ADCEnableFIRFor25Mhz = value \]

**Variable**
- **(Type)** - Description
  - \( sa \) - A SpectrumAnalyzer (object)
  - \( value \) - (Boolean) Choose from:
    - **0** - OFF - Disable 25 MHz FIR filter.
    - **1** - ON - Enable 25 MHz FIR filter.

**Return Type**
Boolean

**Default**
0

**Examples**
\[
\begin{align*}
sa.ADCEnableFIRFor25Mhz = 0 & \quad \text{'Write} \\
\text{filter} = sa.ADCEnableFIRFor25Mhz & \quad \text{'Read}
\end{align*}
\]

**C++ Syntax**

HRESULT put_ADCEnableFIRFor25Mhz(VARIANT_BOOL bEnable);

HRESULT get_ADCEnableFIRFor25Mhz(VARIANT_BOOL* bEnable);

**Interface**
ISpectrumAnalyzer4
**Description**  
Set and read the ADC filter width mode. The entered frequency value is rounded to the closest value supported by the VNA (11 MHz or 38 MHz).

**VB Syntax**  
```
sa.ADCFilter = value
```

**Variable**  
(Type) - Description  
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sa</td>
<td>A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Double) Choose a value within the range supported by the VNA.</td>
</tr>
</tbody>
</table>

Learn about these choices.

**Return Type**  
Double

**Default**  
11 MHz

```vbnet
sa.ADCFilter = 11MHz 'Write
value = sa.ADCFilter 'Read
```

**Examples**  
See an example program.

**C++ Syntax**  
```cpp
HRESULT put_ADCFilter(double cutfreq);

HRESULT get_ADCFilter(double* cutfreq);
```

**Interface**  
ISpectrumAnalyzer
### ADCMultRecPeriod Property

**Description**  
Set and read the period to wait between ADC record chunks.

**VB Syntax**  
```vb
sa.ADCMultRecPeriod = value
```

**Variable**  
- **Type**: (object)  
- **Value**  
  - `sa`  
  - A `SpectrumAnalyzer` object
  - `value`  
  - `Long`  
  - Choose a value between 64 and 33554432.

**Return Type**  
Long

**Default**  
64

**Examples**  
```vb
sa.ADCMultRecPeriod = 256 'Write
value = sa.ADCMultRecPeriod 'Read
```

See an example program.

**C++ Syntax**  
```c++
HRESULT put_ADCMultRecPeriod(long val);
HRESULT get_ADCMultRecPeriod(long* val);
```

**Interface**  
`ISpectrumAnalyzer`
About Multiple Recordings

ADCMultRecSize Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the size of the ADC record chunks.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>sa.ADCMultRecSize = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td></td>
<td><code>sa</code> A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td></td>
<td><code>value</code> (Long) Choose a value between 1 and (ADC record size / 2).</td>
</tr>
</tbody>
</table>

Learn about these settings.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Long</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>32</td>
</tr>
</tbody>
</table>
| Examples    | `sa.ADCMultRecSize = 256` 'Write'  
`value = sa.ADCMultRecSize` 'Read' |

See an example program.

| C++ Syntax  | HRESULT put_ADCMultRecSize(long val);  
HRESULT get_ADCMultRecSize(long* val); |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>ISpectrumAnalyzer</td>
</tr>
</tbody>
</table>
**ADCMultRecState Property**

**Description**  
Set and read the ON/OFF state of the multiple recording function. Multiple recording allows the ADC Record Size to be divided and acquired in smaller "chunks" and also to specify a wait period between these acquisitions.

**VB Syntax**  
`sa.ADCMultRecState = value`

**Variable**  
- **sa** - A SpectrumAnalyzer *(object)*
- **value** - *(Boolean)* Choose from:
  - **0 - OFF** - ADC record size "chunking" OFF.
  - **1 - ON** - ADC record size "chunking" ON.

Learn about these settings.

**Return Type**  
Boolean

**Default**  
0

**Examples**  
`sa.ADCMultRecState = OFF`  
`value = sa.ADCMultRecState`

See an example program.

**C++ Syntax**  
```cpp
HRESULT put_ADMultRecState(VARIANT_BOOL bEnable);
HRESULT get_ADMultRecState(VARIANT_BOOL* bEnable);
```

**Interface**  
ISpectrumAnalyzer
ADCRecordSize Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the current ADC record size.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = sa.ADCRecordSize</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>sa</code></td>
<td>A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) Variable to store the returned value.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Example</td>
<td><code>value = sa.ADCRecordSize</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_ADCRecordSize(long* val);</td>
</tr>
<tr>
<td>Interface</td>
<td>ISpectrumAnalyzer5</td>
</tr>
</tbody>
</table>
## ADCRecordSizeMax Property

**Description**  
Read the current maximum available ADC record size.

**VB Syntax**  
`value = sa.ADCRecordSizeMax`

**Variable**  
*(Type) - Description*  
- `sa` A `SpectrumAnalyzer (object)`  
- `value` *(Long)* Variable to store the returned value.

**Return Type**  
Long

**Example**  
`value = sa.ADCRecordSizeMax 'Read`

**C++ Syntax**  
`HRESULT get_ADCRecordSizeMax(long* val);`

**Interface**  
ISpectrumAnalyzer5
### ADCRecordSizeMin Property

**Description**  
Read the current minimum available ADC record size.

**VB Syntax**  
```vb
value = sa.ADCRecordSizeMin
```

**Variable**  
- `sa`: A `SpectrumAnalyzer` (object)
- `value`: (Long) Variable to store the returned value.

**Return Type**  
Long

**Example**  
```vb
value = sa.ADCRecordSizeMin 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_ADCRecordSizeMin(long* val);
```

**Interface**  
`ISpectrumAnalyzer5`
**ADCSampleRate Property**

**Description**
Set and read the ADC sample frequency setting. The entered frequency is rounded to the closest value supported by the VNA (25 MHz or 100 MHz).

**VB Syntax**

```
sa.ADCSampleRate = value
```

**Variable**

- **sa** (Type) - Description: A SpectrumAnalyzer object
- **value** (Double) Choose from 100 MHz or 25 MHz.

Learn about these settings.

**Return Type**

Double

**Default**

100 MHz

**Examples**

```
sa.ADCSampleRate = 100MHz  'Write
value = sa.ADCSampleRate 'Read
```

See an example program.

**C++ Syntax**

```
HRESULT put_ADCSampleRate(double rate);
HRESULT get_ADCSampleRate(double* rate);
```

**Interface**

ISpectrumAnalyzer
## ADCStackingMax Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the current maximum available ADC stacking size.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>value = sa.ADCStackingMax</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>sa</td>
<td>A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Long) Variable to store the returned value.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>value = sa.ADCStackingMax <code>Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_ADCStackingMax(long* val);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISpectrumAnalyzer5</td>
</tr>
</tbody>
</table>
## ADCStackingState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the ON/OFF state of the ADC sample stacking.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>sa.ADCStackingState = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>sa</code></td>
<td>A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean) Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>0 - OFF</strong> - ADC sample stacking is set to OFF.</td>
</tr>
<tr>
<td></td>
<td><strong>1 - ON</strong> - ADC sample stacking is set to ON.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>sa.ADCStackingState = OFF</code> Write</td>
</tr>
<tr>
<td></td>
<td><code>value = sa.ADCStackingState</code> Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT put_ADCStackingState(VARIANT_BOOL bEnable);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_ADCStackingState(VARIANT_BOOL* bEnable);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISpectrumAnalyzer</td>
</tr>
</tbody>
</table>
### ADCStacking Property

**Description**  
Set and read the ADC stacking value.

**VB Syntax**  
```
sa.ADCStacking = value
```

**Variable**  
**Type** - Description

- **sa**: A `SpectrumAnalyzer` (object)
- **value**: (Double) Choose a value between 0 and 65535.

**Learn about these settings.**

**Return Type**  
Double

**Default**  
0

**Examples**  
```
sa.ADCStacking = 1  Write
value = sa.ADCStacking  'Read
```

**See an example program.**

**C++ Syntax**  
```
HRESULT put_ADCStacking(double val);
HRESULT get_ADCStacking(double* val);
```

**Interface**  
`ISpectrumAnalyzer`
### AfterSweepCmd Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the After Sweep command for an external DC Source and an external DC Meter.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>extDC.AfterSweepCmd = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>extDC</code></td>
<td>A <code>ExternalDCDevice</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(String) The SCPI command to be sent at the end of a sweep.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>&quot; &quot; Empty String</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>extDC.AfterSweepCmd = &quot;OUTP OFF&quot;</code> 'Write`</td>
</tr>
<tr>
<td></td>
<td><code>value = extDC.AfterSweepCmd</code> 'Read`</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_AfterSweepCmd( BSTR* cmd);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_AfterSweepCmd( BSTR cmd);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IExternalDCDevice3</td>
</tr>
</tbody>
</table>
# ALCLevelingMode Property

**Description**  
Sets and returns the ALC mode for the specified channel and port. Use `GetSupportedALCModes` to return a list of valid ALC modes for the VNA.

Learn more about ALC mode.

**VB Syntax**  
```
chan.ALCLevelingMode (sourcePort) = value
```

**Variable**  
*(Type) - Description*

- `chan` *(object)* - A `Channel` object
- `sourcePort` *(long integer)* - The source port for which to make this setting. If ports are remapped, specify the logical port number.

**Note:** If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source VNA-x model, then you must use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

**value** *(enum as `naALCLevelingMode`)* - Choose from:

- 0 `naALCInternal`
- 2 `naALCOpenLoop` (PNA-X only)

**Return Type**  
Enum

**Default**  
`naALCInternal`

**Examples**  
```vbnet
chan.ALCLevelingMode(1) = 2 'Write
state = chan.ALCLevelingMode(4) 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_ALCLevelingMode(long port, tagNAALCLevelingMode* pVal);

HRESULT put_ALCLevelingMode(long port,tagNAALCLevelingMode newVal);
```

**Interface**  
`IChannel9`
**Alignment Property**

**Description**  
Sets the way the PNA computes the DC value of the frequency-domain measurement. The correct DC value is required for inverse-FFT accuracy, and if not estimated properly, can cause distortions in the time-domain measurement in the form of an undesired slope in the waveform.

**VB Syntax**  
trans.Alignment = value

**Variable**  
- **trans**  
  A Transform (object)
- **value**  
  (Enum As NATransformAlignment) - Choose from:

  0 - naTransformAlignmentLegacy - The DC value is extrapolated using three data points. The transform offset is calculated using the delay of the first frequency point. This is the same algorithm used in the HP 8510 network analyzer.

  1 - naTransformAlignmentNormalize - The DC value is extrapolated using three data points. The transform offset is set to zero at t=0 minus six rise-times. This mode requires that a good S-parameter calibration has been performed, which can be verified by observing a flat time-domain response at t=0 when measuring a load located at the physical point corresponding to t=0. Setting the time domain trace to zero at a time before t=0 stabilizes the trace for determining impedances after time t=0, resulting in improved behavior compared to Legacy mode. This method is similar to that used with PLTS, and is very useful in determining the time-domain-transform response of transmission lines and printed-circuit-board characteristics.

**Return Type**  
Enum

**Default**  
0 - naTransformAlignmentNormalize

**Examples**  
trans.Alignment = naTransformAlignmentNormalize 'Write
value = trans.Alignment  'Read

**C++ Syntax**  
HRESULT get_Alignment(enum tagNATransformAlignment *pVal);

HRESULT put_Alignment(enum tagNATransformAlignment newVal);

**Interface**  
ITransform3
AllowArbitrarySegments Property

Description: Enables you to setup a segment sweep with arbitrary frequencies. The start and stop frequencies of each segment can overlap other segments. Also, each segment can have a start frequency that is greater than its stop frequency which causes a reverse sweep over that segment. Learn more about Arbitrary Segment Sweep.

VB Syntax: `segs.AllowArbitrarySegments = value`

Variable (Type) - Description:

- `segs`: A `Segments` collection (object)
- `value`: (boolean)
  - `True`: Allows the setup of arbitrary segment sweep.
  - `False`: Prevents the setup of arbitrary segment sweep.

Return Type: Boolean

Default: False

Examples:

- `segs.AllowArbitrarySegments = True` 'Write
- `AllowArbSegs = AllowArbitrarySegments` 'Read

C++ Syntax:

- `HRESULT get_AllowArbitrarySegments(VARIANT_BOOL *pVal)`
- `HRESULT put_AllowArbitrarySegments(VARIANT_BOOL newVal)`

Interface: ISegments3
AlternateSweep Property

Sets sweeps to either alternate or chopped.

VB Syntax

```vbnet
object.AlternateSweep = value
```

Variable (Type) - Description

- **object**
  - Channel (object)
  - or
  - CalSet (object) - Read-only property

Value (boolean) - Choose either:

- **False** - Sweep mode set to **Chopped** - reflection and transmission are measured on the same sweep.
- **True** - Sweep mode set to **Alternate** - reflection and transmission measured on separate sweeps. Improves Mixer bounce and Isolation measurements. Increases cycle time.

Return Type

boolean

Default

False (0)

Examples

```vbnet
chan.AlternateSweep = True 'Write
altSwp = chan.AlternateSweep 'Read
```

C++ Syntax

```c++
HRESULT AlternateSweep(VARIANT_BOOL *pVal)
HRESULT AlternateSweep(VARIANT_BOOL newVal)
```

Interface

IChannel

ICalSet3
### AmbientTemperature Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the temperature at which the current noise measurement is occurring. Learn more.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>noiseCal.AmbientTemperature = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>noiseCal</code></td>
<td>A NoiseCal <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(double)</em> Ambient temperature in Kelvin.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>295</td>
</tr>
<tr>
<td>Examples</td>
<td><code>noise.AmbientTemperature = 289</code> <em>Write</em></td>
</tr>
<tr>
<td></td>
<td><code>temp = noise.AmbientTemperature</code> <em>Read</em></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_AmbientTemperature(Double* pValue)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_AmbientTemperature(Double pNewValue)</td>
</tr>
<tr>
<td>Interface</td>
<td>INoiseCal</td>
</tr>
</tbody>
</table>
## AnalysisCWFreq Property

**Description**  
Set and return the CW frequency for a compression analysis trace.

**VB Syntax**  
`gcaMeas.AnalysisCWFreq = value`

**Variable**  
(Type) - Description

- **gcaMeas**  
  A `GainCompressionMeas (object)`

- **value**  
  (Double)  
  CW frequency in Hz. Choose a frequency within the range of the gain compression channel.

**Return Type**  
Double

**Default**  
Not Applicable

**Examples**  
`gcaMeas.AnalysisCWFreq = 1e9  'Write`

`cwfreq = gca.AnalysisCWFreq  'Read`

**C++ Syntax**  
`HRESULT get_AnalysisCWFreq(Double* value)`

`HRESULT put_AnalysisCWFreq(Double  value)`

**Interface**  
`IGainCompressionMeas`
## AnalysisEnable Property

| Description       | Set and read the (ON | OFF) state of Gain Compression Analysis. |
|-------------------|---------------------------------------------|
| **VB Syntax**     | `gcaMeas.AnalysisEnable = value`            |
| **Variable**      | (Type) - Description                        |
| `gcaMeas`         | A GainCompressionMeas (object)              |
| `value`           | (Boolean) Choose from:                      |
|                   | **False** - Disable GCA analysis trace.     |
|                   | **True** - Enable GCA analysis trace.       |
| **Return Type**   | Boolean                                     |
| **Default**       | False                                       |
| **Examples**      | `gcaMeas.AnalysisEnable = True 'Write`      |
|                   | `analysis = gca.AnalysisEnable 'Read`      |
| **C++ Syntax**    | HRESULT get_AnalysisEnable(VARIANT_BOOL* value) |
|                   | HRESULT put_AnalysisEnable(VARIANT_BOOL  value) |
| **Interface**     | IGainCompressionMeas                       |
### AnalysisIsDiscreteFreq Property

**Description**
Sets and returns whether the CW frequency for the compression analysis trace can be set to only the discrete frequencies or provides interpolation.

**VB Syntax**

```
gcaMeas.AnalysisIsDiscreteFreq = value
```

**Variable (Type) - Description**

- **gcaMeas**
  A GainCompressionMeas (object)

- **value**
  (Boolean) Choose from:
  - **False** - Interpolated data points.
  - **True** - Discrete data points only.

**Return Type**
Boolean

**Default**
False

**Examples**

```
gcaMeas.AnalysisIsDiscreteFreq = True 'Write
isDisc = gca.AnalysisIsDiscreteFreq 'Read
```

**C++ Syntax**

```
HRESULT get_AnalysisIsDiscreteFreq(VARIANT_BOOL* value)
HRESULT put_AnalysisIsDiscreteFreq(VARIANT_BOOL  value)
```

**Interface**
IGainCompressionMeas
### AnalysisXAxis Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the type of data to display on the x-axis of a compression analysis trace.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>gcaMeas.AnalysisXAxis = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>gcaMeas</code></td>
<td>A <code>GainCompressionMeas (object)</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Enum as NAGCAAnalysisXAxis) Choose from:</td>
</tr>
<tr>
<td></td>
<td><code>nAPsourceAsXAxis (0)</code> - Power from the source.</td>
</tr>
<tr>
<td></td>
<td><code>nAPinAsXAxis (1)</code> - Input power to the DUT.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum</td>
</tr>
<tr>
<td>Default</td>
<td><code>nAPinAsXAxis (1)</code></td>
</tr>
<tr>
<td>Examples</td>
<td><code>gcaMeas.AnalysisXAxis = nAPinAsXAxis 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>xAxis = gca.AnalysisXAxis 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_AnalysisXAxis(tagNAGCAAnalysisXAxis* value)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_AnalysisXAxis(tagNAGCAAnalysisXAxis value)</td>
</tr>
<tr>
<td>Interface</td>
<td>IGainCompressionMeas</td>
</tr>
</tbody>
</table>
### Application Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the name of the Analyzer making measurements on the channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan.Application</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>chan</code> - <strong>Type</strong> - Description</td>
</tr>
<tr>
<td></td>
<td>A Channel <strong>object</strong></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>object</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>None</td>
</tr>
</tbody>
</table>
| **Examples** | `rfna = chan.Application`  
  *returns the Analyzer name* |
| **C++ Syntax** | HRESULT get_Application(IApplication** Application) |
| **Interface** | IChannel |
**ArrangeWindows Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the arrangement of all the windows. Overlay, Stack2, Split3 and Quad4 will create windows.</th>
</tr>
</thead>
<tbody>
<tr>
<td>To control the state of one window, use <code>app.WindowState</code>.</td>
<td></td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
app.ArrangeWindows = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum <code>NAWindowModes</code>) - Choose from:</td>
</tr>
</tbody>
</table>

- 0 - naTile
- 1 - naCascade
- 2 - naOverlay
- 3 - naStack2
- 4 - naSplit3
- 5 - naQuad4
- 6 - naTracePerWindow
- 7 - naChannelPerWindow
- 8 - naLtoR

**Return Type**

Not Applicable

**Default**

naTile

**Examples**

```vbnet
app.ArrangeWindow = naTile 'Write
```

**C++ Syntax**

```cpp
HRESULT put_ArrangeWindows(tagNAWindowModes newVal)
```

**Interface**

IApplication
About Attenuation

AttenuatorMode Property

Description
Sets or returns the mode of operation of the attenuator control for the specified port number. This command is automatically set to Manual when an Attenuator value is set.

VB Syntax
object.AttenuatorMode(portNum) = value

Variable (Type) - Description

object Channel (object)

or

CalSet (object) - Read-only property

portNum (long) - Port number of attenuator control to be changed.

value (enum NAModes) - Choose from:

0 - naAuto - Attenuator control set to automatic. The analyzer will set the attenuator control appropriately to deliver the specified power at the source.


Return Type
NAModes

Default
0 - Auto

Examples
chan.AttenuatorMode(1) = naAuto 'Write
attn = chan.AttenuatorMode(1) 'Read

C++ Syntax
HRESULT get_AttenuatorMode(long port, tagNAModes* pVal)
HRESULT put_AttenuatorMode(long port, tagNAModes newVal)

Interface
IChannel
ICalSet3
Attenuator Property

**Description**
Sets or returns the value of the source attenuator for the specified port number. Sending this command automatically sets AttenuatorMode to Manual.

**VB Syntax**

```
object.Attenuator(portNum) = value
```

**Variable**

- **object** Channel (object)

  **or**

  CalSet (object) - Read-only property

- **portNum** (long integer) - Port number of source attenuator to be changed.

- **value** (double) - Attenuation value. The range of settable values depends on the VNA model. To determine the valid settings, do one of the following:

  - See VNA models and options to see the range and step size for each model / option.
  - To determine the maximum attenuator value use MaximumSourceStepAttenuator.

If an invalid attenuation setting is entered, the VNA will select the next lower valid value. For example, if 19 is entered, then for an E8361A, 10 dB attenuation will be selected.

**Return Type**
Double

**Default**
20 dB

**Examples**

```
chan.Attenuator(1) = 20 'Write
attn = chan.Attenuator(cnum) 'Read
```

**C++ Syntax**

```
HRESULT get_Attenuator(long port, double *pVal)
HRESULT put_Attenuator(long port, double newVal)
```

**Interface**

- IChannel
- ICalSet3
AutoBandwidth Property

Description
Set and read the default values for DFT bandwidth.

VB Syntax
```
sa.dft.AutoBandwidth = value
```

Variable
(Type) - Description
- `sa.dft` A SpectrumAnalyzerDFT (object)
- `value` (Boolean) Choose from:
  - **0 - OFF** - DFT minimum and maximum values are set manually:
    - Narrow - 500 kHz to 11 MHz
    - Wide - 500 kHz to 44 MHz
  - **1 - ON** - DFT minimum and maximum values are set to their default values:
    - Narrow - 1 MHz to 10 MHz
    - Wide - 1 MHz to 34 MHz

Return Type
Boolean

Default
1

Examples
```
sa.dft.AutoBandwidth = 0  'Write
bwidth = sa.dft.AutoBandwidth  'Read
```

C++ Syntax
```
HRESULT put_AutoBandwidth(VARIANT_BOOL bEnable);
HRESULT get_AutoBandwidth(VARIANT_BOOL* bEnable);
```

Interface
ISpectrumAnalyzerDFT
### AutoCWSweepTime Property

**Description**
This command replaces AutoIFBandWidth Property.

Sets and returns the state of automatic CW sweep time (used in Pulse Profile mode).

**VB Syntax**
```
pulseMeas.AutoCWSweepTime = bool
```

**Variable**
- **(Type)** - Description
- **pulseMeas** - A PulseMeasurementControl *(object)*
  - **bool** - True - In Pulse Profile mode, adjusts the default X-axis start time to zero and the stop time to double the Pulse Width. This allows you to see one complete pulse.
  - **False** - The Sweep Time is not changed.

**Return Type**
Boolean

**Default**
True

**Examples**
```
pulse.AutoCWSweepTime = True 'Write
value = pulse.AutoCWSweepTime 'Read
```

**C++ Syntax**
```
HRESULT get_AutoCWSweepTime(VARIANT_BOOL *pVal);
HRESULT put_AutoCWSweepTime(VARIANT_BOOL newVal);
```

**Interface**
IPulseMeasurementControl2
**AutoDetection Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Choose to automatically or manually set pulse mode (Narrowband or Wideband) for the channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pulseMeas.AutoDetection = bool</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>pulseMeas</code></td>
<td>A PulseMeasurementControl (object)</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><strong>False</strong> - Manually set the pulse mode. Use <code>WideBandDetectionState</code> to set the pulse mode. <strong>True</strong> - Automatically set the pulse mode.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>True</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pulse.AutoDetection = True</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>value = pulse.AutoDetection</code> 'Read'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_AutoDetection(VARIANT_BOOL *pVal);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_AutoDetection(VARIANT_BOOL newVal);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPulseMeasurementControl</td>
</tr>
</tbody>
</table>
### AutoIFBandWidth Property - Superseded

<table>
<thead>
<tr>
<th>Description</th>
<th>This command is replaced by: AutoCWSweepTime Property. In Wideband pulse mode, choose to set the IF bandwidth automatically or manually.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pulseMeas.AutoIFBandWidth = bool</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>pulseMeas</code></td>
<td>A PulseMeasurementControl (object)</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><strong>False</strong> - Manually set the IFBW for the measurement. <strong>True</strong> - Automatically set the IFBW for the measurement.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>True</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pulse.AutoIFBandWidth = True</code>  'Write  <code>value = pulse.AutoIFBandWidth</code>  'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_AutoIFBandWidth(VARIANT_BOOL *pVal);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_AutoIFBandWidth(VARIANT_BOOL newVal);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPulseMeasurementControl</td>
</tr>
</tbody>
</table>
# AutoIFBWAdjustment Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Set and read auto IFBW adjustment ON</th>
<th>OFF state for Gain Compression measurements.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( gca.)AutoIFBWAdjustment = value</td>
<td></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>( gca ) (GainCompression object)</td>
<td>(value) Auto IFBW adjustment state. Choose from:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>False - Sets auto IFBW adjustment OFF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>True - Sets auto IFBW adjustment ON</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>ON</td>
<td></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>( gca.)AutoIFBWAdjustment = True ( 'Write )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>aifbw = gca.AutoIFBWAdjustment ( 'Read )</td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_AutoIFBWAdjustment(VARIANT_BOOL* bState)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HRESULT put_AutoIFBWAdjustment(VARIANT_BOOL bState)</td>
<td></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IGainCompression</td>
<td></td>
</tr>
</tbody>
</table>
About Pulse Measurements

AutoOptimizePRF Property

<table>
<thead>
<tr>
<th>Description</th>
<th>In Narrowband pulse mode, choose to set the Pulse Repetition Frequency automatically or manually. This is labeled &quot;Optimize Pulse Frequency&quot; on the user-interface. To make changes manually, use MasterFrequency Property or MasterPeriod Property.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>pulse.Meas.AutoOptimizePRF = bool</td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description [pulseMeas] A PulseMeasurementControl [object] bool</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Manually set the PRF for the measurement.</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong>  - Automatically set the PRF for the measurement.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>True</td>
</tr>
<tr>
<td>Examples</td>
<td>pulse.AutoOptimizePRF = True 'Write</td>
</tr>
<tr>
<td></td>
<td>value = pulse.AutoOptimizePRF 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_AutoOptimizePRF(VARIANT_BOOL *pVal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_AutoOptimizePRF(VARIANT_BOOL newVal);</td>
</tr>
<tr>
<td>Interface</td>
<td>IPulseMeasurementControl</td>
</tr>
</tbody>
</table>
AutoOrient Property

Description: Sets ECAL module automatic orientation ON or OFF.

VB Syntax: `obj.AutoOrient = bool`

Variable (Type) - Description

`obj` SMCTYPE (object)

or

VMCTYPE (object)

`bool` (Boolean)

True - Set AutoOrientation ON

False - Set AutoOrientation OFF

Return Type: Boolean

Default: True

Examples: `Smc.AutoOrient = True`

C++ Syntax: `HRRESULT put_AutoOrient(VARIANT_BOOL bAutoOrient);`

`HRRESULT get_AutoOrient(VARIANT_BOOL *bAutoOrient);`

Interface: SMCTYPE

VMCTYPE
<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Sets the state of auto orientation for a noise tuner during Noise Figure for NFX.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>nfCal.AutoOrientTuner = bool</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>nfCal</code></td>
<td>A <code>NoiseCal</code> <em>(object)</em></td>
</tr>
<tr>
<td><code>bool</code></td>
<td>(Boolean)</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>Set AutoOrientTuneration ON</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>Set AutoOrientTuneration OFF</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>True</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>nfCal.AutoOrientTuner = True</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_AutoOrientTuner(VARIANT_BOOL bEnable);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_AutoOrientTuner(VARIANT_BOOL *bEnable);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>INoiseCal2</code></td>
</tr>
</tbody>
</table>
**AutoPortExtConfig Property**

**Description**
Sets the frequency span that is used to calculate Automatic Port Extension. Learn more about calculating Automatic Port Extension.

**VB Syntax**
```
fixture.AutoPortExtConfig = value
```

**Variable (Type) - Description**
- **fixture**
  A Fixturing (object)
- **value**
  (ENUM as NAAutoPortExtConfig)

0 **naAPEC_CSPN** - Use current span.

1 **naAPEC_AMKR** - Use active marker frequency.

2 **naAPEC_USPN** - Use custom user span. Use **AutoPortExtSearchStart Property** and **AutoPortExtSearchStop Property** to specify start and stop frequency.

**Return Type**
ENUM

**Default**
0 **naAPEC_CSPN**

**Examples**
```
fixture.AutoPortExtConfig = naAPEC_AMKR
value = fixture.AutoPortExtConfig 'Read
```

**C++ Syntax**
```
HRESULT get_AutoPortExtConfig(tagNAAutoPortExtConfig *pVal );

HRESULT put_AutoPortExtConfig(tagNAAutoPortExtConfig Val );
```

**Interface**
IFixturing2
### AutoPortExtDCOffset Property

**Description**
Specifies whether or not to include DC Offset as part of automatic port extension. Learn more about Automatic DC Offset. Only allowed when AutoPortExtLoss Property is set to ON.

**VB Syntax**
```vbnet
fixture.AutoPortExtDCOffset = bool
```

**Variable (Type) - Description**
- **fixture**: A Fixturing (object)
- **bool**: True - Includes DC Offset correction.
  - False - Does NOT include DC Offset correction.

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
fixture.AutoPortExtDCOffset = True
value = fixture.AutoPortExtDCOffset 'Read
```

**C++ Syntax**
```cpp
HRESULT get_AutoPortExtDCOffset(VARIANT_BOOL *pState);
HRESULT put_AutoPortExtDCOffset(VARIANT_BOOL bState);
```

**Interface**
IFixturing2
AutoPortExtLoss Property

**Description**
Specifies whether or not to include loss correction as part of automatic port extension. Learn more about Loss Compensation in port extension.

**VB Syntax**
```vbnet
fixture.AutoPortExtLoss = bool
```

**Variable**
- **Type** - Description
  - `fixture` A Fixturing (object)
  - `bool` True - Includes Loss correction.
    - False - Does NOT include Loss correction.

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
fixture.AutoPortExtLoss = True
value = fixture.AutoPortExtLoss 'Read
```

**C++ Syntax**
```c++
HRESULT get_AutoPortExtLoss(VARIANT_BOOL *pState);
HRESULT put_AutoPortExtLoss(VARIANT_BOOL bState);
```

**Interface**
IFixturing2
AutoPortExtSearchStart Property

**Description**
Set the start frequency for custom user span. Only applies when `fixture.AutoPortExtConfig = 0 naAPEC_CSPN`.

Learn more about User Span.

**VB Syntax**

```
fixture.AutoPortExtSearchStart = value
```

**Variable**

- **Type**: Description
- **fixture**: A Fixturing (object)
- **value**: (Double) User span start value. Must be within the frequency range of the active channel and less than the value set by `AutoPortExtSearchStop Property`

**Return Type**
Double

**Default**
Start frequency of the current active channel.

**Examples**

```
fixture.AutoPortExtSearchStart = 1E9
value = fixture.AutoPortExtSearchStart 'Read
```

**C++ Syntax**

```
HRESULT get_AutoPortExtSearchStart(double *pdVal);
HRESULT put_AutoPortExtSearchStart(double dVal);
```

**Interface**
IFxituring2
AutoPortExtSearchStop Property

**Description**
Set the stop frequency for custom user span. Only applies when `fixture.AutoPortExtConfig = 0 naAPEC_CSPN`.

Learn more about User Span.

Only applies when `fixture.AutoPortExtConfig = 0 naAPEC_CSPN`.

**VB Syntax**
```
fixture.AutoPortExtSearchStop = value
```

**Variable (Type) - Description**
- **fixture** (A Fixturing (object))
- **value** (Double) User span stop value. Must be within the frequency range of the active channel and greater than the value set by `AutoPortExtSearchStart Property`.

**Return Type**
Double

**Default**
Stop frequency of the current active channel.

**Examples**
```
fixture.AutoPortExtSearchStop = 1E9
value = fixture.AutoPortExtSearchStop  'Read
```

**C++ Syntax**
```
HRESULT get_AutoPortExtSearchStop(double *pdVal);

HRESULT put_AutoPortExtSearchStop(double dVal);
```

**Interface**
IFixturing2
**AutoPortExtState Property**

**Description**
Enables and disables automatic port extensions on the specified port. All enabled ports will have their reference plane automatically adjusted after performing Automatic Port Extension.

**VB Syntax**
```vbnet
fixture.AutoPortExtState (port) = bool
```

**Variable (Type) - Description**
- `fixture` (A Fixturing `object`)
- `port` (Port number to enable or disable. `Integer`)
- `bool` (Boolean)
  - **True** - Enables Auto Port Extensions
  - **False** - Disables Auto Port Extensions

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
fixture.AutoPortExtState(1) = True
value = fixture.AutoPortExtState(2)  'Read
```

**C++ Syntax**
```cpp
HRESULT get_AutoPortExtState(short port, VARIANT_BOOL *pState );
HRESULT put_AutoPortExtState(short port, VARIANT_BOOL bVal);
```

**Interface**
IFixturing2
### AutoPulseTiming Property

**Description**
In Narrowband pulse mode, choose to set the delay and width automatically or manually. This is labeled "Autoselect Width and Delay" on the user-interface.

**VB Syntax**
```
pulseMeas.AutoPulseTiming = bool
```

**Variable (Type) - Description**
- **pulseMeas**: A `PulseMeasurementControl` (object)
- **bool**: 
  - **False**: Manually set the delay and width for the measurement.
  - **True**: Automatically set the delay and width for the measurement.

**Return Type**
Boolean

**Default**
True

**Examples**
```
pulse.AutoPulseTiming = True  'Write
value = pulse.AutoPulseTiming  'Read
```

**C++ Syntax**
```
HRESULT get_AutoPulseTiming(VARIANT_BOOL *pVal);
HRESULT put_AutoPulseTiming(VARIANT_BOOL newVal);
```

**Interface**
`IPulseMeasurementControl`
# About Pulse Measurements

## AutoSelectPulseGen Property

**Description**
In Narrowband pulse mode, choose to set the pulse generator used to drive the source modulation automatically or manually.

**VB Syntax**

```vbnet
pulseMeas.AutoSelectPulseGen = bool
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pulseMeas</td>
<td>A PulseMeasurementControl (object)</td>
</tr>
<tr>
<td>bool</td>
<td></td>
</tr>
</tbody>
</table>

- **False** - Manually set source modulation drive for the measurement.
- **True** - Automatically set source modulation drive for the measurement.

**Return Type**
Boolean

**Default**
True

**Examples**

```vbnet
pulse.AutoSelectPulseGen = True 'Write
value = pulse.AutoSelectPulseGen 'Read
```

**C++ Syntax**

```c++
HRESULT get_AutoSelectPulseGen(VARIANT_BOOL *pVal);
HRESULT put_AutoSelectPulseGen(VARIANT_BOOL newVal);
```

**Interface**
IPulseMeasurementControl
### AuxiliaryTriggerCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of aux trigger input / output connector pairs in the VNA</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = app.AuxiliaryTriggerCount</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long Integer)</em> Variable to store the returned value.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 = PNA-X models</td>
</tr>
<tr>
<td></td>
<td>1 = All other VNA models</td>
</tr>
<tr>
<td></td>
<td>An Application <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>ioConns = app.AuxiliaryTriggerCount</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT AuxiliaryTriggerCount(long *count);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication11</td>
</tr>
</tbody>
</table>
**AuxTriggerScopeIsGlobal Property**

**Description**
Sets the Trigger OUT behavior to either Global or Channel. Learn more about this setting.

This command will cause the VNA to Preset.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

See the AuxTrigger Object.

**VB Syntax**
`pref.AuxTriggerScopeIsGlobal = value`

**Variable (Type) - Description**
- `pref` (A Preferences object)
- `value` (Boolean) - Choose from:
  - **True** - Trigger properties apply to ALL channels (Global).
    - Default setting for E836x and PNA-L models.
    - Allows use of command to configure the external trigger properties.
    - "Per Point" trigger property is not settable. Use the channel's Point trigger setting.
  - **False** - External Trigger properties apply to each channel independently.
    - Default setting for PNA-X models.
    - Must use AuxTrigger commands to configure the external trigger properties. `ExternalTriggerConnectionBehavior Property` will NOT work.
    - "Per Point" trigger output property is set using the channel's Point trigger setting AND `TriggerOutInterval Property`.

**Return Type**
Boolean

**Default**
- **True** - E836xB and PNA-L models
- **False** - PNA-X models

**Examples**
- `pref.AuxTriggerScopeIsGlobal = 1`  'Write
- `auxTrigPref = pref.AuxTriggerScopeIsGlobal`  'Read
C++ Syntax

HRESULT get_AuxTriggerScopeIsGlobal(VARIANT_BOOL * pref);
HRESULT put_AuxTriggerScopeIsGlobal(VARIANT_BOOL pref);

Interface
IPreferences5
## AvailableMeasurementClasses Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a list of available measurement classes on the VNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.AvailableMeasurementClasses</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Variant) - Variable to store the returned list of measurement classes.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**Examples**

```vba
'Read all measurement classes
Set app = CreateObject("AgilentPNA835x.Application")
Set cap = app.Capabilities
meas = cap.AvailableMeasurementClasses
dim i
For i = 0 To UBound(meas)
    msg = msg & meas(i) & vbCrLf
Next
MsgBox msg
```

**C++ Syntax**

```cpp
HRESULT get_AvailableMeasurementClasses(Variant *value);
```

**Interface**

ICapabilities7
**AverageMode Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the type of averaging to perform: Point or Sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan.AverageMode = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(<em>Type</em>) - Description</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>A Channel (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(<em>Enum as naAverageMode</em>) - Average Type. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <strong>naPoint</strong>  Averaging measurements are made on each data point before stepping to the next data point.</td>
</tr>
<tr>
<td></td>
<td>1 - <strong>naSweep</strong>  Averaging measurements are made on subsequent sweeps until the required number of averaging sweeps are performed.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1 - <strong>naSweep</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chan.AverageMode = naSweep</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>avgType = chan.AverageMode</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_AverageMode(NAAverageMode * mode);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_AverageMode(NAAverageMode mode);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel16</td>
</tr>
</tbody>
</table>
About Averaging

AveragingCount Property

Description
Returns the number of sweeps that have been acquired and averaged into the measurements on this channel. AveragingFactor specifies the number of sweeps to average. AveragingCount indicates the progress toward that goal.

VB Syntax
value = chan.AveragingCount

Variable (Type) - Description
chan A Channel (object)
value (Long Integer) - Variable to store the returned count

Return Type
Long Integer

Default
Not Applicable

Example
avgcount = chan.AveragingCount

C++ Syntax
HRESULT get_AveragingCount(long* count)

Interface
IChannel
### AveragingFactor Property

**Description**  
Specifies the number of measurements to combine for an average. Must also turn averaging ON by setting `chan.Averaging = 1`.

**VB Syntax**  
`chan.AveragingFactor = value`

**Variable**  
- **Type** - Description  
  - `chan`  
    - A Channel (object)  
  - `value`  
    - (Long Integer) - Number of measurement sweeps to average. Choose any number between 1 and 65536 ($2^{16}$).

**Return Type**  
Long Integer

**Default**  
1

**Examples**  
- `chan.AveragingFactor = 5`  
  - `Write`
- `avgfact = chan.AveragingFactor`  
  - `Read`

**C++ Syntax**  
- `HRESULT get_AveragingFactor(long *pVal)`
- `HRESULT put_AveragingFactor(long newVal)`

**Interface**  
IChannel
### Averaging Property

**Description**

Turns trace averaging ON or OFF for all measurements on the channel. Averaging is only allowed on ratioed measurements; not on single input measurements.

**VB Syntax**

```vbnet
chan.Averaging = state
```

**Variable**

- **(chan)**: A Channel (object)
- **(state)**: (boolean)
  - **False**: Turns averaging OFF
  - **True**: Turns averaging ON

**Return Type**

Boolean

**Default**

False

**Examples**

```vbnet
chan.Average = True 'Write
averag = chan.Averaging 'Read
```

**C++ Syntax**

```csharp
HRESULT get_Averaging(BOOL *pVal)
HRESULT put_Averaging(BOOL newVal)
```

**Interface**

IChannel
## AvoidSpurs Property

**Description**
Sets and returns the state of the avoid spurs feature.

**VB Syntax**
```
mixer.AvoidSpurs = boolean
```

**Variable**

**Type** - Description

- **mixer** A Mixer *(object)*
  
  A Converter *(object)*

- **value** *(Boolean)* - State of avoid spurs feature. Choose from

  - **False** Avoid spurs OFF
  - **True** Avoid spurs ON

**Return Type**
Boolean

**Default**
False

**Examples**
```
conv.AvoidSpurs = True 'Write

variable = conv.AvoidSpurs 'Read
```

**C++ Syntax**

```
HRESULT get_AvoidSpurs(Bool *bVal)

HRESULT put_AvoidSpurs(Bool newVal)
```

**Interface**
IMixer3

IConverter5
B_BalPortNegative Property

Description
Returns the negative balanced port number in the balanced DUT topology.

Use SetBPort Method to set the single balanced port DUT topology.

VB Syntax
var = balTopology.B_BalPortNegative

Variable (Type) - Description
balTopology A BalancedTopology (object)
var (Long Integer) Variable to store the returned value.

Return Type Long Integer
Default Not Applicable
Examples variable = balTopology.B_BalPortNegative  'Read

C++ Syntax HRESULT get_B_BalPortNegative(long *NegPort)

Interface IBalancedTopology3
B_BalPortPositive Method

Description
Returns the positive balanced port number in the balanced DUT topology.
Use SetBPort Method to set the single balanced port DUT topology.

VB Syntax
var = balTopology.B_BalPortPositive

Variable (Type) - Description
balTopology A BalancedTopology (object)
var (Long Integer) Variable to store the returned value.

Return Type
Long Integer

Default
Not Applicable

Examples
variable = balTopology.B_BalPortPositive 'Read

C++ Syntax
HRESULT get_B_BalPortPositive(long *PosPort)

Interface
IBalancedTopology3
### Background Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return the background color for the VNA display or hardcopy print.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>colors.Background = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>colors</code></td>
<td>A <strong>ComColors</strong> <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long Integer)</em> - RGB color of the Background pen.</td>
</tr>
</tbody>
</table>

Convert the three RGB colors to an integer as follows:

\[
RGB = R + (G \times 2^8) + (B \times 2^{16})
\]

To find the three RGB values from the Display Colors dialog, click **Change Color**, then **Define Custom Color**.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Long</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Display = 0,0,0 (Black)</td>
</tr>
<tr>
<td></td>
<td>Print = 255,255,255 (White)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
<th>R = 10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>G = 10</td>
</tr>
<tr>
<td></td>
<td>B = 10</td>
</tr>
<tr>
<td>RGB</td>
<td>= R + (G \times 2^8) + (B \times 2^{16})</td>
</tr>
<tr>
<td><code>colors.Background = RGB</code></td>
<td><strong>Write</strong></td>
</tr>
<tr>
<td><code>color = colors.Background</code></td>
<td><strong>Read</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C++ Syntax</th>
<th>HRESULT get_Background(long* pVal);</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HRESULT put_Background(long newVal);</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Interface</th>
<th>IColors</th>
</tr>
</thead>
</table>
### BackOff Property

**Description**
Sets and returns the backoff value used to calculate various PNOP parameters. Also set PinOffset Property.

A sweep must be executed (single or continuous) and SearchPowerNormalOperatingPoint Method must be sent before reading marker results.

To turn off the PNOP markers, either turn them off individually or DeleteAllMarkers.

To search a User Range with the PNOP search, first activate marker 1. The user range used with the PNOP search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

**VB Syntax**

```
pnop.BackOff = value
```

**Variable**

- **Type** - Description
- **pnop** - a PNOP (object)
- **value** - (double) - Backoff value in dB. Choose any number between -500 and 500

**Return Type**
Double

**Default**
0 dB

**Examples**

```
backoff = pnop.BackOff 'Read
```

See example program

**C++ Syntax**

- HRESULT put_BackOff(double newVal);
- HRESULT get_BackOff(double* pNewVal)

**Interface**
IPNOP
# BackOffGain Property

**Description**
Returns the BackOffGain result of the PNOP marker search.

\[ \text{PBO Gain} = \text{PBO Out} - \text{PBO In} \]

**VB Syntax**
```vbnet
bOffGain = pnop.BackOffGain
```

**Variable**
- **Type**: (double) - Variable to store returned value
- **pnop**: A PNOP (object)

**Return Type**
Double

**Default**
Not applicable

**Examples**
```vbnet
bOffGain = pnop.BackOffGain 'Read
```

[See example program](#)

**C++ Syntax**
```cpp
HRESULT get_BackOffGain(double* pNewVal)
```

**Interface**
IPNOP
## BackOffPin Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the BackOffPin result of the PNOP marker search.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>( bOffPin = \text{pnop.BackOffPin} )</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>( bOffPin )</td>
<td>(double) - Variable to store returned value</td>
</tr>
<tr>
<td>( \text{pnop} )</td>
<td>A PNOP (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>( bOffPin = \text{pnop.BackOffPin} ) 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_BackOffPin(double* pNewVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IPNOP</td>
</tr>
</tbody>
</table>

PBO In = Marker 2 X-axis

See example program
### BackOffPout Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the BackOffPout result of the PNOP marker search.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PBO Out</strong></td>
<td>Marker 2 Y-axis</td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
bOffPout = pnop.BackOffPout
```

**Variable**

- **(Type)** - Description
- **bOffPout** *(double)* - Variable to store returned value
- **pnop** - A PNOP *(object)*

**Return Type**

Double

**Default**

Not applicable

**Examples**

```vbnet
bOffPout = pnop.BackOffPout 'Read
```

See example program

**C++ Syntax**

```cpp
HRESULT get_BackOffPout(double* pNewVal)
```

**Interface**

IPNOP
### BalancedMode Property

**Description**
Sets and returns whether the balanced transform is ON or OFF

**VB Syntax**

```vbnet
balMeas.BalancedMode = value
```

**Variable (Type) - Description**

- `value` *(Boolean)* - State of balanced transform. Choose from
  - **False** Balanced Transform OFF
  - **True** Balanced Transform ON

**Return Type**
Boolean

**Default**
**False**

**Examples**

```vbnet
balMeas.BalancedMode = True 'Write
variable = balMeas.BalancedMode 'Read
```

**C++ Syntax**

```cpp
HRESULT get_BalancedMode(VARIANT_BOOL *bVal)

HRESULT put_BalancedMode(VARIANT_BOOL newVal)
```

**Interface**
IBalancedMeasurement
**BalancedPortTrueState(bpnum) Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the True Mode state for a specified balanced port.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>balStim.BalancedPortTrueState(bpnum) = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>balStim</code></td>
<td>A BalancedStimulus <em>(object)</em></td>
</tr>
<tr>
<td><code>bpnum</code></td>
<td>Balanced port number. This corresponds to the Balanced Port number in the GUI.</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Boolean)</em> Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>ON or 1</strong> Turns True Mode ON.</td>
</tr>
<tr>
<td></td>
<td><strong>OFF or 0</strong> Turns True Mode OFF.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
<tr>
<td>Examples</td>
<td><code>balStim.BalancedPortTrueState(2) = ON</code></td>
</tr>
<tr>
<td></td>
<td><code>var = balStim.BalancedPortTrueState(2)</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_BalancedPortTrueState (int bpnum, VARIANT_BOOL *pState)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_BalancedPortTrueState (int bpnum, VARIANT_BOOL pState)</td>
</tr>
<tr>
<td>Interface</td>
<td>IBalancedStimulus3</td>
</tr>
</tbody>
</table>
## BalPort1PhaseOffset Property

**Description**
Sets and returns the phase offset between the two ports that comprise Balanced port 1. `balStim.Mode` must be set to a True Stimulus mode. Applicable only with Opt S93460A/B - iTMSA.

**VB Syntax**
```
balStim.BalPort1PhaseOffset = value
```

**Variable**
- **Type**: Description
- **balStim**: A `BalancedStimulus` (object)
- **value**: (Double) - Phase Offset in degrees. Choose a value between -360 and 360.

**Return Type**
- **Double**

**Default**
- **0**

**Examples**
```
balStim.BalPort1PhaseOffset = 10 'Write

variable = balStim.BalPort1PhaseOffset 'Read
```

**C++ Syntax**
- `HRESULT get_BalPort1PhaseOffset (double *pVal)`
- `HRESULT put_BalPort1PhaseOffset (double newVal)`

**Interface**
- `IBalancedStimulus`
### BalPort1StartPhase Property

**Description**
Sets and returns the start phase of a phase sweep.

**VB Syntax**

```
balStim.BalPort1StartPhase = value
```

**Variable**
*balStim* - A BalancedStimulus (object)

*value* - (Double) - Start phase in degrees. Choose a value between -360 and 360.

**Return Type**
Double

**Default**
0

**Examples**

```vbnet
balStim.BalPort1StartPhase = 10 'Write
variable = balStim.BalPort1StartPhase 'Read
```

**C++ Syntax**

```cpp
HRESULT get_BalPort1StartPhase (double *pVal)
HRESULT put_BalPort1StartPhase (double newVal)
```

**Interface**
IBalancedStimulus2
### BalPort1StopPhase Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the stop phase of a phase sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>\textit{balStim.BalPort1StopPhase} = \textit{value}</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>\textit{balStim}</td>
<td>A \textit{BalancedStimulus} (object)</td>
</tr>
<tr>
<td>\textit{value}</td>
<td>(Double) - Stop phase in degrees. Choose a value between -360 and 360.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
<tr>
<td>Examples</td>
<td>\textit{balStim.BalPort1StopPhase} = 10 \texttt{Write}</td>
</tr>
<tr>
<td></td>
<td>\texttt{variable} = \textit{balStim.BalPort1StopPhase} \texttt{Read}</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_BalPort1StopPhase (double *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_BalPort1StopPhase (double newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IBalancedStimulus2</td>
</tr>
</tbody>
</table>
### BalPort2PhaseOffset Property

**Description**  
Sets and returns the phase offset between the two ports that comprise Balanced port 2. `balStim.Mode` must be set to a True Stimulus mode. Applicable only with Opt S93460A/B - iTMSA.

**VB Syntax**  
`balStim.BalPort2PhaseOffset = value`

**Variable**  
`balStim` - A `BalancedStimulus` object

`value` - (Double) - Phase Offset in degrees. Choose a value between -360 and 360.

**Return Type**  
Double

**Default**  
0

**Examples**  
`balStim.BalPort2PhaseOffset = 10 'Write`

`variable = balStim.BalPort2PhaseOffset 'Read`

**C++ Syntax**  
`HRESULT get_BalPort2PhaseOffset (double *pVal)`

`HRESULT put_BalPort2PhaseOffset (double newVal)`

**Interface**  
IBalancedStimulus
## BalPort2PowerOffset Property

**Description**
Sets and returns the power offset between the two ports that comprise Balanced port 2. \( \text{balStim.Mode} \) must be set to a True Stimulus mode. Applicable only with Opt S93460A/B - iTMSA.

**VB Syntax**

\[
\text{balStim.BalPort2PowerOffset} = \text{value}
\]

**Variable**

- **balStim**
  - A **BalancedStimulus (object)**

- **value**
  - **(Double)** - Power Offset in dB.

**Return Type**
Double

**Default**
0

**Examples**

```
balStim.BalPort2PowerOffset = 2 'Write
variable = balStim.BalPort2PowerOffset  'Read
```

**C++ Syntax**

```c++
HRESULT get_BalPort2PowerOffset (double *pVal)
HRESULT put_BalPort2PowerOffset (double newVal)
```

**Interface**
IBalancedStimulus
### BalPort2StartPhase Property

**Description**
Sets and returns the start phase of a phase sweep.

**VB Syntax**
```vbnet
balStim.BalPort2StartPhase = value
```

**Variable**
- **balStim** (Type) - Description
  A BalancedStimulus (object)
- **value** (Double) - Start phase in degrees. Choose a value between -360 and 360.

**Return Type**
Double

**Default**
0

**Examples**
```vbnet
balStim.BalPort2StartPhase = 10 'Write

variable = balStim.BalPort2StartPhase 'Read
```

**C++ Syntax**
```cpp
HRESULT get_BalPort2StartPhase (double *pVal)

HRESULT put_BalPort2StartPhase (double newVal)
```

**Interface**
IBalancedStimulus2
# BalPort2StopPhase Property

Sets and returns the stop phase of a phase sweep.

**VB Syntax**

\[
\text{balStim.BalPort2StopPhase} = \text{value}
\]

**Variable**

*balStim* A *BalancedStimulus* (object)

*value* (Double) - Stop phase in degrees. Choose a value between -360 and 360.

**Return Type**

Double

**Default**

0

**Examples**

\[
\begin{align*}
\text{balStim.BalPort2StopPhase} &= 10 \quad \text{'Write} \\
\text{variable} &= \text{balStim.BalPort2StopPhase} \quad \text{'Read}
\end{align*}
\]

**C++ Syntax**

\[
\begin{align*}
\text{HRESULT get_BalPort2StopPhase (double \ast pVal)} \\
\text{HRESULT put_BalPort2StopPhase (double newVal)}
\end{align*}
\]

**Interface**

IBalancedStimulus2
BalSMeasurement Property

Description
Sets and returns the measurement for the Balanced - Single-ended topology.

VB Syntax
`balMeas.BalSMeasurement = value`

Variable (Type) - Description
`balMeas` A `BalancedMeasurement` (object)
`value` (String) - Balanced - Single-ended measurement parameter. Not case sensitive. Choose from:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sdd11</td>
<td>Sdc11</td>
<td>Sds12</td>
</tr>
<tr>
<td>Scd11</td>
<td>Scs11</td>
<td>Scs12</td>
</tr>
<tr>
<td>Ssd21</td>
<td>Ssc21</td>
<td>Sss22</td>
</tr>
<tr>
<td>Imb</td>
<td>CMRR1</td>
<td>CMRR2</td>
</tr>
<tr>
<td></td>
<td>(Ssd21/Ssc21)</td>
<td>(Sds12/Scs12)</td>
</tr>
</tbody>
</table>

Return Type
String

Default
Sdd11

Examples
`balMeas.BalSMeasurement = "Sdd11"` 'Write
`variable = balMeas.BalSMeasurement` 'Read

C++ Syntax
`HRESULT get_BalSMeasurement(BSTR *pVal)`
`HRESULT put_BalSMeasurement(BSTR newVal)`

Interface
`IBalancedMeasurement3`
**BandDensityACPRState Property**

**Description**  
Sets and reads the ACPR density marker.

**VB Syntax**  
`mkr.BandDensityACPRState = value`

**Variable**  
*(Type) - Description*

- `mkr` A Marker *(object)*
- `value` *(Enum as NAStates)* Choose from:
  - `naOFF` (or 0) - Turn ACPR density marker OFF
  - `naON` (or 1) - Turn ACPR density marker ON

**Return Type**  
Boolean

**Default**  
`naOFF`

**Examples**

```
Write
mkr.BandDensityACPRState = 0

value = mkr.BandDensityACPRState 'Read
```

**C++ Syntax**

```
HRESULT get_BandDensityACPRState(tagNAStates* pState)

HRESULT put_BandDensityACPRState(tagNAStates pState)
```

**Interface**  
IMarker9
About Spectrum Analyzer Markers

BandDensityBW Property

**Description**: Sets and reads the bandwidth of the band density marker.

**VB Syntax**: `mkr.BandDensityBW = value`

**Variable**

- `mkr` (Type) - Marker (object)
- `value` (Double) - Choose a bandwidth

**Return Type**: Double

**Default**: 1 MHz

**Examples**

```
 mkr.BandDensityBW = 1e6 'Write

value = mkr.BandDensityBW 'Read
```

**C++ Syntax**

- `HRESULT get_BandDensityBW(double* pVal)`
- `HRESULT put_BandDensityBW(double newVal)`

**Interface**: IMarker8
# BandDensityEQSPan Property

**Description**  
Sets and reads the frequency span used by Power Density to normalize the power.

**VB Syntax**  
\[ mkr.BandDensityEQSPan = value \]

**Variable**  
- **mkr**  
  A Marker (object)
- **value**  
  (Double) Choose a span

**Return Type**  
Double

**Default**  
1 MHz

**Examples**  
- \[ mkr.BandDensityEQSPan = 1e6 \]  
  'Write
- \[ value = mkr.BandDensityEQSPan \]  
  'Read

**C++ Syntax**  
- HRESULT get_BandDensityEQSPan(double* pVal)
- HRESULT put_BandDensityEQSPan(double newVal)

**Interface**  
IMarker8
**BandDensityNoiseState Property**

**Description**
Sets and reads the band noise density marker state.

**VB Syntax**

```
mkr.BandDensityNoiseState = value
```

**Variable**

- **mkr**
  - Marker (object)
- **value**
  - Enum as NAStates

Choose from:

- naOFF (or 0) - Turn band noise density marker OFF
- naON (or 1) - Turn band noise density marker ON

**Return Type**
Boolean

**Default**
aOFF

**Examples**

```vbnet
mkr.BandDensityNoiseState = 0 'Write
value = mkr.BandDensityNoiseState 'Read
```

**C++ Syntax**

```
HRESULT get_BandDensityNoiseState(tagNAStates* pState)
HRESULT put_BandDensityNoiseState(tagNAStates pState)
```

**Interface**
IMarker8
# About Spectrum Analyzer Markers

## BandDensityNPRState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the NPR density marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mkr.BandDensityNPRState = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>mkr</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Enum as NAStates) Choose from:</td>
</tr>
<tr>
<td></td>
<td>- naOFF (or 0) - Turn NPR density marker OFF</td>
</tr>
<tr>
<td></td>
<td>- naON (or 1) - Turn NPR density marker ON</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>naOFF</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mkr.BandDensityNPRState = 0</code> Write</td>
</tr>
<tr>
<td></td>
<td><code>value = mkr.BandDensityNPRState 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_BandDensityNPRState(tagNAStates* pState)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_BandDensityNPRState(tagNAStates pState)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker9</td>
</tr>
</tbody>
</table>
## BandDensityPowerBW Property

**Description**
Sets and reads the band power density bandwidth.

**VB Syntax**
```vbnet
mkr.BandDensityPowerBW = value
```

**Variable**
- **mkr** (Type) - Marker (object)
- **value** (Double) Choose a bandwidth

**Return Type**
Double

**Default**
1 MHz

**Examples**
```vbnet
mkr.BandDensityPowerBW = 1e6 'Write
value = mkr.BandDensityPowerBW 'Read
```

**C++ Syntax**
```cpp
HRESULT get_BandDensityPowerBW(double* pVal)
HRESULT put_BandDensityPowerBW(double newVal)
```

**Interface**
IMarker8
### BandDensityPowerState Property

**Description**  
Sets and reads the band power density state.

**VB Syntax**  
\[ mkr.BandDensityPowerState = value \]

**Variable**  
(Type) - Description

- \( mkr \) A Marker (object)
- \( value \) (Enum as NAs) Choose from:
  - naOFF (or 0) - Turn band power density marker OFF
  - naON (or 1) - Turn band power density marker ON

**Return Type**  
Boolean

**Default**  
naOFF

**Examples**

- \[ mkr.BandDensityPowerState = 0 \] Write
- \[ value = mkr.BandDensityPowerState \] Read

**C++ Syntax**

- HRESULT get_BandDensityPowerState(tagNAs pState)
- HRESULT put_BandDensityPowerState(tagNAs pState)

**Interface**  
IMarker8
### BandDensityToneBW Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the bandwidth of the band tone density marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mkr.BandDensityToneBW = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>mkr</code></td>
<td>A <code>Marker</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Choose a bandwidth</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1 MHz</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mkr.BandDensityToneBW = 1e6 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = mkr.BandDensityToneBW 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_BandDensityToneBW(double* pVal)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_BandDensityToneBW(double newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker8</td>
</tr>
</tbody>
</table>
# About Spectrum Analyzer Markers

## BandDensityToneSpacing Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the tone density tone spacing.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mkr.BandDensityToneSpacing = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>mkr</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Choose a spacing value.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1 MHz</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mkr.BandDensityToneSpacing = 1e6</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>value = mkr.BandDensityToneSpacing</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_BandDensityToneSpacing(double* pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_BandDensityToneSpacing(double newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker8</td>
</tr>
</tbody>
</table>
### BandDensityToneState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the band tone density state.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mkr.BandDensityToneState = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>mkr</code></td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Enum as NASTates)</em> Choose from:</td>
</tr>
<tr>
<td></td>
<td>naOFF (or 0) - Turn band tone density marker OFF</td>
</tr>
<tr>
<td></td>
<td>naON (or 1) - Turn band tone density marker ON</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>naOFF</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mkr.BandDensityToneState = 0</code> <em>Write</em></td>
</tr>
<tr>
<td></td>
<td><code>value = mkr.BandDensityToneState</code> <em>Read</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_BandDensityToneState(tagNASTates* pState)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_BandDensityToneState(tagNASTates pState)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker8</td>
</tr>
</tbody>
</table>
### BandDensityValue Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the spectral band power density.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = mkr.BandDensityValue</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>mkr</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Band power density value.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>None</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = mkr.BandDensityValue</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_BandDensityValue(double* pVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker8</td>
</tr>
<tr>
<td>Description</td>
<td>Returns the band noise level in dBm/Hz from the band noise marker.</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>VB Syntax</td>
<td><code>value = mkr.BandNoisedBmpHz</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>Type</strong> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>Variable to store the returned Y-axis value.</td>
</tr>
<tr>
<td><code>mkr</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = mkr.BandNoisedBmpHz</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_BandNoisedBmpHz(double* pVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker7</td>
</tr>
</tbody>
</table>
### BandnoiseSpan Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the frequency span of the band noise marker. This area is marked by two vertical dotted lines on the screen and the marker's y-axis value is set to the measured power value. Noise and power on the same marker share the same span.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mkr.BandnoiseSpan = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td></td>
<td><code>mkr</code> A Marker <em>(object)</em></td>
</tr>
<tr>
<td></td>
<td><code>value</code> <strong>(Double)</strong> Choose a frequency span within the frequency range of the analyzer.</td>
</tr>
<tr>
<td>Return Type</td>
<td><strong>Double</strong></td>
</tr>
<tr>
<td>Default</td>
<td><strong>1 MHz</strong></td>
</tr>
<tr>
<td>Examples</td>
<td>See example program</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_BandnoiseSpan(double* pVal)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_BandnoiseSpan(double newVal)</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker6</td>
</tr>
</tbody>
</table>
Write/Read

About Spectrum Analyzer Markers

### BandnoiseState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the state of the band noise marker. This command makes a band noise marker from a generic marker. First turn a marker ON with the <code>meas.MarkerState</code> command. Then use this command to make it a Bandnoise marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mkr.BandnoiseState = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td>mkr</td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td>value</td>
<td><em>(Enum as NAStates)</em> Choose from:</td>
</tr>
<tr>
<td></td>
<td>naOFF (or 0) - Turn band noise marker OFF</td>
</tr>
<tr>
<td></td>
<td>naON (or 1) - Turn band noise marker ON</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>naOFF</td>
</tr>
<tr>
<td>Examples</td>
<td>See example program</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_BandnoiseState(VARIANT_BOOL* bState)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_BandnoiseState(VARIANT_BOOL bState)</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker6</td>
</tr>
</tbody>
</table>
### BandPowerdBm Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the band power level from the band power marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = mkr.BandPowerdBm</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>Variable to store the returned Y-axis power value.</td>
</tr>
<tr>
<td><code>mkr</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = mkr.BandPowerdBm</code> 'Read'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_BandPowerdBm(double* pVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker7</td>
</tr>
</tbody>
</table>
# BandpowerSpan Property

**Description**
Sets and reads the frequency span of the band power marker. This area is marked by two vertical dotted lines on the screen and the marker's y-axis value is set to the measured power value. Noise and power on the same marker share the same span.

**VB Syntax**
```
mkr.BandpowerSpan = value
```

**Variable**
- **mkr** (Type) - Description
  - **mkr** A Marker (object)
  - **value** (Double) Choose a frequency span within the frequency range of the analyzer.

**Return Type**
Double

**Default**
1 MHz

**Examples**
See example program

**C++ Syntax**
```
HRESULT get_BandpowerSpan(double* pVal)

HRESULT put_BandpowerSpan(double newVal)
```

**Interface**
IMarker6
### BandpowerState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the state of the band noise marker. This command makes a band noise marker from a generic marker. First turn a marker ON with the <code>meas.MarkerState</code> command. Then use this command to make it a bandpower marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>$mkr.BandpowerState = value$</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>$mkr$</td>
<td>A Marker (object)</td>
</tr>
<tr>
<td>$value$</td>
<td>(Enum as NASTates) Choose from:</td>
</tr>
<tr>
<td></td>
<td>- naOFF (or 0) - Turn band noise marker OFF</td>
</tr>
<tr>
<td></td>
<td>- naON (or 1) - Turn band noise marker ON</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>naOFF</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>See example program</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_BandpowerState(VARIANT_BOOL* bState)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_BandpowerState(VARIANT_BOOL bState)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker6</td>
</tr>
</tbody>
</table>
### BandwidthSearch Property

**Description**
Sets the bandwidth search preference to start a bandwidth or notch search in either peak or marker mode.

**VB Syntax**

```
pref.BandwidthSearch = value
```

**Variable (Type) - Description**

- `pref` A Preferences *(object)*
- `value` *(Enum)* - Choose from:

  0 - `searchPeak` - Search starts at the maximum y-value of the full sweep of data.

  1 - `searchMarker` - Search starts at the x and y position of the active marker.

**Return Type**
Enum

**Default**
searchPeak

**Examples**

```
pref.BandwidthSearch = 1 'Write
bwsearchmode = pref.BandwidthSearch 'Read
```

**C++ Syntax**

```
HRESULT put_BandwidthSearch(enum BandwidthSearchMode preference)
HRESULT get_BandwidthTarget(enum BandwidthSearchMode* preference)
```

**Interface**
IPreferences18
# BandwidthShape Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the resolution bandwidth shape.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>sa.BandwidthShape = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>sa</code></td>
<td>A <code>SpectrumAnalyzer (object)</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Enum as NASABandwidthShape)</em> Choose from:</td>
</tr>
<tr>
<td>0</td>
<td><code>naNoWindow</code></td>
</tr>
<tr>
<td>1</td>
<td><code>naWindowFlatTop</code></td>
</tr>
<tr>
<td>2</td>
<td><code>naWindowGaussian</code></td>
</tr>
<tr>
<td>3</td>
<td><code>naWindowBlackman</code></td>
</tr>
<tr>
<td>4</td>
<td><code>naWindowKaiser</code></td>
</tr>
</tbody>
</table>

*Learn about these choices.*

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Enum</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>2 - <code>naWindowGaussian</code></td>
</tr>
</tbody>
</table>

**Examples**

```vba
da.BandwidthShape = naWindowFlatTop 'Write
data = da.BandwidthShape 'Read
```

*See an example program.*

**C++ Syntax**

```cpp
HRESULT put_BandwidthShape(tagNASABandwidthShape shape);
HRESULT get_BandwidthShape(tagNASABandwidthShape* shape);
```

**Interface**

`ISpectrumAnalyzer`
**BandwidthTarget Property**

**Description**
Sets the insertion loss value at which the bandwidth of a filter is measured (using BandwidthTracking or SearchFilterBandwidth). For example, if you want to determine the filter bandwidth 3 db below the bandpass peak value, set BandwidthTarget to -3.

**VB Syntax**

```vbnet
meas.BandwidthTarget = value
```

**Variable (Type) - Description**

- **meas** (A Measurement object)
- **value** (single) - Target value. Choose any number between -500 and 500

**Return Type**
Single

**Default**
-3

**Examples**

```vbnet
meas.BandwidthTarget = -3 'Write
```

```vbnet
fbw = meas.BandwidthTarget 'Read
```

**C++ Syntax**

```c++
HRESULT put_BandwidthTarget(float target)
HRESULT get_BandwidthTarget(float* target)
```

**Interface**
IMeasurement
**BandwidthTracking Property**

**Description**

Searches continually (every sweep) for the current BandwidthTarget (default is -3). To search the filter bandwidth for ONE SWEEP only (not continually), use meas.SearchFilterBandwidth.

This feature uses markers 1-4. To turn off these markers, either turn them off individually or DeleteAllMarkers.

The bandwidth statistics are displayed on the analyzer screen. To get the bandwidth statistics, use either GetFilterStatistics or FilterBW, FilterCF, FilterLoss, or FilterQ.

The analyzer screen will show either Bandwidth statistics OR Trace statistics; not both.

To restrict the search to a UserRange with the bandwidth search, first activate marker 1 and set the desired UserRange. Then send the SearchFilterBandwidth command. The user range used with bandwidth search only applies to marker 1 searching for the max value. The other markers may fall outside the user range.

**VB Syntax**

```
meas.BandwidthTracking = value
```

**Variable**

(Type) - Description

- **meas**
  A Measurement (object)

- **value**
  (boolean)
  - **True** - Turns bandwidth tracking ON
  - **False** - Turns bandwidth tracking OFF

**Return Type**

Boolean

**Default**

**False**

**Examples**

```
meas.BandwidthTracking = False 'Write
bwtrack = meas.BandwidthTracking 'Read
```

**C++ Syntax**

```
HRESULT put_BandwidthTracking(VARIANT_BOOL state)
HRESULT get_BandwidthTracking(VARIANT_BOOL* state)
```

**Interface**

IMeasurement
## BandwidthNarrowMax Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the maximum value for narrow DFT bandwidth. The maximum narrow DFT bandwidth setting is 11 MHz. The <code>AutoBandwidth</code> must be set to OFF to set this value manually.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>sa.dft.BandwidthNarrowMax = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>sa.dft</code></td>
<td>A <code>SpectrumAnalyzerDFT</code> (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Examples</td>
<td><code>sa.dft.BandwidthNarrowMax = 11e6</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>value = sa.dft.BandwidthNarrowMax</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT put_BandwidthNarrowMax(double maxbw);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_BandwidthNarrowMax(double* maxbw);</code></td>
</tr>
<tr>
<td>Interface</td>
<td><code>ISpectrumAnalyzerDFT</code></td>
</tr>
</tbody>
</table>
### BandwidthNarrowMin Property

**Description**
Set and read the minimum value for narrow DFT bandwidth. The minimum narrow DFT bandwidth setting is 500 kHz. The `AutoBandwidth` must be set to OFF to set this value manually.

**VB Syntax**
```vbnet
sa.dft.BandwidthNarrowMin = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sa.dft</code></td>
<td>A <code>SpectrumAnalyzerDFT (object)</code></td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Examples**
```vbnet
sa.dft.BandwidthNarrowMin = 5e5 'Write
value = sa.dft.BandwidthNarrowMin 'Read
```

**C++ Syntax**
```c++
HRESULT put_BandwidthNarrowMin(double minbw);
HRESULT get_BandwidthNarrowMin(double* minbw);
```

**Interface**
`ISpectrumAnalyzerDFT`
BandwidthWideMax Property

Description: Set and read the maximum value for wide DFT bandwidth. The maximum wide DFT bandwidth setting is 44 MHz. The AutoBandwidth must be set to OFF to set this value manually.

VB Syntax: `sa.dft.BandwidthWideMax = value`

Variable (Type) - Description
- `sa.dft`: A SpectrumAnalyzerDFT object

Return Type: Double

Examples:
- `sa.dft.BandwidthWideMax = 44e6 'Write`
- `value = sa.dft.BandwidthWideMax 'Read`

C++ Syntax:
- `HRESULT put_BandwidthWideMax(double maxbw);`
- `HRESULT get_BandwidthWideMax(double* maxbw);`

Interface: ISpectrumAnalyzerDFT
## BandwidthWideMin Property

**Description**
Set and read the minimum value for wide DFT bandwidth. The minimum wide DFT bandwidth setting is 500 kHz. The `AutoBandwidth` must be set to OFF to set this value manually.

**VB Syntax**
```
sa.dft.BandwidthWideMin = value
```

**Variable**
*sa.dft* - A `SpectrumAnalyzerDFT` (object)

**Return Type**
Double

**Examples**
```
sa.dft.BandwidthWideMin = 5e5 'Write
value = sa.dft.BandwidthWideMin 'Read
```

**C++ Syntax**
```
HRESULT put_BandwidthWideMin(double minbw);
HRESULT get_BandwidthWideMin(double* minbw);
```

**Interface**
`ISpectrumAnalyzerDFT`
BB_BalPort1Negative Property

Description
With a Balanced - Balanced topology, returns the VNA port number that is connected to the Negative side of the DUT's logical Port 1.

Use SetBBPorts Method to set the port mapping for a Balanced - Balanced topology.

VB Syntax
var = balTopology.BB_BalPort1Negative

Variable (Type) - Description
var (Long Integer) Variable to store the returned value.

balTopology A BalancedTopology (object)

Return Type Long Integer

Default Not Applicable

Examples variable = balTop.BB_BalPort1Negative 'Read

C++ Syntax HRESULT get_BB_BalPort1Negative(long *bVal)

Interface IBalancedTopology
BB_BalPort1Positive Property

Description
With a Balanced - Balanced topology, returns the VNA port number that is connected to the Positive side of the DUT's logical Port 1.

Use SetBBPorts Method to set the port mapping for a Balanced - Balanced topology.

VB Syntax
var = balTopology.BB_BalPort1Positive

Variable (Type) - Description
var (Long Integer) Variable to store the returned value.

balTopology A BalancedTopology (object)

Return Type Long Integer
Default Not Applicable

Examples
variable = balTop.BB_BalPort1Positive 'Read

C++ Syntax
HRESULT get_BB_BalPort1Positive(long *bVal)

Interface IBalancedTopology
### BB_BalPort2Negative Property

**Description**
With a Balanced - Balanced topology, returns the VNA port number that is connected to the Negative side of the DUT's logical Port 2.

Use `SetBBPorts Method` to set the port mapping for a Balanced - Balanced topology.

**VB Syntax**
```
var = balTopology.BB_BalPort2Negative
```

**Variable (Type) - Description**
- `var` (Long Integer) Variable to store the returned value.

**balTopology**
A `BalancedTopology` (object)

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**
```
variable = balTop.BB_BalPort2Negative  'Read
```

**C++ Syntax**
```
HRESULT get_BB_BalPort2Negative(long *bVal)
```

**Interface**
IBalancedTopology
**BB_BalPort2Positive Property**

**Description**
With a Balanced - Balanced topology, returns the VNA port number that is connected to the Positive side of the DUT's logical Port 2.

Use `SetBBPorts Method` to set the port mapping for a Balanced - Balanced topology.

**VB Syntax**
```vbnet
var = balTopology.BB_BalPort2Positive
```

**Variable (Type) - Description**

- `balTopology`  
  A `BalancedTopology` *(object)*

- `var`  
  (Long Integer) Variable to store the returned value.

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**
```
variable = balTop.BB_BalPort2Positive  'Read
```

**C++ Syntax**
```cpp
HRESULT get_BB_BalPort2Positive(long *bVal)
```

**Interface**
IBalancedTopology
BBalMeasurement Property

Description
Sets and returns the measurement for the Balanced - Balanced topology.

VB Syntax
balMeas.BBalMeasurement = value

Variable (Type) - Description
balMeas A BalancedMeasurement (object)

value (String) - Balanced - Balanced Measurement parameter. Not case sensitive. Choose from:

<table>
<thead>
<tr>
<th>Sdd11</th>
<th>Sdd12</th>
<th>Sdc11</th>
<th>Sdc12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sdd21</td>
<td>Sdd22</td>
<td>Sdc21</td>
<td>Sdc22</td>
</tr>
<tr>
<td>Scd11</td>
<td>Scd12</td>
<td>Scc11</td>
<td>Scc12</td>
</tr>
<tr>
<td>Scd21</td>
<td>Scd22</td>
<td>Scc21</td>
<td>Scc22</td>
</tr>
<tr>
<td>Imb1</td>
<td>Imb2</td>
<td>CMRR  -(Sdd21/Scc21)</td>
<td></td>
</tr>
</tbody>
</table>

Return Type
String

Default
Sdd11

Examples

balMeas.BBalMeasurement = "Sdd11" 'Write

variable = balMeas.BBalMeasurement 'Read

C++ Syntax
HRESULT get_BBalMeasurement(BSTR *pVal)

HRESULT put_BBalMeasurement(BSTR newVal)

Interface
IBalancedMeasurement
## BeforeSweepCmd Property

**Description**
Sets and returns the *Before Sweep command* for an external DC Source and an external DC Meter.

**VB Syntax**
```vbnet
extDC.BeforeSweepCmd = value
```

**Variable**
<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>extDC</code></td>
<td>A <code>ExternalDCDevice</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(String) The SCPI command to be sent at the beginning of a sweep.</td>
</tr>
</tbody>
</table>

**Return Type**
String

**Default**
" " Empty String

**Examples**
```vbnet
extDC.BeforeSweepCmd = "OUTP ON" 'Write
value = extDC.BeforeSweepCmd 'Read
```

**C++ Syntax**
```cpp
HRESULT get_BeforeSweepCmd( BSTR* cmd);
HRESULT put_BeforeSweepCmd( BSTR cmd);
```

**Interface**
`IExternalDCDevice3`
### BeginResponse Property

**Description**
When constructing a limit line, specifies the amplitude value of the start of a limit segment.

**VB Syntax**
`limtseg.BeginResponse = value`

**Variable**
- **Type** - Description
  - `limtseg` - A LimitSegment *(object)*
  - `value` - *(double)* - Amplitude value. No units

**Return Type**
Double

**Default**
0

**Examples**
```
Set limtseg = meas.LimitTest(1)
limtseg.BeginResponse = 10 'Write

BegResp = limtseg.BeginResponse 'Read
```

**C++ Syntax**
```
HRESULT get_BeginResponse(double *pVal)
HRESULT put_BeginResponse(double newVal)
```

**Interface**
ILimitSegment
**BeginStimulus Property**

**Description**
When constructing a limit line, specifies the beginning X-axis value.

**VB Syntax**

```vbnet
limtseg.BeginStimulus = value
```

**Variable**

- **Type** - Description
  - `limtseg` A LimitSegment (object)
  - `value` (double) - Stimulus value. No units

**Return Type**
Double

**Default**
0

**Examples**

Set `limtseg = meas.LimitTest(1)`
- `limtseg.Type = naLimitSegmentType_Maximum`
- `limtseg.BeginStimulus = 3e9`
- `limtseg.EndStimulus = 4e9`
- `limtseg.BeginResponse = 10`
- `limtseg.EndResponse = 10`

```vbnet
BegStim = limtseg.BeginStimulus 'Read
```

**C++ Syntax**

```cpp
HRESULT get_BeginStimulus(double *pVal)
HRESULT put_BeginStimulus(double newVal)
```

**Interface**
ILimitSegment
### BinaryFileEnabled Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables/disables binary file (*.bin) output.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>sa.dft.BinaryFileEnabled = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>A SpectrumAnalyzerDFT (object)</td>
</tr>
<tr>
<td><code>sa.dft</code></td>
<td></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean) Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - OFF - Disable binary file output.</td>
</tr>
<tr>
<td></td>
<td>1 - ON - Enable binary file output.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>OFF</td>
</tr>
<tr>
<td>Examples</td>
<td><code>sa.dft.BinaryFileEnabled = ON</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>value = sa.dft.BinaryFileEnabled</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_BinaryFileEnabled(VARIANT_BOOL* enable)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_BinaryFileEnabled(VARIANT_BOOL enable)</td>
</tr>
<tr>
<td>Interface</td>
<td>ISpectrumAnalyzerDFT</td>
</tr>
</tbody>
</table>
### BroadbandTuningSpan Property

**Description**
Sets and returns the frequency span for the broadband tuning sweep.

**VB Syntax**
```
obj.BroadbandTuningSpan = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>An EmbeddedLO (object) or A ConverterEmbeddedLO (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Double) Broadband frequency span in Hz.</td>
</tr>
</tbody>
</table>

**Return Type**
(Double)

**Default**
3 MHz

**Examples**
```
embedLO.BroadbandTuningSpan = 1E6 'write
value = embedLO.BroadbandTuningSpan 'read
```

**C++ Syntax**
```
HRESULT get_BroadbandTuningSpan(double* span);
HRESULT put_BroadbandTuningSpan(double span);
```

**Interface**
IEmbededLO
### BS_BalPortNegative Property

**Description**  
With a Balanced - Single-ended topology, returns the VNA port number that is connected to the Negative side of the DUT's balanced port.

Use **SetSBPorts Method** to set the port mapping for a Balanced - Single-ended topology.

**VB Syntax**  
```vbnet
var = balTopology.BS_BalPortNegative
```

**Variable**  
(Type) - Description

- `var` (Long Integer) Variable to store the returned value.

- `balTopology` A **BalancedTopology (object)**

**Return Type**  
Long Integer

**Default**  
Not Applicable

**Examples**  
`variable = balTop.BS_BalPortNegative 'Read`

**C++ Syntax**  
```cpp
HRESULT get_BS_BalPortNegative(long *bVal)
```

**Interface**  
IBalancedTopology2
## About Balanced Measurements

### BS_BalPortPositive Property

**Description**
With a Balanced - Single-ended topology, returns the VNA port number that is connected to the Positive side of the DUT's balanced port.

Use **SetSBPorts Method** to set the port mapping for a Balanced - Single-ended topology.

**VB Syntax**

```vb
var = balTopology.BS_BalPortPositive
```

**Variable** *(Type)* - Description

- `var` *(Long Integer)* Variable to store the returned value.

- `balTopology` A **BalancedTopology** *(object)*

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**

```vb
variable = balTop.BS_BalPortPositive  'Read
```

**C++ Syntax**

```cpp
HRESULT get_BS_BalPortPositive(long *bVal)
```

**Interface**
IBalancedTopology2
### BS_SEPort Property

<table>
<thead>
<tr>
<th>Description</th>
<th>With a Balanced - Single-ended topology, returns the VNA port number that is connected to the Single-ended port. Use <code>SetSBPorts Method</code> to set the port mapping for a Balanced - Single-ended topology.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>var = balTopology.BS_SEPort</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td></td>
<td><code>var</code> (Long Integer) Variable to store the returned value.</td>
</tr>
<tr>
<td><strong>balTopology</strong></td>
<td>A <code>BalancedTopology</code> <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>variable = balTop.BS_SEPort</code> <em>Read</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_BS_SEPort(long *bVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IBalancedTopology2</td>
</tr>
</tbody>
</table>
### BSS_BalPortNegative Property

**Description**
With a Balanced - Single-ended - Single-ended topology, returns the VNA port number that is connected to the Negative side of the DUT's balanced port.

Use `SetBSSPorts Method` to set the port mapping for a Balanced - Single-ended - Single-ended topology.

**VB Syntax**
```
var = balTopology.BSS_BalPortNegative
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>var</code></td>
<td>(Long Integer) Variable to store the returned value.</td>
</tr>
</tbody>
</table>

**balTopology**
ABalancedTopology (object)

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**
```
variable = balTop.BSS_BalPortNegative 'Read
```

**C++ Syntax**
```
HRESULT get_BSS_BalPortNegative(long *pbalanceNegative)
```

**Interface**
IBalancedTopology4
# BSS_BalPortPositive Property

**Description**
With a Balanced - Single-ended - Single-ended topology, returns the VNA port number that is connected to the Positive side of the DUT's balanced port.

Use the [SetBSSPorts Method](#) to set the port mapping for a Balanced - Single-ended - Single-ended topology.

**VB Syntax**
```vbnet
var = balTopology.BSS_BalPortPositive
```

**Variable**
- **Type** - Description
- `var` (Long Integer) Variable to store the returned value.

**balTopology**
A BalancedTopology (object)

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**
```vbnet
variable = balTop.BSS_BalPortPositive  'Read
```

**C++ Syntax**
```csharp
HRESULT get_BSS_BalPortPositive(long *pbalancePositive)
```

**Interface**
IBalancedTopology4
### BSS_SEPort1 Property

**Description**  
With a Balanced - Single-ended - Single-ended topology, returns the VNA port number that is connected to the Single-ended port 1.

Use [SetBSSPorts Method](#) to set the port mapping for a Balanced - Single-ended - Single-ended topology.

**VB Syntax**  
```vbnet
var = balTopology.BSS_SEPort1
```

**Variable**  
*(Type)* - Description

<table>
<thead>
<tr>
<th>var</th>
<th>(Long Integer) Variable to store the returned value.</th>
</tr>
</thead>
</table>

**balTopology**  
A BalancedTopology *(object)*

**Return Type**  
Long Integer

**Default**  
Not Applicable

**Examples**  
```vbnet
variable = balTop.BSS_SEPort1 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_BSS_SEPort1(long *pSEPort1)
```

**Interface**  
IBalancedTopology4
### BSS_SEPort2 Property

**Description**  
With a Balanced - Single-ended - Single-ended topology, returns the VNA port number that is connected to the Single-ended port 2.

Use **SetBSSPorts Method** to set the port mapping for a Balanced - Single-ended - Single-ended topology.

**VB Syntax**  
\`\`var = balTopology.BSS_SEPort2\`\`

**Variable (Type) - Description**  
- **var** (Long Integer) Variable to store the returned value.

**balTopology**  
A **BalancedTopology** *(object)*

**Return Type**  
Long Integer

**Default**  
Not Applicable

**Examples**  
\`\`variable = balTop.BSS_SEPort2 \`\`

**C++ Syntax**  
\`\`HRESULT get_BSS_SEPort2(long *pSEPort2)\`\`

**Interface**  
IBalancedTopology4
**BSSMeasurement Property**

**Description**

**VB Syntax**
`balMeas.BSSMeasurement = value`

**Variable**
(Type) - Description

- `balMeas`: A `BalancedMeasurement` (object)
- `value`: (String) - Balanced - Single-Ended - Single-Ended Measurement parameter. Not case sensitive. Choose from:

<table>
<thead>
<tr>
<th>Sdd11</th>
<th>Sdc11</th>
<th>Sds12</th>
<th>Sds13</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scd11</td>
<td>Scc11</td>
<td>Scs12</td>
<td>Scs13</td>
</tr>
<tr>
<td>Ssd21</td>
<td>Ssc21</td>
<td>Sss22</td>
<td>Sss23</td>
</tr>
<tr>
<td>Ssd31</td>
<td>Ssc31</td>
<td>Sss32</td>
<td>Sss33</td>
</tr>
<tr>
<td>Imbal1</td>
<td>Imbal2</td>
<td>Sds12/Scs12</td>
<td>Sds13/Scs13</td>
</tr>
</tbody>
</table>

**Return Type**
String

**Default**
Sdd11

**Examples**

```
balMeas.BSSMeasurement = "Sdd11" 'Write

variable = balMeas.BSSMeasurement 'Read
```

**C++ Syntax**

```c++
HRESULT get_BSSMeasurement(BSTR *pVal)

HRESULT put_BSSMeasurement(BSTR newVal)
```

**Interface**
`IBalancedMeasurement4`

---

1412
BucketNumber Property

Description
Sets or returns the bucket number (data point) for the active marker. When the markers are interpolated (non-discrete), the returned value is the nearest marker bucket position.

VB Syntax
mark.BucketNumber = value

Variable (Type) - Description
mark A Marker (object)
value (long integer) - Data point. Choose any number between 0 and the measurement's number of data points - 1. For example, with Number of points = 201, choose between 0 and 200

Return Type
Long Integer

Default
The first marker is set to the middle of the span. Subsequent markers are set to the bucket number of the previously active marker.

Examples
mark.BucketNumber = 100 'moves the active marker to data point 100 - Write
pointNumber = mark.BucketNumber 'returns the data point number of the marker object. When the markers are interpolated (non-discrete), the returned value is the nearest marker bucket position.

C++ Syntax
HRESULT get_BucketNumber(long *pVal)
HRESULT put_BucketNumber(long newVal)

Interface IMarker
### C0 Property

**Description**
Sets and Returns the C0 (C-zero) value (the first capacitance value) for the calibration standard.

To set the other capacitance values, use C1, C2, C3

**VB Syntax**
```
calstd.C0 = value
```

**Variable**
- **calstd** (object) - A CalStandard (object). Use calKit.GetCalStandard to get a handle to the standard.
- **value** (single) - Value for C0 in femtofarads (1E-15)

**Return Type**
Single

**Default**
Not Applicable

**Examples**
```
calstd.C0 = 15 'Write the value of C0 to 15 femtofarads
```
```
cap0 = calstd.C0 'Read the value of C0
```

**C++ Syntax**
- `HRESULT get_C0(float *pVal)`
- `HRESULT put_C0(float newVal)`

**Interface**
ICalStandard
## C1 Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Returns the C1 value (the second capacitance value) for the calibration standard. To set the other capacitance values, use C0, C2, C3.</th>
</tr>
</thead>
</table>

### VB Syntax

```vbnet
calstd.C1 = value
```

### Variable (Type) - Description

- **value**: (single) - Value for C1.

### Return Type

Single

### Default

Not Applicable

### Examples

- `calstd.C1 = 15 'Write the value of C1.
- `cap1 = calstd.C1 'Read the value of C1.

### C++ Syntax

```cpp
HRESULT get_C1(float *pVal)
HRESULT put_C1(float newVal)
```

### Interface

ICalStandard
# C2 Property

**Description**  
Sets and Returns the C2 value (the third capacitance value) for the calibration standard.

To set the other capacitance values, use C0, C1, C3.

**VB Syntax**  
`calstd.C2 = value`

**Variable**  

*value* - Value for C2.

**Return Type**  
Single

**Default**  
Not Applicable

**Examples**  
```
calstd.C2 = 15 'Write the value of C2.
cap2 = calstd.C2 'Read the value of C2
```

**C++ Syntax**  
```
HRESULT get_C2(float *pVal)
HRESULT put_C2(float newVal)
```

**Interface**  
ICalStandard
### C3 Property

**Description**  Sets and Returns the C3 value (the fourth capacitance value) for the calibration standard.

To set the other capacitance values, use \( C0, C1, C2 \)

**VB Syntax**  
\`
calstd.C3 = value
``

**Variable**  
- `value`  (single) - Value for C3.

**Return Type**  Single

**Default**  Not Applicable

**Examples**  
- `calstd.C3 = 15`  'Write the value of C3.'
- `cap3 = calstd.C3`  'Read the value of C3'

**C++ Syntax**  
- `HRESULT get_C3(float *pVal)`
- `HRESULT put_C3(float newVal)`

**Interface**  ICalStandard
## Cable Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the name of the cable that is assigned to the port object for Dynamic Uncertainty.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>port.Cable = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>port</code></td>
<td>A <em>Port</em> <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(String)</em> - Cable name.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>port.Cable = &quot;MyNewCable&quot;</code> <em>Write the value</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_Cable([out,retval] BSTR *pCableName);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_Cable([in] BSTR cableName);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPort</td>
</tr>
</tbody>
</table>
CableRepeatabilityEnabled Property

**Description**
Sets and returns the ON/OFF state of allowing cable repeatability data to contribute to the uncertainty of a calibration performed using Dynamic Uncertainty. Repeatability data must also be present for the ports at the time the calibration is performed.

**VB Syntax**
```
uncertMan.CableRepeatabilityEnabled = value
```

**Variable**
(***Type*** - Description

- **uncertMan** An **UncertaintyManager** Object
- **value** (Boolean) Enable state. Choose from:
  - **True** - Cable repeatability uncertainty ON.
  - **False** - Cable repeatability uncertainty OFF.

**Return Type**
Boolean

**Default**
True

**Examples**
```
uncertMan.CableRepeatabilityEnabled = True
```

See example program

**C++ Syntax**
```
HRESULT get_CableRepeatabilityEnabled([out,retval] VARIANT_BOOL* pState);

HRESULT put_CableRepeatabilityEnabled([in] VARIANT_BOOL state);
```

**Interface**
IUncertaintyManager
### CableRepeatabilityUncertainty Property

**Description**
Sets and returns whether the cable/connection repeatability contribution is currently included in the uncertainty values for the measurement trace.

**VB Syntax**

```vbnet
uncert.CableRepeatabilityUncertainty = value
```

**Variable** *(Type) - Description*

- ** uncert** An Uncertainty *(object)*
- ** value** (Boolean) Choose from:
  - **False** - Cable repeatability is NOT included.
  - **True** - Cable repeatability IS included.

**Return Type**
Boolean

**Default**
True

**Examples**

```vbnet
uncert.CableRepeatabilityUncertainty = True
```

**C++ Syntax**

```cpp
HRESULT get_CableRepeatabilityUncertainty(VARIANT_BOOL *pState);
HRESULT put_CableRepeatabilityUncertainty(VARIANT_BOOL state);
```

**Interface**
IUncertainty
## CalFactor Property

**Description**  
Sets or returns the cal factor value associated with a power sensor cal factor segment.

**VB Syntax**  
`calFactSeg.CalFactor = value`

**Variable**  
(Type) - Description  
`calFactSeg`  
A PowerSensorCalFactorSegment (Object) or  
A PowerSensorCalFactorSegmentPMAR (Object)

`value`  
(double) – Cal factor in percent. Choose any value between 1 and 150

**Return Type**  
Double

**Default**  
0

**Examples**  
`calFactSeg.CalFactor = 98 'Write`  
`factor = calFactSeg.CalFactor 'Read`

**C++ Syntax**  
HRESULT put_CalFactor(Double newVal);  
HRESULT get_CalFactor(Double *pVal);

**Interface**  
IPowerSensorCalFactorSegment
About Performing a Calibration

CalibrationType Property  **Superseded**

**Description**

**Note:** This command has been replaced by `CalibrationTypeID_property`, which provides selection of Calibration Type by string.

Specifies the type of calibration to perform or apply to the active S-Parameter measurement. This command determine the ports involved in the CalType by the ports being used by the active measurement.

For example:

- If the measurement is an S23, it uses ports 2 and 3.
- If the measurement is an S22 it will use the legacy load port to figure out which two ports form the caltype. The legacy load port is set using `CreateMeasurement`.
- If `naCalType_ThreePort_SOLT` is specified on a 4-port VNA, an `E_NA_DEPRECATED_COMMAND` error is returned. There is no way to determine the intended three ports.
- If `naCalType_FourPort_SOLT` is specified on a 4-port VNA, it is obvious that the ports involved are ports 1,2,3, and 4.

**Note:** For FCA measurements, use `CalibrationName` and `CalibrationTypeID`.

**VB Syntax**

```vb
meas.CalibrationType = type
```

**Variable**

- **(Type)** - Description
  - `meas` A Measurement (object)
  - `type` (enum NACalType) - Calibration type. Choose from:
    - 0 - `naCalType_Response_Open`
    - 1 - `naCalType_Response_Short`
    - 2 - `naCalType_Response_Thru`
    - 3 - `naCalType_Response_Thru_And_Isol`
    - 4 - `naCalType_OnePort`
    - 5 - `naCalType_TwoPort_SOLT`
    - 6 - `naCalType_TwoPort_TRL`
```
7 - naCalType_None
8 - naCalType_ThreePort_SOLT
9 - Custom
10 - naCalType_FourPort_SOLT

Return Type NACalType
Default naCalType_None

Examples
meas.CalibrationType = naCalType_Response_Open 'Write
meascal = meas.CalibrationType 'Read

C++ Syntax
HRESULT put_CalibrationType (tagNACalType CalType)
HRESULT get_CalibrationType (tagNACalType* pCalType)

Interface IMeasurement
## CalibrationName Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the name of the current Cal Type.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>value = meas.CalibrationName</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(string) - Variable to store the returned value.</td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>ct = meas.CalibrationName</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_CalibrationName( BSTR* CalibrationName);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
CalibrationPort Property - Obsolete

Description: Beginning with Rev 6.0, this command is no longer necessary. Because of improved calibration techniques, Both is always selected although a power meter measurement is performed only on port 1.

Specifies which SMC port to calibrate.

VB Syntax: 

```vbnet
SMC.CalibrationPort = value
```

Variable (Type) - Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMC</td>
<td>SMCTYPE (object)</td>
<td></td>
</tr>
<tr>
<td>value</td>
<td>(String)</td>
<td>Port number to be calibrated. Choose from:</td>
</tr>
</tbody>
</table>

- 1 - SMC forward
- 2 - SMC reverse
- Both

Return Type: String

Default: 1

Examples: 

```vbnet
value = SMC.CalibrationPort = "Both"
```

C++ Syntax: 

```cpp
HRESULT put_CalibrationPort(BSTR port);
HRESULT get_CalibrationPort(BSTR *port);
```

Interface: 

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMCType</td>
<td></td>
</tr>
<tr>
<td>VMCTYPE</td>
<td></td>
</tr>
</tbody>
</table>
CalibrationPorts Property

Description
For each channel to be calibrated, sets and returns the ports to be calibrated. The port numbers need to be specified only for standard channels. Apps channels have designated input/output/LO ports.

Select any of the native VNA ports.

VB Syntax

```vbnet
calAll.CalibrationPorts (chan) = ports
```

Variable

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>calAll</code></td>
<td>A CalibrateAllChannels (object)</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>Channel to be calibrated with a Calibrate All Channels calibration.</td>
</tr>
<tr>
<td><code>port</code></td>
<td>(Variant Array) Ports to be calibrated for the specified channel.</td>
</tr>
</tbody>
</table>

Return Type
Variant Array

Default
Standard channels - ports 1 and 2

Apps channels - the designated input and output ports.

Examples

```vbnet
calAll.CalibrationPorts(2) = 0 'Ports to calibrate for channel 2
value = calAll.CalibrationPorts(2) 'returns the ports for channel 2
```

C++ Syntax

```cpp
HRESULT get_CalibrationPorts (long channel, Variant ports);
HRESULT put_CalibrationPorts (long channel, Variant* ports);
```

Interface
ICalibrateAllChannels
### CalibrationTypeID Property

**Description**  
**Note:** This command replaces Calibration Type Property.

Sets or returns the current cal type for the measurement using a Cal Type Name.

This command is used to set the Cal Type after recalling a Cal Set. Learn more

You can also use the CLSID or GUID associated with the Cal Type.

**VB Syntax**  
```vbnet
meas.CalibrationTypeID = id
```

**Variable**  
- **meas** (Type) - A Measurement (object)
- **id** (String) - Cal type. Case sensitive. Use one of the following:

#### For Full Calibrations:

This command does not distinguish between TRL and SOLT. The same number of error terms is applied for both Cal Types.

"Full n Port(x,y,z,...)"

where

- `n =` the number of ports to calibrate
- `x,y,z =` the port numbers to calibrate

For example:

"Full 7 Port(2,3,4,5,6,7,8)"

#### For Response Calibrations:

"Response(param)" OR

"ResponseAndIsolation(param)"

Where `param =`

- S-parameter. For example""Response(S21)"

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meas</td>
<td>Measurement</td>
<td>A Measurement</td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>Cal type. Case sensitive.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Note:</th>
</tr>
</thead>
<tbody>
<tr>
<td>This command replaces Calibration Type Property.</td>
</tr>
</tbody>
</table>

| Sets or returns the current cal type for the measurement using a Cal Type Name. |

| This command is used to set the Cal Type after recalling a Cal Set. Learn more |

| You can also use the CLSID or GUID associated with the Cal Type. |
- "ResponseAndIsolation(A/R)"

- Single or ratioed receivers using either logical receiver notation or physical receiver notation. For example:
  - "Response (A)"
  - "ResponseAndIsolation(a3/b4)"

**For FCA Calibrations:**

- "SMC_2P" (Response + Input + Output) All four sweeps required. Most accurate.
- "SMCRsp+IN" No Output match. All four sweeps required.
- "SMCRsp+OUT" No Output match. All four sweeps required.
- "SMCRsp" No Input or Output match. Saves two sweeps.

For VMC, multiple Cal types are not available.

**For Gain Compression Cal**

where r = receive port; s = source port

- "GCA 2P (r,s)" - full 2-port cal
- "GCA Enh Resp (r,s)" - Enhanced Response Cal

<table>
<thead>
<tr>
<th>Return Type</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td></td>
</tr>
</tbody>
</table>
| Dim pna
Dim m
Set pna = CreateObject("AgilentPNA835x.Application")
Set m = pna.ActiveMeasurement
m.CalibrationTypeID = "Scalar Mixer Cal"
m.ErrorCorrection = True
MsgBox m.CalibrationName |

<table>
<thead>
<tr>
<th>C++ Syntax</th>
<th>HRESULT get_CalibrationTypeID( BSTR* CalibrationTypeID );</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HRESULT put_CalibrationTypeID( BSTR CalibrationTypeID );</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
Write/Read

About SweptIMDCal

**CalibrationFrequencies Property**

**Description**
Sets and returns the whether to perform the source power cal at the center frequencies midway between the main tones, or at all main tone frequencies.

**VB Syntax**

```
imd.CalibrationFrequencies = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>imd</code></td>
<td>A SweptIMDCal (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Enum as NAIMDCalibrationFrequencies) Choose from:</td>
</tr>
</tbody>
</table>

0 - **naIMDCenterFrequencies** - Perform source power calibration at only the center frequencies midway between the main tones.

1 - **naIMDALLFrequencies** - Perform source power calibration at all main tone frequencies.

**Return Type**
Enum

**Default**
0 - naIMDCenterFrequencies

**Examples**

```
imd.CalibrationFrequencies = naIMDALLFrequencies 'Write

calFreq = imd.CalibrationFrequencies 'Read
```

**C++ Syntax**

```
HRESULT get_CalibrationFrequencies(tagNAIMDCalibrationFrequencies * Val)
HRESULT put_CalibrationFrequencies(tagNAIMDCalibrationFrequencies newVal)
```

**Interface**
ISweptIMD
CalKitType Property

**Description**
Sets and returns a calibration kit type to be used for **UNGUIDED** calibration and for cal kit modification. To get a handle to this kit, use `app.ActiveCalKit`.

Although an unlimited number of cal kits can be imported into the VNA, ONLY mechanical cal kits #1 through #95 can be accessed.

There is also a CalKitType property for use during a Guided, SMC, and VMC Calibration.

**VB Syntax**

```vbnet
object.CalKitType = value
```

**Variable**

- **object** - Description
- `calkit` (object) or `Application` (object)

**Note:** `app.CalKitType` and `calkit.calKitType` perform exactly the same function.

**value**

- **(enum naCalKit)** - Calibration Kit type. Choose from:
  1 - naCalKit_User1
  2 - naCalKit_User2
  3 - naCalKit_User3
  4 - naCalKit_User4
  ..
  ..
  94 - naCalKit_User94
  95 - naCalKit_User95

These enumerated values correspond with the calibration kit ID on the Advanced Cal Kit Modify dialog box.

To change the cal kit name, use Name property.

**Note:** Always check the list of available cal kits using CalKitTypes to ensure that the correct cal kit is selected.

**Return Type**

NACalKit
<table>
<thead>
<tr>
<th>Default</th>
<th>Not Applicable</th>
</tr>
</thead>
</table>
| Examples  | `calkit.CalKitType = naCalKit_User27`  
            `kiType = app.CalKitType` |
| C++ Syntax| `HRESULT get_CalKitType(tagNACalKit *pVal);`  
            `HRESULT put_CalKitType(tagNACalKit newVal);` |
| Interface | `IApplication`  
            `ICalKit` |
CalKitType Property

Description
Sets and returns the ECal or mechanical cal kit for the specified port number to be used during the calibration.

There is also a CalKitType Property for use during an Unguided Cal.

Note: Sliding loads are not fully supported from the GuidedCalibration object. The Measure button must be pressed manually on the VNA.

Note: This command replaces Do1PortEcal Property and Do2PortEcal Property for SMC and VMC Calibrations.

VB Syntax
object.CalKitType (port) = value

Variable (Type) - Description
object Any of the following:
GuidedCalibration (object)
SMCType (object)
VMCType (object)

port (Long) Port number to which the cal kit will be assigned.
For Guided Cals and SMC, select port number.
For VMC calibrations:
   • 1 - Mixer Input.
   • Any unused port can be used for the mixer output.
   • Output port of MUT +1 - Output port of the calibration mixer. Generally this is port 3.

value (string) - Calibration Kit type. Case-sensitive.
Use GetCompatibleCalKits to return a list of valid Cal Kits.

Return Type String
Default Not Applicable

Examples 'Note: All of the following specify port 1 only
'Mechanical Cal Kit
guidedCal.CalKitType(1) = "85052C"

' Standard ECal modules

guidedCal.CalKitType(1) = "N4691-60004 ECal"

' Non-factory ECal characterizations are specified as follows:

guidedCal.CalKitType(1) = "N4691-60004 User 1 ECal"

' When two or more ECal modules with the same model number are
' connected, also specify the serial number as follows:

guidedCal.CalKitType(1) = "N4691-60004 ECal 01234"

' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:

'guidedCal.CalKitType(1) = "N4691-60004 MyDskChar ECal 01234"

' Turn on auto orientation for the ECal (default behavior).

value = smc.CalKitType(1)  'Read

C++ Syntax

HRESULT get_CalKitType( long port, BSTR *calkit)

HRESULT put_CalKitType( long port, BSTR calkit)

Interface

IGuidedCalibration

SMCType

VMCType
## CalKitType Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the name of the Cal Kit to use for unguided cal.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>cal.CalKitType (port) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>cal</code></td>
<td>Calibrator (object)</td>
</tr>
<tr>
<td><code>port</code></td>
<td>(Long) Currently unused</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(String) Cal Kit name enclosed in quotes. Use <code>CalKitTypes</code> to read a list of all available Cal Kits in the VNA.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Last kit selected</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>cal.CalKitType(1) = &quot;85052B&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_CalKitType(long port, BSTR calKit);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_CalKitType(long port, BSTR* pCalKit);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalibrator10</td>
</tr>
</tbody>
</table>
**CalKitTypes Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the names of the available mechanical cal kits in your VNA that can be used for unguided calibrations.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>value = cal.CalKitTypes (port)</td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td>cal</td>
<td>Calibrator (object)</td>
</tr>
<tr>
<td>port</td>
<td>Port number</td>
</tr>
<tr>
<td>value</td>
<td>(Variant) Variable to store the returned cal kit types.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Variant</td>
</tr>
<tr>
<td>Examples</td>
<td>value = cal.CalKitTypes(4)</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_CalKitTypes(VARIANT* calkits);</td>
</tr>
<tr>
<td>Interface</td>
<td>ICalibrator10</td>
</tr>
</tbody>
</table>
### CalMethod Property

**Description**
Sets and returns the method for performing calibration on a noise channel.

**VB Syntax**
`noise.CalMethod = value`

**Variable (Type) - Description**
- `noise` A NoiseCal *(object)*
- `value` *(string)* Cal Method. Choose from:
  - "VectorFull" or "Vector"
  - "SParameter" (Not available for NFX measurements)
  - "ScalarFull" or "Scalar"

**Return Type**
String

**Default**
"VectorFull"

**Examples**
- `noise.CalMethod = "VectorFull"` *Write*
- `calMethod = noise.CalMethod` *Read*

**C++ Syntax**
- `HRESULT get_CalMethod(BSTR* pValue)`
- `HRESULT put_CalMethod(BSTR pNewValue)`

**Interface**
INoiseCal
### CalMethod Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the method by which the match-correction portion of an IMD calibration is performed. <a href="#">Learn more.</a></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>imd.CalMethod = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>imd</code></td>
<td>A SweptIMDCal <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Enum as NAIMDCalMethod) Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>0 - naIMDMatchCorrectedResponse</strong> - Performs a full 2-port cal for full match-correction.</td>
</tr>
<tr>
<td></td>
<td><strong>1 - naIMDResponseOnly</strong> - Performs only a response cal instead of a full 2-port cal.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td><strong>0 - naIMDMatchCorrectedResponse</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>imd.CalMethod = naIMDMatchCorrectedResponse</code> <em>(Write)</em></td>
</tr>
<tr>
<td></td>
<td><code>calMeth = imd.CalMethod</code> <em>(Read)</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_CalMethod(tagNAIMDCalMethod * Val)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_CalMethod(tagNAIMDCalMethod newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISweptIMD</td>
</tr>
</tbody>
</table>
### CalPower Property Superseded

**Description**  
This command is replaced with `chan.TestPortPower`.  
Returns the RF power level for the channel.

**VB Syntax**  
```vbnet
value = powerCalibrator.CalPower(chan, sourcePort)
```

**Variable**  
<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>double</td>
<td>Variable to store the returned Cal power value in dBm.</td>
</tr>
<tr>
<td>powerCalibrator</td>
<td>object</td>
<td>A SourcePowerCalibrator object</td>
</tr>
<tr>
<td>chan</td>
<td>long integer</td>
<td>Channel number of the VNA.</td>
</tr>
<tr>
<td>sourcePort</td>
<td>long integer</td>
<td>Source port number.</td>
</tr>
</tbody>
</table>

Use `GetPortNumber` to return the port number of a source that only has a string name, such as an External Source.

**Return Type**  
None

**Default**  
Not applicable

**Examples**  
```
Set powerCalibrator = pna.SourcePowerCalibrator
power = powerCalibrator.CalPower(1,2) 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_CalPower(long channel, long sourcePort, double *pVal);
```

**Interface**  
ISourcePowerCalibrator
### CalSet Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the Cal Set name into which the calibration will be saved. The phase reference cal can NOT be saved to a cal register. The Cal Set is saved by calling <code>GenerateErrorTerms</code> at the conclusion of the Guided Cal.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>phasRef.CalSet = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>phasRef</code> - Description</td>
</tr>
<tr>
<td><strong>(Type)</strong></td>
<td>A <code>PhaseReferenceCalibration</code> Object</td>
</tr>
<tr>
<td><strong>value</strong></td>
<td>(String) Cal Set name.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>phase.CalSet = &quot;PhaseRefCal&quot;</code></td>
</tr>
<tr>
<td><strong>See example program</strong></td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_CalSet(BSTR CalSet* pVals);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_CalSet(BSTR CalSet newVals);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPhaseReferenceCalibration</td>
</tr>
</tbody>
</table>
Center Property

Description
Sets or returns the Center time of either Gating or Time Domain transform windows

VB Syntax
object.Center = value

Variable
(Type) - Description

object (object) As Gating
or
(object) As Transform

value (double) - Center time in seconds. Choose any number between:
 ± (points-1) / frequency span

Return Type
Double

Default
0

Examples
trans.Center = 4.5e-9 'sets the Center time of a transform window
-gate.Center = 4.5e-9 'sets the Center time of a gating window
-cnt = trans.Center 'Read

C++ Syntax
HRESULT get_Center(double *pVal)
HRESULT put_Center(double newVal)

Interface
ITransform
IGating
### Center Property

**Description**
Returns the stimulus value of the center data point for the measurement. This function does NOT work for segment sweep measurements. To understand how this property is useful, see IMeasurement2 Interface.

**VB Syntax**

```
value = meas.Center
```

**Variable**

- `value` *(Double)* - Variable to store the returned value.
- `meas` A Measurement *(object)*

**Return Type**
Double

**Default**
Not Applicable

**Examples**

```
Print meas.Center 'prints the center data point
```

**C++ Syntax**

```
HRESULT get_Center(double * Val);
```

**Interface**
IMeasurement2
### CenterFrequency Property

**Description**
Sets or returns the center frequency of the channel
or
Sets or returns the center frequency of the segment.

See the Measurement2 Interface to learn how this method differs from meas.Center.

**VB Syntax**
```
object.CenterFrequency = value
```

**Variable (Type) - Description**
- **object**
  - A Channel (object)

  or

  - A Segment (object)

- **value**
  - (double) - Center frequency in Hertz. Choose any number between the minimum and maximum frequencies of the analyzer.

**Return Type**
Double

**Default**
Center of the frequency range

**Examples**
- `chan.CenterFrequency = 4.5e9` 'sets the center frequency of a linear sweep for the channel object -Write
- `centfreq = chan.CenterFrequency` 'Read

**C++ Syntax**
```
HRESULT get_CenterFrequency(double *pVal)
HRESULT put_CenterFrequency(double newVal)
```

**Interface**
IChannel
ISegment
## CenterFrequencyStepSize Property

**Description**
Sets and reads the center frequency step size of the analyzer. This command sets the manual step size (only valid when CenterFrequencyStepSizeMode is FALSE).

**VB Syntax**
```
chan.CenterFrequencyStepSize = value
```

**Variable**

- `chan` *(Type)* - A `Channel` object
- `value` *(Double)* - Choose a value (in Hz.) below the stop frequency of the analyzer.

**Return Type**
Double

**Default**
Default is 40 MHz. When CenterFrequencyStepSizeMode is TRUE, this value is ignored.

**Examples**
```
chan.CenterFrequencyStepSize = 1e5 'Write
value = chan.CenterFrequencyStepSize 'Read
```

**C++ Syntax**
```
HRESULT get_CenterFrequencyStepSize(Double* pVal);
HRESULT put_CenterFrequencyStepSize(Double pVal);
```

**Interface**
IChannel25
CenterFrequencyStepSizeMode Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads how the center frequency step size is determined. When TRUE, center steps by 5% of span. When FALSE, center steps by STEP:SIZE value.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>chan.CenterFrequencyStepSizeMode = value</td>
</tr>
<tr>
<td>Variable</td>
<td>(object) - A Channel object</td>
</tr>
<tr>
<td>value</td>
<td>(Enum as NAModes)</td>
</tr>
<tr>
<td>Choose from:</td>
<td>0 - naAUTO - Step size is set automatically.</td>
</tr>
<tr>
<td></td>
<td>1 - naMaunal - Step size is set manually using CenterFrequencyStepSize Property.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum</td>
</tr>
<tr>
<td>Default</td>
<td>1 - naMaunal</td>
</tr>
<tr>
<td>Examples</td>
<td>chan.CenterFrequencyStepSizeMode = naAUTO 'Write</td>
</tr>
<tr>
<td></td>
<td>value = chan.CenterFrequencyStepSizeMode 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_CenterFrequencyStepSizeMode(tagNAModes* pVal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_CenterFrequencyStepSizeMode(tagNAModes pVal);</td>
</tr>
<tr>
<td>Interface</td>
<td>IChannel25</td>
</tr>
</tbody>
</table>
## ChanActive Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return the state of activation of the SMU device.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>extSMU.ChanActive &lt;chanNum&gt;,&lt;value&gt;</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>extSMU</td>
<td>An ExteranlSMUDevice (object)</td>
</tr>
<tr>
<td>&lt;chanNum&gt;</td>
<td>Channel number for the external SMU</td>
</tr>
<tr>
<td>&lt;value&gt;</td>
<td>(Boolean) Choose from:</td>
</tr>
<tr>
<td>True</td>
<td>Channel is active.</td>
</tr>
<tr>
<td>False</td>
<td>Channel is NOT active.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>extSMU.ChanActive = 2,True  \textbf{Write}  \textbf{variable} = extSMU.ChanActive  \textbf{Read}</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_ChanActive( long chanNum, VARIANT_BOOL* value)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_ChanActive( long chanNum, VARIANT_BOOL value)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IExternalSMUDevice</td>
</tr>
</tbody>
</table>
**ChannelNumber Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Channel number of the Channel or Measurement object.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>object.ChannelNumber</code></td>
</tr>
</tbody>
</table>

**Variable**

<table>
<thead>
<tr>
<th>(Type) - Description</th>
<th></th>
</tr>
</thead>
</table>
| object              | A Channel *(object)*  
or      | A Measurement *(object)*  |

**Return Type** Long Integer

**Default** Not applicable

**Examples**

- `chanNum = chan.ChannelNumber`  
  *returns the channel number*
- `chanNum = meas.ChannelNumber`  
  *returns the channel number of the measurement*

**C++ Syntax**

- `HRESULT get_ChannelNumber(long *pVal)`

**Interface**

- IChannel
- IMeasurement
# Channels Property

**Description**  
Sets and returns the list of channels to be calibrated during the Cal All session.

**VB Syntax**  
`calAll.Channels = chans`

**Variable**  
*(Type)* - Description

- `calAll` A `CalibrateAllChannels` *(object)*
- `chans` (Variant) Array of channel numbers to be calibrated, separated by commas. These channels must already exist.

**Return Type**  
Variant

**Default**  
The existing channels

**Examples**  
`calAll.Channels = Array(1,2,3) 'sets channels to be cal'd`

`chans = calAll.Channels 'returns the channel numbers to be cal'd`

**C++ Syntax**

```
HRESULT get_Channels (VARIANT selectedChannels);

HRESULT put_Channels (VARIANT* selectedChannels);
```

**Interface**  
`ICalibrateAllChannels`
CharacterizationNumber Property

**Description**  
Sets and reads the number to which the user characterization will be stored in the ECal module. The number must be set before sending `Initialize` or the default value (1) will be used.

**VB Syntax**  
`userChar.CharacterizationNumber = value`

**Variable**
- **(Type)** - Description
- **userChar** - An `IECalUserCharacterizer` Object
- **value** - (Long) User Characterization number. Choose a value between 1 and 12.

**Return Type**  
Long Integer

**Default**  
Not Applicable

**Examples**

```vbnet
userChar.CharacterizationNumber = 5
```

**C++ Syntax**  
`HRESULT put_CharacterizationNumber(long *Number);`

**Interface**  
`IECalUserCharacterizer`
### CharacterizeMixerOnly Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns whether to perform ONLY a mixer characterization.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>$\text{VMC}.\text{CharacterizeMixerOnly} = \text{bool}$</td>
</tr>
<tr>
<td><strong>Variable</strong> (Type) - Description</td>
<td></td>
</tr>
<tr>
<td>$VMC$</td>
<td>VMCType (object)</td>
</tr>
<tr>
<td>$\text{bool}$</td>
<td>(Boolean)</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>Perform ONLY mixer characterization.</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>Perform both mixer characterization and calibration.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>value = VMC.CharacterizeMixerOnly</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT put_CharacterizeMixerOnly(VARIANT_BOOL bCharMixerOnly);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_CharacterizeMixerOnly(VARIANT_BOOL *bCharMixerOnly);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>VMCType</td>
</tr>
</tbody>
</table>
### CharFileName Property

#### Description
Specifies the mixer characterization (.S2P) file and immediately loads the file. Also specify the use of a characterization file with LoadCharFromFile Property.

#### VB Syntax
```vbnet
VMC.CharFileName = value
```

#### Variable (Type) - Description

- **VMC**: VMCType (object)
- **value**: String (Full path, file name, and extension of the mixer characterization file.)

#### Return Type
Not Applicable

#### Default
Not Applicable

#### Examples
```vbnet
VMC.CharFileName = "c:\users\public\network\analyzer\documents\default.S2P"
```

#### C++ Syntax
```cpp
HRESULT put_CharFileName(BSTR filename);
HRESULT get_CharFileName(BSTR *filename);
```

#### Interface
VMCType
### CharMixerReverse Property

**Description**
Specifies the direction in which to characterize the calibration mixer. [Learn more about the calibration mixer.](#)

**VB Syntax**

```vbnet
VMC.CharMixerReverse = bool
```

**Variable**

- **(Type)** - **Description**
  - **VMC**
    - VMCType (object)
  - **bool** (Boolean)

- **0** - Characterize the calibration mixer in the SAME direction as that specified in the mixer setup.
- **1** - Characterize the calibration mixer in the REVERSE direction as that specified in the mixer setup.

**Return Type**
Boolean

**Default**
0

**Examples**

```vbnet
VMC.CharMixerReverse = 0
```

**C++ Syntax**

```c++
HRESULT put_CharMixerReverse(VARIANT_BOOL bcharReverse);
HRESULT get_CharMixerReverse(VARIANT_BOOL *bcharReverse);
```

**Interface**
VMCType2
CitiContents Property - **Superseded**

**Description**
This command is replaced with SaveData Method

Specifies the contents of subsequent citifile saves using app.SaveCitiDataData or app.SaveCitiFormattedData

**VB Syntax**
`pref.CitiContents = value`

**Variable (Type) - Description**

- **pref**: A Preferences (object)
- **value**: (string) - Contents that will be saved with subsequent save commands. Choose from:
  - "Single" - Single trace
  - "Displayed" - All displayed traces
  - "Auto" - All displayed traces

**Return Type**
String

**Default**
"Auto"

**Examples**
```
pref.CitiContents = "Single" 'Write
content = pref.CitiContents 'Read
```

**C++ Syntax**
```
HRESULT get_CitiContents(BSTR *Contents)
HRESULT put_CitiContents(BSTR Contents)
```

**Interface**
IPreferences
## CitiFormat Property - Superseded

**Description**  
This command is replaced with `SaveData Method`.

Specifies the format of subsequent citifile saves using `app.SaveCitiFormattedData`.

**VB Syntax**  
`pref.CitiFormat = value`

**Variable**  
*(Type) - Description*

- `pref`  
  A `Preferences` *(object)*

- `value`  
  *(string)* - Format in which the citifile will be saved with subsequent save commands. Choose from:

  - "MA" - Linear Magnitude / degrees
  - "DB" - Log Mag / degrees
  - "RI" - Real / Imaginary
  - "Auto" - Format in which the trace is already displayed. If other than Log Mag, Linear Magnitude, or Real/Imag, then the format will be in Real/Imag.

**Return Type**  
String

**Default**  
"Auto"

**Examples**  
`pref.CitiFormat = "MA" 'Write`

`format = pref.CitiFormat 'Read`

**C++ Syntax**  
`HRESULT get_CitiFormat(BSTR *Format)`

`HRESULT put_CitiFormat(BSTR Format)`

**Interface**  
`IPreferences`
**CmnModeZConvPortImag Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the imaginary part of the impedance value for the common port impedance conversion function.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>fixture.CmnModeZConvPortImag(portNum) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong> (Type) - Description</td>
<td></td>
</tr>
<tr>
<td><code>fixture</code></td>
<td>A <code>Fixturing</code> (object)</td>
</tr>
<tr>
<td><code>portNum</code></td>
<td>(Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Imaginary part of the Impedance value. Choose a value between 0 and 1E18.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>fixture.CmnModeZConvPortImag(2) = 75 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = fixture.CmnModeZConvPortImag(1) 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_CmnModeZConvPortImag( short portNum, double *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_CmnModeZConvPortImag( short portNum, double newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IFixturing2</code></td>
</tr>
</tbody>
</table>
## CmnModeZConvPortReal Property

**Description**
Sets the real part of the impedance value for the common port impedance conversion function.

**VB Syntax**
```vbnet
fixture.CmnModeZConvPortReal(portNum) = value
```

**Variable**
- **fixture** *(Type)* - Description
  A Fixturing *(object)*
- **portNum** *(Integer)* - Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.
- **value** *(Double)* - Real part of the Impedance value. Choose a value between 0 and 1E18.

**Return Type**
Double

**Default**
See Common Mode Port Z Conversion Default

**Examples**
```vbnet
fixture.CmnModeZConvPortReal(2) = 75 'Write
value = fixture.CmnModeZConvPortReal(1) 'Read
```

**C++ Syntax**
```cpp
HRESULT get_CmnModeZConvPortReal( short portNum, double *pVal)
HRESULT put_CmnModeZConvPortReal( short portNum, double newVal)
```

**Interface**
IFixturing2
**CmnModeZConvPortZ0 Property**

**Description**  
Sets the impedance value for the common port impedance conversion function. Set either this single value or set the real and imaginary parts separately. The imaginary part is set to 0.0 using this command.

**VB Syntax**  
`fixture.CmnModeZConvPortZ0(portNum) = value`

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fixture</code></td>
<td>A Fixturing (object)</td>
</tr>
<tr>
<td><code>portNum</code></td>
<td>(Integer) The number of balanced ports. For example, if the device topology is SE-BAL-SE-BAL-SE-BAL, then this configuration would have 6 logical ports and 3 balanced ports. Learn more about logical and balanced ports.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Impedance value. Choose a value between 0 and 1E7.</td>
</tr>
</tbody>
</table>

**Return Type**  
Double

**Default**  
See Common Mode Port Z Conversion Default

**Examples**  

```vbnet
fixture.CmnModeZConvPortZ0(2) = 75 'Write
value = fixture.CmnModeZConvPortZ0(1) 'Read
```

**C++ Syntax**  
`HRESULT get_CmnModeZConvPortZ0( short portNum, double *pVal)`

`HRESULT put_CmnModeZConvPortZ0( short portNum, double newVal)`

**Interface**  
IFixturing2
Write/Read

About Fixturing

CmnModeZConvState Property

Description
Turns ON or OFF 4-port common port impedance conversion function. Must also set the fixture simulator function to ON using FixturingState Property.

VB Syntax
fixure.CmnModeZConvState = value

Variable
(fixture) - Description
value (Boolean)

False - Turns common port impedance conversion OFF
True - Turns common port impedance conversion ON

Return Type
Boolean

Default
False

Examples
fixture.CmnModeZConvState = False 'Write
value = fixture.CmnModeZConvState 'Read

C++ Syntax
HRESULT get_CmnModeZConvState( VARIANT_BOOL *pVal)

HRESULT put_CmnModeZConvState( VARIANT_BOOL newVal)

Interface
IFixturing2
## CompatibleCalKits Property

**Description**

Returns a comma-separated list of valid kits that use the specified connector type. This includes mechanical cal kits, applicable characterizations found within ECal modules currently connected to the VNA, and all user characterizations stored in VNA disk memory.

*Note:* The serial number is returned for ALL ECal modules that are connected with the connector type of the specified port. Previously, the returned list would include the serial numbers to distinguish the ECal modules only when two or more identical ECal models were connected to the VNA.

### VB Syntax

```vbnet
value = obj.CompatibleCalKits (port)
```

### Variable (Type) - Description

- **value** (Variant) Variable to store the returned list of Cal Kits.
- **obj** Any of the following:
  - `GuidedCalibration` (object) *Superseded* Replaced with `GetCompatibleCalKits` Method
  - `SMCType` (object)
  - `VMCType` (object)

- **port** (Long) Port number for which you want compatible kits.

First set the `ConnectorType` for the port.

### Return Type

Variant

### Default

Not Applicable

### Examples

```vbnet
Dim kits As Variant
kits = MySMC.CompatibleCalKits(1)
```

### C++ Syntax

```cpp
HRESULT get_CompatibleCalKits(long port, VARIANT* Kits);
```

### Interface

IGuidedCalibration

- `SMCType`
- `VMCType`
CompositeNormalizationMode Property

**Description**
Sets and returns the method by which CTB and CSO calculations are performed.

**VB Syntax**
```
imd.CompositeNormalizationMode = value
```

**Variable**
(Type) - Description
- `imd` A SweptIMD Object
- `value` (Enum)

0 - **naNone** - the normalized power is not used in calculation

1 - **naNumberOfCarriers** - CTB and CSO is corrected by subtracting $10 \times \log(N/2)$, where

- $N = \#\ of\ carriers\ for\ CTB$
- $N = \#\ of\ distortion\ products\ for\ CSO$

2 - **naPdBm** - the composited normalized power for CTB or CSO is treated as a dBm value

3 - **naPdBmV** - the composited normalized power for CTB or CSO is treated as a dBmV value.

**Note:** Power values are stored using the currently-set units. Therefore, first set units with this command, then set power values using:

- `CompositeNormalizedCSOPower`
- `CompositeNormalizedCTBPower`

**Return Type**
Enum

**Default**
**naNumberOfCarriers**

**Examples**
```
imd.CompositeNormalizationMode = naNone 'Write
value = imd.CompositeNormalizationMode 'Read
```

**C++ Syntax**
```
HRESULT get_CompositeNormalizationMode(tagNAIMDCompositeNormalizationMode *pVal)

HRESULT put_CompositeNormalizationMode(tagNAIMDCompositeNormalizationMode pVal)
```
Interface    ISweptIMD
### CompositeNormalizedCSOPower Property

**Description**
Sets and returns the CSO Power for POWER normalization mode. Valid only with measurement parameters: CSO2Lo and CSO2Hi and for Normalization Modes dBm and dBmV.

**VB Syntax**

```
imd.CompositeNormalizedCSOPower = value
```

**Variable** *(Type)* - Description

- **imd** *(Type)* - A SweptIMD Object
- **value** *(Type)* - (Double) Power level. The units are determined by `CompositeNormalizationMode` Property, which must be set first.

**Return Type**
Double

**Default**
0

**Examples**

```
imd.CompositeNormalizedCSOPower = -5 'Write
value = imd.CompositeNormalizedCSOPower 'Read
```

**C++ Syntax**

```
HRESULT get_CompositeNormalizedCSOPower(double *pVal)
HRESULT put_CompositeNormalizedCSOPower(double  pVal)
```

**Interface**
ISweptIMD
### CompositeNormalizedCTBPower Property

**Description**
Sets and returns the CSO Power. Valid only with measurement parameters: CTB Lo and CTB Hi and for Normalization Modes dBm and dBmV.

**VB Syntax**

```
imd.CompositeNormalizedCTBPower = value
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>imd</td>
<td>A SweptIMD Object</td>
</tr>
<tr>
<td>value</td>
<td>(Double) Power level. The units are determined by CompositeNormalizationMode Property, which must be set first.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
0

**Examples**

```
imd.CompositeNormalizedCTBPower = -5 'Write
value = imd.CompositeNormalizedCTBPower 'Read
```

**C++ Syntax**

```
HRESULT get_CompositeNormalizedCTBPower(double *pVal)
HRESULT put_CompositeNormalizedCTBPower(double  pVal)
```

**Interface**
ISweptIMD
### Compression Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Compression result of the PNOP marker search.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pnop Comp</td>
<td>Pnop Gain - Linear Gain (not shown on marker readout).</td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
comp = pnop.Compression
```

**Variable**

- **Type**: Description
- **comp** (double) - Variable to store returned value
- **pnop** A PNOP (object)

**Return Type**

Double

**Default**

Not applicable

**Examples**

```vbnet
comp = pnop.Compression 'Read
```

See example program

**C++ Syntax**

```cpp
HRESULT get_Compression(double* pNewVal)
```

**Interface**

IPNOP
### CompressionAlgorithm Property

**Description**
Set and read the algorithm method used to compute gain compression.

**VB Syntax**
`(gca.CompressionAlgorithm = value`

**Variable**
- **Type**
- **Description**
  - `gca` A GainCompression (object)
  - `value` (tagNAGCACompressionAlgorithm) - Algorithm method. Choose from:
    - `naCompressionFromLinearGain (0)`
    - `naCompressionFromMaximumGain (1)`
    - `naBackoffCompression (2)`
    - `naXYCompression (3)`
    - `naSaturation (4)`

**Return Type**
Enum

**Default**
`naCompressionFromLinearGain (0)`

**Examples**
```
gca.CompressionAlgorithm = naXYCompression 'Write
compAlg = gca.CompressionAlgorithm 'Read
```

**C++ Syntax**
```
HRESULT get_CompressionAlgorithm(tagNAGCACompressionAlgorithm* pVal)
HRESULT put_CompressionAlgorithm(tagNAGCACompressionAlgorithm newVal)
```

**Interface**
IGainCompression
# CompressionBackoff Property

**Description**  
Set and read value for the BackOff compression algorithm.

**VB Syntax**  
`gca.CompressionBackoff = value`

**Variable**  
<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>gca</code></td>
<td>A GainCompression (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double)  Backoff value in dB. Choose from 30 to (-30)</td>
</tr>
</tbody>
</table>

**Return Type**  
Double

**Default**  
10

**Examples**  
- `gca.CompressionBackoff = 7`  
- `acqMode = gca.CompressionBackoff`

**C++ Syntax**  
- `HRESULT get_CompressionBackoff(double* pValue)`
- `HRESULT put_CompressionBackoff(double newValue)`

**Interface**  
IGainCompression
### CompressionDeltaX Property

**Description**
Set and read the "X" value in the delta X/Y compression algorithm.

**VB Syntax**

```vbnet
gca.CompressionDeltaX = value
```

**Variable**

- `gca` (Type) - A `GainCompression` object
- `value` (Type) - X value in dB. Choose from 30 to (-30)

**Return Type**
Double

**Default**
10

**Examples**

```vbnet
(gca.CompressionDeltaX = 'Write

xDelta = gca.CompressionDeltaX 'Read
```

**C++ Syntax**

```cpp
HRESULT get_CompressionDeltaX(double* pVal)

HRESULT put_CompressionDeltaX(double newVal)
```

**Interface**
 IGainCompression
## CompressionDeltaY

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the 'Y&quot; value in the delta X/Y compression algorithm.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>gca.CompressionDeltaY = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>gca</td>
<td>A GainCompression (object)</td>
</tr>
<tr>
<td>value</td>
<td>(double) -</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>9</td>
</tr>
<tr>
<td>Examples</td>
<td><code>gca.CompressionDeltaY = 7   'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>xDelta = gca.CompressionDeltaY   'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_CompressionDeltaY(double* pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_CompressionDeltaY(double newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IGainCompression</td>
</tr>
</tbody>
</table>
### CompressionInterpolation Property

**Description**
Sets whether or not interpolation should be performed on 2D measured compression data. Applies ONLY to 2D acquisition modes.

**VB Syntax**
```vbnet
gca.CompressionInterpolation = value
```

**Variable**
<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>gca</code></td>
<td>A GainCompression (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(boolean) - Choose from:</td>
</tr>
<tr>
<td>True</td>
<td>Interpolate the results</td>
</tr>
<tr>
<td>False</td>
<td>Do NOT interpolate the results but return the value closest to compression.</td>
</tr>
</tbody>
</table>

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
gca.CompressionInterpolation = True 'Write
compInt = gca.CompressionInterpolation 'Read
```

**C++ Syntax**
```cpp
HRESULT get_CompressionInterpolation(VARIANT_BOOL* pVal)

HRESULT put_CompressionInterpolation(VARIANT_BOOL newVal)
```

**Interface**
IGainCompression
### CompressionLevel Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the desired gain reduction (from reference gain). This value is used for Compression Methods: Compression from Linear Gain and Compression from Maximum Gain.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>gca.CompressionLevel = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>gca</code></td>
<td>A GainCompression (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) - Compression level in dB. Choose a value greater than 0.1 dB.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>1</td>
</tr>
<tr>
<td>Examples</td>
<td><code>gca.CompressionLevel = 1.5</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>compLevel = gca.CompressionLevel</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_CompressionLevel(double* pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_CompressionLevel(double newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IGainCompression</td>
</tr>
</tbody>
</table>
## CompressionMax Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Compression Max result of a PSat or PNOP marker search.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Comp Max = Gain Max - Linear Gain (not shown on PNOP marker readout).</td>
</tr>
</tbody>
</table>

### VB Syntax

```vbnet
compMax = pMarker.CompressionMax
```

### Variable (Type) - Description

- `compMax` (double) - Variable to store returned value
- `pMarker` A PNOP (object) or a PSaturation (object)

### Return Type

- Double

### Default

- Not applicable

### Examples

```vbnet
compMax = pMark.CompressionMax  'Read
```

See example program

### C++ Syntax

```cpp
HRESULT get_CompressionMax(double* pNewVal)
```

### Interface

- IPNOP or IPSaturation
## CompressionLevel (Marker) Property

**Description**  
Set and read the marker compression level. First use `SearchCompressionPoint` to create the compression marker.

**VB Syntax**  
```vbnet
mkr.CompressionLevel = value
```

**Variable (Type) - Description**
- `mkr`  
  A Marker (object)
- `value`  
  (Double) - Compression level in dB. Choose any number between: -500 dB to 500 dB

Standard gain compression values are positive.

**Return Type**  
Double

**Default**  
1

**Examples**  
See example program

**C++ Syntax**
```c++
HRESULT get_CompressionLevel(double* pVal)
HRESULT put_CompressionLevel(double newVal)
```

**Interface**  
IMarker4
### CompressionPin Property

**Description**
Reads the input power at the marker compression level. First issue `SearchCompressionPoint Method` or `Tracking Property`.

**VB Syntax**

```
value = mkr.CompressionPin
```

**Variable (Type) - Description**

- `mkr` A Marker (object)
- `value` (Double) - Variable to store the returned input power value.

**Return Type**
Double

**Default**
Not Applicable

**Examples**

```
compLevel=mkr.CompressionPin 'Read
```

See example program

**C++ Syntax**

```
HRESULT get_CompressionPin(double* pVal)
```

**Interface**
IMarker4
### CompressionPout Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Reads the output power at the marker compression level. First issue SearchCompressionPoint Method or Tracking Property.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = mkr.CompressionPout</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td>mkr</td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td>value</td>
<td><em>(Double)</em> - Variable to store the returned output power value.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>compLevel=mkr.CompressionPout</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_CompressionPout(double* pVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker4</td>
</tr>
</tbody>
</table>
### CompressionSaturation Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Compression Saturation result of a PSat marker search.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comp Sat</td>
<td>Gain Sat - Gain Linear</td>
</tr>
</tbody>
</table>

#### VB Syntax

```vb
compSat = pSat.CompressionSaturation
```

#### Variable (Type) - Description

- `compSat` (double) - Variable to store returned value
- `pSat` A `PSaturation` (object)

#### Return Type

Double

#### Default

Not applicable

#### Examples

```vb
compSat = pSat.CompressionSaturation 'Read
```

See example program

#### C++ Syntax

```cpp
HRESULT get_CompressionSaturation(double* pNewVal)
```

#### Interface

`IPSaturation`
**ConfigurationFile Property**

**Description**
Recalls an Interface Control file from the hard drive into the analyzer.

**VB Syntax**
```vbnet
IntControl.ConfigurationFile = filename
```

**Variable**
(Type) - Description

*IntControl*  An InterfaceControl (object)

*filename*  (string) - Full path, file name, and extension (.xml) of the file to recall.

Files are typically stored in "c:\users\public\network analyzer\documents"

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vbnet
IntControl.ConfigurationFile = "c:\users\public\network analyzer\documents\MySettings.xml"
```

**C++ Syntax**
```cpp
HRESULT put_ConfigurationFile(BSTR bstrFile)
```

**Interface**
IInterfaceControl
### Configurations Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns an array of stored configuration names that can be used with DeleteConfiguration Method and LoadConfiguration Method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>names = pathMgr. Configurations</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>names</code></td>
<td><em>(Variant array)</em> Variable to store the returned configuration names.</td>
</tr>
<tr>
<td><code>pathMgr</code></td>
<td>PathConfigurationManager <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant array</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>names = path. Configurations</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_Configurations (VARIANT* configurations );</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPathConfigurationManager</td>
</tr>
</tbody>
</table>
## ConfirmPreset Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return preset confirmation. If preset confirmation is OFF, pressing the green PRESET key presets the instrument and opens the Preset softkey menu. If preset confirmation is ON, pressing the green PRESET causes the Preset menu to appear.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pref.ConfirmPreset = bool</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>pref</code></td>
<td>A <code>Preferences</code> (object)</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><strong>(boolean)</strong></td>
</tr>
<tr>
<td><code>False</code></td>
<td>Set preset confirmation to OFF as the default.</td>
</tr>
<tr>
<td><code>True</code></td>
<td>Set preset confirmation to ON as the default.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pref.ConfirmPreset = True 'Write prefer = pref.ConfirmPreset 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_ConfirmPreset(VARIANT_BOOL *presetState)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_ConfirmPreset(VARIANT_BOOL presetState)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IPreferences18</code></td>
</tr>
</tbody>
</table>
### ConnectorType Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or queries the connector type for the specified port.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>obj.ConnectorType (port) = value</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>obj</td>
<td>Any of the following:</td>
</tr>
<tr>
<td>GuidedCalibration (object)</td>
<td></td>
</tr>
<tr>
<td>SMCType (object)</td>
<td></td>
</tr>
<tr>
<td>VMCType (object)</td>
<td></td>
</tr>
<tr>
<td>port</td>
<td>(Long) Port number of the connector type.</td>
</tr>
<tr>
<td>For Guided Cals and SMC, select port number.</td>
<td></td>
</tr>
<tr>
<td>For VMC calibrations:</td>
<td></td>
</tr>
<tr>
<td>● 1 - Mixer Input.</td>
<td></td>
</tr>
<tr>
<td>● Any unused port can be used for the mixer output.</td>
<td></td>
</tr>
<tr>
<td>● Output port of MUT +1 - Output port of the calibration mixer. Generally this is port 3.</td>
<td></td>
</tr>
<tr>
<td><strong>value</strong></td>
<td>(String) - Connector type. <strong>Case-sensitive.</strong></td>
</tr>
<tr>
<td>Use ValidConnectorType Property to list connector types.</td>
<td></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>SMC.ConnectorType(1) = &quot;APC 3.5 male&quot;</td>
</tr>
<tr>
<td></td>
<td>value = SMC.ConnectorType(1)</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_ConnectorType(long port, BSTR *connector)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_ConnectorType(long port, BSTR connector)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IGuidedCalibration</td>
</tr>
<tr>
<td></td>
<td>SMCType</td>
</tr>
<tr>
<td></td>
<td>VMCType</td>
</tr>
</tbody>
</table>
ConnectorType Property

**Description**  Sets or queries the connector type for the specified port.

**VB Syntax**  `ecalUser.ConnectorType (port) = value`

**Variable**  
- **Type** - **Description**
  - `ecalUser`  An ECalUserCharacterizer (object)
  - `port`  (Enum) ECal port for which connector type is to be set. Choose from:
    - 1 or `naECalPort_A`
    - 2 or `naECalPort_B`
    - 3 or `naECalPort_C`
    - 4 or `naECalPort_D`
  - `value`  (String) - Connector type.

When the User Characterization is to be stored in the ECal module, then the connector type is limited to a Factory-defined connector type. See the list.

When the User Characterization is to be stored in VNA disk memory, then the connector type can also be a User-defined connector type.

**Return Type**  String

**Default**  "" (Empty String)

**Examples**

```vbnet
ecalUser.ConnectorType(naECalPort_B) = "APC 3.5 male" ' Write
Value = ecalUser.ConnectorType(naECalPort_B)
```

**C++ Syntax**

```cpp
HRESULT get_ConnectorType(NAECalPort port, BSTR *connector);

HRESULT put_ConnectorType(NAECalPort port, BSTR connector);
```

**Interface**  IECalUserCharacterizer
ControlLines Property

Description
Sets the control lines of the specified test set. Control lines, provided through the front panel connector of a test set, are used to control external equipment such as a part handler. See your test set documentation to learn more about control lines.

VB Syntax
\[ tset.\text{ControlLines}(\text{chNum}) = \text{value} \]

Variable (Type) - Description
\( tset \) A TestsetControl object.

OR

An E5091Testset object.

\( \text{chNum} \) (Integer) Channel number of the measurement.

\( \text{value} \) (Double) Data value used to set control lines. Values are obtained by adding weights from the following table that correspond to individual lines.

<table>
<thead>
<tr>
<th>Line</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td>8</td>
<td>128</td>
</tr>
</tbody>
</table>

- The E5091A interprets SENS:MULT1:OUTP 0 as all lines LOW.
- All "Z" and "H" series test sets interpret SENS:MULT1:OUTP 0 as all lines HIGH.

Refer to your test set documentation for setting control line values.

Return Type
Variant

Default
0

Examples
'For a Z5623A K64 test set, the following sets line 3 and 4 OFF; all other lines ON.'
testset1.ControlLines(2) = 12

See E5091A Example Program

See External Testset Program

**C++ Syntax**

HRESULT get_ControlLines(long channelNum, VARIANT *stateByte);
HRESULT put_ControlLines(long channelNum, VARIANT stateByte);

**Interface**

ITestsetControl

IE5091Testset

---
**CorrectionSubsettingState Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return the ON/OFF subset correction state. Learn more.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>corrMethods.CorrrectionSubsettingState = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>corrMethods</code></td>
<td>CorrectionMethods (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean)</td>
</tr>
<tr>
<td>True</td>
<td>Turns subset correction ON</td>
</tr>
<tr>
<td>False</td>
<td>Turns subset correction OFF</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>True (-1)</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>corrMethods.CorrrectionSubsettingState = True</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_CorrectionSubsettingState(BOOL *state);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_CorrectionSubsettingState(BOOL state);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICorrectionMethods2</td>
</tr>
</tbody>
</table>
Count Property

**Description**
Returns the number of items in a collection of objects.

**VB Syntax**
`object.Count`

**Variable**
*(Type) - Description*
*object* Any of the following *(objects)*:

- Cables Collection
- CalFactorSegments Collection
- CalFactorSegmentsPMAR Collection
- Cal Sets Collection
- Channels Collection
- E5091Testset Collection
- ExternalDevices Collection
- ExternalTestsets Collection
- GuidedCalibrationPowerSensors Collection
- LimitTest Collection
- Measurements Collection
- NaWindows Collection
- Ports Collection
- PowerLossSegments Collection
- PowerLossSegmentsPMAR_Collection.htm
- PowerSensors Collection
- Segments Collection
- Traces Collection
- PowerMeterInterfaces Collection

**Return Type**
Long Integer

**Default**
Not applicable
### Examples
```
numofchans = chans.Count 'return the number of channels
-Read
```

### C++ Syntax
```
HRESULT get_Count(long *pInterface)
```

### Interface
All listed above
**CouplePorts Property**

**Description**
Turns ON and OFF port power coupling. ON means the power level is the same for both ports. OFF means the power level may be set independently for each port.

**VB Syntax**
```vbnet
object.CouplePorts = value
```

**Variable**
*(Type) - Description*

`object` Channel *(object)*  

or

CalSet *(object)* - Read-only property

`value` *(enum NAStates)* Choose from:

- 0 - **NaOff** - Turns coupling OFF
- 1 - **NaOn** - Turns coupling ON

**Return Type**
Long Integer

- 1 - ON
- 0 - OFF

**Default**
NaON (1)

**Examples**
```
chan.CouplePorts = NaOff 'Write
```
```
couplport = chan.CouplePorts 'Read
```

**C++ Syntax**
```
HRESULT get_CouplePorts(tagNAStates *pState)
HRESULT put_CouplePorts(tagNAStates newState)
```

**Interface**
IChannel

|CalSet3
## CoupleChannelParams Property

**Description**
Turns ON and OFF Time Domain Trace Coupling. All of the measurements in the specified channel are coupled.

- To select Transform parameters to couple, use `Trans.CoupledParameters Property`
- To select Gating parameters to couple, use `Gate.CoupledParameters Property`

**VB Syntax**
```
chan.CoupleChannelParams = state
```

**Variable**
- `chan` (Type): A Channel (object)
- `state` (boolean)
  - `False` - Turns Trace Coupling OFF
  - `True` - Turns Trace Coupling ON

**Return Type**
Boolean

**Default**
True

**Examples**
```
chan.CoupleChannelParams = False 'Write
couple = chan.CoupleChannelParams 'Read
```

**C++ Syntax**
```
HRESULT get_CoupleChannelParams(VARIANT_BOOL *isCoupled);
HRESULT put_CoupleChannelParams(VARIANT_BOOL isCoupled);
```

**Interface**
IChannel5
### Coupled Property

**Description**
Sets and returns the state of coupling (ON or OFF) of this range to the primary range.

**VB Syntax**
\[ FOMRange.Coupled = value \]

**Variable**
- **object** (Type) - Description
  - An FOMRange (object)
- **value** (boolean) - State of coupling.

  **True** - Couple range to primary range.

  **False** - Do NOT couple to primary range.

**Return Type**
Boolean

**Default**
True

**Examples**
- `fomRange.Coupled = False 'this range is NOT coupled to the primary range.`
- `coupl = fomRange.Coupled 'Read`

**C++ Syntax**
- HRESULT get_Coupled(VARIANT_BOOL *pVal)
- HRESULT put_Coupled(VARIANT_BOOL pVal)

**Interface**
IFOMRange
**CoupledMarkers Property**

**Description**
Sets and Reads the state of Coupled Markers (ON and OFF).

See also: CoupledMarkersMethod Property

**VB Syntax**
```vbnet
app.CoupledMarkers = state
```

**Variable**
- **app** An Application *(object)*
- **state** *(boolean)*
  - **False** *(0)* - Turns Coupled Markers OFF
  - **True** *(1)* - Turns Coupled Markers ON

**Return Type**
Boolean
- **False** - OFF
- **True** - ON

**Default**
**False**

**Examples**
```vbnet
app.CoupledMarkers = True 'Write

coupl = app.CoupledMarkers 'Read
```

**C++ Syntax**
```cpp
HRESULT put_CoupledMarkers(VARIANT_BOOL bState)
HRESULT get_CoupledMarkers(VARIANT_BOOL *bState)
```

**Interface**
IApplication
CoupledMarkersMethod Property

Description  Sets and Reads the scope Coupled Markers.

**Note:** This command will not take effect until CoupledMarkers Property is turned on.

**Note:** The preset behavior of Coupled Markers depends on the setting of MarkCoupControlsMkrState, MarkCoupMethPresetIsChan, and MarkCoupPresetIsOn.

VB Syntax

```vbnet
app.CoupledMarkersMethod = value
```

Variable  *(Type)* - Description

- `app` An Application *(object)*
- `value` *(Enum as NAMarkerCouplingMethod)*

- **0** - `naMarkerCouplingAll` - Coupling occurs across all channels.
- **1** - `naMarkerCouplingChannel` - Coupling is limited to traces in the same channel.

Return Type  Enum

Default  **0** - `naMarkerCouplingAll`

Examples

```vbnet
app.CoupledMarkersMethod = naMarkerCouplingAll 'Write

coupl = app.CoupledMarkersMethod 'Read
```

C++ Syntax

```cpp
HRESULT put_CoupledMarkersMethod(tagNAMarkerCouplingMethod, value)

HRESULT get_CoupledMarkersMethod(NAMarkerCouplingMethod, *value)
```

Interface  IApplication20
**CoupledParameters Property (Gating)**

**Description**
Specifies the time domain gating parameters to be coupled. The settings for those parameters will be copied from the active measurement to all other measurements on the channel.

To turn coupling ON and OFF, use `CoupleChannelParams Property`

To specify Transform parameters to couple, use `Transform.CoupledParameters Property`

**VB Syntax**
```
gate.CoupledParameters = value
```

**Variable (Type) - Description**
- `trans` A Gating (object)
- `value` (Enum As NAGatingCoupledParams) - Parameters to couple. To specify more than one parameter, add the numbers. Choose from:
  - 1 - naGatingStimulusCoupled (Start, Stop, Center, and Span TIME settings.)
  - 2 - naGateStateCoupled (ON / OFF)
  - 4 - naGatingShapeCoupled (Minimum, Normal, Wide, and Maximum)
  - 8 - naGatingTypeCoupled (Bandpass and Notch)

**Return Type**
Enum

**Default**
29

**Examples**
```
gate.CoupledParameters = 15 'Couple all parameters
CP = gate.CoupledParameters 'Read
```

**C++ Syntax**

```
HRESULT get_CoupledParameters(long *lParams);

HRESULT put_CoupledParameters(long lParams);
```

**Interface**
IGating2
**CoupledParameters Property (Transform)**

### Description
Specifies the time domain transform parameters to be coupled. The settings for those parameters will be copied from the active measurement to all other measurements on the channel.

To turn coupling ON and OFF, use `CoupleChannelParams Property`

To specify Gating parameters to couple, use `Gate.CoupledParameters Property`

### VB Syntax
```
trans.CoupledParameters = value
```

### Variable (Type) - Description
- **trans**: A `Transform (object)`
- **value**: (Enum As NATransformCoupledParams) - Parameters to couple. To specify more than one parameter, add the numbers. Choose from:
  1. `naTransformStimulusCoupled` (Start, Stop, Center, and Span TIME settings.)
  2. `naTransformStateCoupled` (ON / OFF)
  4. `naTransformWindowCoupled` (Kaiser Beta / Impulse Width)
  8. `naTransformModeCoupled` (Low Pass Impulse, Low Pass Step, Band Pass)
  16. `naTransformDistMkrUnitCoupled` (Distance maker Units)

### Return Type
`Enum`

### Default
29

### Examples
```
trans.CoupledParameters = 31 'Couple all parameters
CP = trans.CoupledParameters 'Read
```

### C++ Syntax
```
HRESULT get_CoupledParameters(long *lParams);
HRESULT put_CoupledParameters(long lParams);
```

### Interface
`ITransform2`
### CouplePhasePortSettings Property

**Description**
Sets and returns whether to couple phase control settings (IFBW, Tolerance, Max Iterations).

**VB Syntax**
```
phase.CouplePhasePortSettings(srcPort) = value
```

**Variable**
- **phase**: A PhaseControl Object
- **srcPort**: (Long Integer) Source port for which to make phase control settings.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port

**value**: (Boolean) Coupling state. Choose from:
- **True** - Couple phase control settings. The phase control settings from <port> are copied to the other phase-controlled ports.
- **False** - Do NOT couple phase control settings. The phase control settings for each phase-controlled port are made independently.

**Return Type**
Boolean

**Default**
False

**Examples**
```
phase.CouplePhasePortSettings 1 = True ' Write
value = phase.CouplePhasePortSettings 2' Read
```

**C++ Syntax**
```
HRESULT get_CouplePhasePortSettings(long port, VARIANT_BOOL* pVal);

HRESULT put_CouplePhasePortSettings(long port, VARIANT_BOOL newVal);
```

**Interface**
IPhaseControl
**CoupleTonePower Property**

**Description**
Sets and returns the ON | OFF state of power coupling for F1 and F2.

**VB Syntax**

```vbnet
object.CoupleTonePower = value
```

**Variable (Type) - Description**

- **object**: A SweptIMD or IMSpectrum Object
- **value**: (Boolean) - Choose from:
  - **True**: F1 and F2 power is coupled.
  - **False**: F1 and F2 power is NOT coupled. Set power levels individually.

**Return Type**
Boolean

**Default**
True

**Examples**

```vbnet
ims.CoupleTonePower = true 'Write
value = ims.CoupleTonePower 'Read
```

**C++ Syntax**

```cpp
HRESULT get_CoupleTonePower(VARIANT_BOOL* val)
HRESULT put_CoupleTonePower(VARIANT_BOOL val)
```

**Interface**
ISweptIMD
IMSpectrum
## CpuRevision Property

**Description**  
Returns a number that corresponds to the CPU speed of the VNA.

**VB Syntax**  
\[ value = cap.CpuRevision \]

**Variable (Type) - Description**  
\( value \) (String) - Variable to store the returned number. Use the following table to learn the CPU speed.

Reported CPU version  - Clock speed

- **1.0** - 266 MHz
- **2.0** - 500 MHz
- **3.0** - 1100 MHz
- **4.0** - 1600 MHz
- **5.0** - 2000 MHz
- **6.0** - 2000 MHz dual core
- **7.0** - 2200 MHz dual core

**cap**  
A `Capabilities (object)`

**Return Type**  
String

**Default**  
Not Applicable

**Examples**  
\[ value = cap.CpuRevision \]  
'Read'

**C++ Syntax**  
`HRESULT get_CpuRevision(BSTR *value);`

**Interface**  
ICapabilities6
# CSONumDistortionProducts Property

**Description**  
Sets and returns the “N = number of distortion products” value for the calculation of the CSO parameter.  
[Learn more.](#)

**VB Syntax**  
```
imd.CSONumDistortionProducts = value
```

**Variable (Type) - Description**  
- `imd` - A SweptIMD Object
- `value` - (Long Integer) Number of distortion products

**Return Type**  
Long Integer

**Default**  
40

**Examples**  
```
imd.CSONumDistortionProducts = True 'Write
value = imd.CSONumDistortionProducts 'Read
```

**C++ Syntax**  
```
HRESULT get_CSONumDistortionProducts(long *pVal)
HRESULT put_CSONumDistortionProducts(long *pVal)
```

**Interface**  
ISweptIMD
### CSOOffset Property

**Description**
Sets and returns the offset that is applied to CSO measurements. Valid only with measurement parameters: CSO2Lo and CSO2Hi.

**VB Syntax**
```vbnet
imd.CSOOffset = value
```

**Variable** *(Type)* - Description

- `imd` A SweptIMD Object
- `value` (Double) Offset value in dBm.

**Return Type**
Double

**Default**
0

**Examples**
```vbnet
imd.CSOOffset = 2 'Write
value = imd.CSOOffset 'Read
```

**C++ Syntax**
```cpp
HRESULT get_CSOOffset(double *pVal)
HRESULT put_CSOOffset(double pVal)
```

**Interface**
ISweptIMD
**CTBOffset Property**

**Description**  
Sets and returns the offset that is applied to CTB measurements. Valid only with measurement parameters: CTB, CTBLo, CTBHi, CTBE, CTBELo, and CTBEHi.

**VB Syntax**  
`imd.CTBOffset = value`

**Variable**  
`imd` (Type) - Description  
A SweptIMD Object

`value` (Type) - Description  
(Double) Offset value in dBm.

**Return Type**  
Double

**Default**  
0

**Examples**  
`imd.CTBOffset = 2 'Write`

`value = imd.CTBOffset 'Read`

**C++ Syntax**  
`HRESULT get_CTBOffset(double *pVal)`
`HRESULT put_CTBOffset(double pVal)`

**Interface**  
ISweptIMD
**CTBXMODNumCarriers Property**

**Description**
Sets the “N = Total number of carriers” value used in the calculation of the XMOD and CTB parameter.

**Learn more.**

**VB Syntax**

```vbnet
imd.CTBXMODNumCarriers = value
```

**Variable (Type) - Description**

- **imd** A SweptIMD Object
- **value** (Long Integer) Number of carriers.

**Return Type**
Long Integer

**Default**
40

**Examples**

```vbnet
imd.CTBXMODNumCarriers = 15 'Write
value = imd.CTBXMODNumCarriers 'Read
```

**C++ Syntax**

```c++
HRESULT get_CTBXMODNumCarriers(double *pVal)
HRESULT put_CTBXMODNumCarriers(double *pVal)
```

**Interface**
ISweptIMD
### CurrentLimit Property

**Description**
Sets and returns the maximum output current value of the external DC Source. This command supports Keysight B2900A and N6700 series devices only.

**VB Syntax**
```
extDC.CurrentLimit (devicename) = value
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>extDC</code></td>
<td>A <code>ExternalDCDevice</code> (object)</td>
</tr>
<tr>
<td><code>devicename</code></td>
<td>(String) Name of the device.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Current limit value.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
0

**Examples**
```
extDC.CurrentLimit("myDCDevice") = 4 'Write
Limit = extDC.CurrentLimit("myDCDevice") 'Read
```

**C++ Syntax**
```c++
HRESULT get_CurrentLimit(BSTR devicename, double *cLimit)
HRESULT put_CurrentLimit(BSTR devicename, double newLimit)
```

**Interface**
`IExternalDCDevice2`
**CustomBalNegativePort Property**

**Description**  
Returns the VNA physical port number that is connected to the Negative side of the specified logical balanced port of the DUT.

**VB Syntax**  
```
var = balTopology.CustomBalNegativePort(logicalBalancedPort)
```

**Variable**  
*(Type)* - Description

- `balTopology` A `BalancedTopology` *(object)*
- `var` (Long Integer) Variable to store the returned value.
- `logicalBalancedPort` (Long Integer) Logical balanced port number.

**Return Type**  
Long Integer

**Default**  
Not Applicable

**Examples**  
'In the following example, 6 logical ports are mapped to 8 physical VNA ports.  
'Logical ports 3 and 6 are balanced.  
'Logical port 3 is queried returning physical port 4.

```c
int portlist[] = {1,2,3,4,5,6,7,8}
balTopology.SetCustomDuTTopology("SSBSSB", portlist)  
variable = balTopology.CustomBalNegativePort(3)  
```

**C++ Syntax**  
```
HRESULT get_CustomBalNegativePort(long *logicalBalancedPort)
```

**Interface**  
`IBalancedTopology3`
### CustomBalPositivePort Property

**Description**
Returns the VNA physical port number that is connected to the Positive side of the specified logical balanced port of the DUT.

**VB Syntax**
```
var = balTopology.CustomBalPositivePort(logicalBalancedPort)
```

**Variable**
- **balTopology**: A `BalancedTopology (object)`
- **var**: (Long Integer) Variable to store the returned value.
- **logicalBalancedPort**: (Long Integer) Logical balanced port number.

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**
```
'In the following example, 6 logical ports are mapped to 8 physical VNA ports.
'Logical ports 3 and 6 are balanced.
'Logical port 3 is queried returning physical port 3.
Int portlist[] = {1,2,3,4,5,6,7,8}
balTopology.SetCustomDuTTopology ("SSBSSB", portlist)
variable = balTopology.CustomBalPositivePort(3) 'Read
```

**C++ Syntax**
```
HRESULT get_CustomBalPositivePort(long *logicalBalancedPort)
```

**Interface**
IBalancedTopology3
**CustomCalConfiguration Property**

**Description**
Calibration for the following VNA Applications is performed using the GuidedCalibration Object. This command provides access to additional Properties and Methods which extends the GuidedCal Object.

<table>
<thead>
<tr>
<th>Meas Type</th>
<th>Custom Cal Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gain Compression</td>
<td>IGainCompressionCal Object</td>
</tr>
<tr>
<td>Noise Figure</td>
<td>INoiseCal Object</td>
</tr>
<tr>
<td>Swept IMD</td>
<td>ISweptIMD Cal Object</td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
set custCal = cal.CustomCalConfiguration()
```

**Variable**

- **(Type)** - Description
  - `custCal` *(object)* The handle to an interface that provides application-specific calibration properties.
  - `cal` GuidedCalibration Object.

**Return Type**

Depends on the MeasType. See above table.

**Default**

None

**Examples**

See examples: NoiseFigure GainCompression SweptIMD

**C++ Syntax**

```c++
HRESULT CustomCalConfiguration(IDispatch** value);
```

**Interface**

IGuidedCalibration4
CustomChannelConfiguration Property

Description Returns a handle to the custom application object on the active channel. You can either (1) use the handle directly to access measurement properties and methods, or (2) set a variable to the measurement object. The variable retains a handle to the original measurement.

Currently, the custom application objects to which this property provides access are:

- NoiseFigure Object
- GainCompression Object
- SweptIMD Object

VB Syntax
1) set custChan = chan.CustomChannelConfiguration. <setting>
or
2) set custChan = app.ActiveChannel.CustomChannelConfiguration

Variable (Type) - Description

custChan A variable in which the handle to a custom application is returned. (object)
chan A Channel (object)
<setting> A property or method setting on the custom application object.

Return Type Custom application object
Default None

Examples See examples: NoiseFigure GainCompression SweptIMD

C++ Syntax
HRESULT CustomChannelConfiguration(IDispatch** value);

Interface IChannel12
**CustomMeasurementConfiguration Property**

**Description**
Returns a handle to a custom measurement object on the active channel. You can use the handle to access custom measurement properties and methods.

Currently, the custom measurement objects to which this property provides access is:

- **GainCompressionMeas Object**

**VB Syntax**

```vbnet
Set custMeas = meas.CCustomMeasurementConfiguration
```

**Variable**  
*(Type)* - Description  
*custMeas* A variable in which the handle to a custom measurement is returned. *(object)*  
*chan* A *Measurement* *(object)*

**Return Type**  
Custom Measurement object

**Default**  
None

**Examples**

See examples: GainCompression

**C++ Syntax**

```cpp
HRESULT CustomMeasurementConfiguration(IDispatch** value);
```

**Interface**  
IMeasurement12
### CustomPhysicalPortsSequence Property

**Description**
Returns an array of physical port numbers corresponding to the logical ports. To view the type of each logical port, use the CustomPortTypeSequence command.

**VB Syntax**
```vbnet
var = balTopology.CustomPhysicalPortsSequence
```

**Variable**
- **(Type)** - Description
  - `balTopology` **A** BalancedTopology **(object)**
  - `var` **(Variant)** Variable to store the returned value.

**Return Type**
Variant

**Default**
Not Applicable

**Examples**
```
'In the following example, 6 logical ports are mapped to 8 physical VNA ports.
'This query returns the portlist, which is the sequence of the 8 physical VNA ports.
Int portlist[] = {1,2,3,4,5,6,7,8}
balTopology.SetCustomDuTTopology ("SSBSSB", portlist)
variable = balTopology.CustomPhysicalPortsSequence 'Read
```

**C++ Syntax**
```cpp
HRESULT get_CustomPhysicalPortsSequence(VARIANT *physicalPorts)
```

**Interface**
IBalancedTopology3
CustomPortTypeSequence Property

**Description**
Returns a string containing the characters "S" (single-ended) or "B" (balanced) corresponding to the logical port types. To view the physical ports, use the `CustomPhysicalPortsSequence` command.

**VB Syntax**
`var = balTopology.CustomPortTypeSequence`

**Variable (Type) - Description**
- `balTopology` A `BalancedTopology` (object)
- `var` (String) Variable to store the returned value.

**Return Type** String

**Default** Not Applicable

**Examples**
In the following example, 6 logical ports are mapped to 8 physical VNA ports.
This query returns `SSBSSB`, which is the sequence of the 6 logical ports.

```
Int portlist[] = {1,2,3,4,5,6,7,8}
balTopology.SetCustomDuTTopology("SSBSSB", portlist)
variable = balTopology.CustomPortTypeSequence 'Read
```

**C++ Syntax**
`HRESULT get_CustomPortTypeSequence(BSTR *portType)`

**Interface** IBalancedTopology3
## CustomSEPhysicalPort Property

**Description**  
Returns the VNA physical port number that is mapped to the specified single-ended logical port.

**VB Syntax**  
```vbnet
var = balTopology.CustomSEPhysicalPort(logicalSingleEndedPort)
```

**Variable**  
**(Type) - Description**

- `balTopology`  
  A `BalancedTopology` object

- `var`  
  (Long Integer) Variable to store the returned value.

- `logicalSingleEndedPort`  
  (Long Integer) Logical single ended port number.

**Return Type**  
Long Integer

**Default**  
Not Applicable

**Examples**  
In the following example, 6 logical ports are mapped to 8 physical VNA ports.

- Logical ports 1, 2, 4 and 5 are single-ended.
- Logical port 5 is queried returning physical port 6.

```csharp
int portlist[] = {1, 2, 3, 4, 5, 6, 7, 8}
balance.SetCustomDuTTTopoTogy("SSBSSB", portlist)
variable = balance.CustomSEPhysicalPort(5) 'Read
```

**C++ Syntax**  
```c++
HRESULT get_CustomSEPhysicalPort(long *physicalPort)
```

**Interface**  
`IBalancedTopology3`
# CustomTopologyPortCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of configured logical ports.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>var = balTopology.CustomTopologyPortCount</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(<code>Type</code>) - Description</td>
</tr>
<tr>
<td><code>balTopology</code></td>
<td>A <code>BalancedTopology (object)</code></td>
</tr>
<tr>
<td><code>var</code></td>
<td>(Long Integer) Variable to store the returned value.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>'In the following example, 6 logical ports are mapped to 8 physical VNA ports.</td>
</tr>
<tr>
<td></td>
<td>'This query returns 6 (SSBSSB).</td>
</tr>
<tr>
<td></td>
<td>Int portlist[] = {1,2,3,4,5,6,7,8}</td>
</tr>
<tr>
<td></td>
<td><code>balTopology.SetCustomDuTTopology (&quot;SSBSSB&quot;, portlist)</code></td>
</tr>
<tr>
<td></td>
<td><code>variable = balTopology.CustomTopologyPortCount</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_CustomTopologyPortCount(long *logicalPortCount)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IBalancedTopology3</td>
</tr>
</tbody>
</table>
**CWFrequency Property**

**Description**  
Set the Continuous Wave (CW) frequency. Must first send chan.SweepType = naCWTimeSweep.

See Also: calset.CWFrequency Property

**VB Syntax**  
`object.CWFrequency = value`

**Variable (Type) - Description**

- `object` One of the following:
  - Channel (object)
  - FOMRange  (object) Range must be UNCOUPLED.

See also Measurement2 interface.

- `value` (double) CW frequency. Choose any number between: the minimum and maximum frequency limits of the analyzer  
Units are Hz

**Return Type**  
Double

**Default**  
1e9

**Examples**
```
chan.CWFrequency = 5e9  'Write

cwfreq = chan.CWFrequency  'Read
```

**C++ Syntax**

- HRESULT put_CWFrequency(double newVal)
- HRESULT get_CWFrequency(double *pVal)

**Interface**

IChannel

IFOMRange
## CWFrequency (Cal Set) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the CW frequency that is stored in the Cal Set.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = CalSet.CWFrequency(range)</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(double) - returned CW frequency in Hertz.</td>
</tr>
<tr>
<td>CalSet</td>
<td>CalSet (object)</td>
</tr>
<tr>
<td>range</td>
<td>(Long) Choose: 0</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>cw = calset.CWFrequency(0)</code> 'Reads the CW frequency stored in the cal set.'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_CWFrequency(long range, double *pVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>CalSet3</td>
</tr>
</tbody>
</table>
### Data Property

**Description**
Reads the next specified number of data points from the FIFO buffer. The data is returned in 32 bit real/imaginary pair. Data is cleared as it is read.

**Note:** This method is the slowest way to transfer data using COM. However, it is supported by all COM client programming languages. For better performance, try using DataInCompactForm.

**VB Syntax**

```vbnet
array = fifo.Data(count)
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>array</td>
<td>(Variant) Variable to store the returned data.</td>
</tr>
<tr>
<td>fifo</td>
<td>A FIFO Object</td>
</tr>
<tr>
<td>count</td>
<td>(Long Integer) Number of data points to read.</td>
</tr>
</tbody>
</table>

**Return Type**

Variant array. Each VARIANT is typed as a 4-byte floating point number.

**Default**

Not Applicable

**Examples**

```vbnet
value = fifo.Data(500) 'Read
```

**C++ Syntax**

```c++
HRESULT get_Data(long count, VARIANT * data);
```

**Interface**

IFIFO
## DataAndLimits Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return the color of Data and Limit Lines for nth trace in a window.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>trace(n).DataAndLimits = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td></td>
<td><strong>value</strong> <em>(Long Integer)</em> - RGB color of the DataAndLimits pen.</td>
</tr>
</tbody>
</table>

Convert the three RGB colors to an integer as follows:

\[
RGB = R + (G \times 2^8) + (B \times 2^{16})
\]

To find the three RGB values from the Display Colors dialog, click **Change Color**, then **Define Custom Color**.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Long</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Varies for each trace.</td>
</tr>
</tbody>
</table>
| Examples    | R = 10  
G = 10  
B = 10  
RGB = 10+(10\times2^8)+(10\times2^{16})  
trace1.DataAndLimits = RGB 'Write  
color = trace1.DataAndLimits 'Read |
| C++ Syntax  | HRESULT get_DataAndLimits(long* pVal);  
HRESULT put_DataAndLimits(long newVal);  
INTERFACE ITraceColors |
<table>
<thead>
<tr>
<th>Description</th>
<th>Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of bytes.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>varray = fifo.DataAsBytes(bytecount)</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>varray</code></td>
<td>(Variant) Variable to store the returned data.</td>
</tr>
<tr>
<td><code>fifo</code></td>
<td>A FIFO Object</td>
</tr>
<tr>
<td><code>bytecount</code></td>
<td>(Long Integer) Number of data points to read.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = fifo.DataAsBytes(4096) 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_DataAsBytes(long bytecount, VARIANT* varray);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFIFO2</td>
</tr>
</tbody>
</table>
# DataAsFloat32 Property

Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of 32-bit floating point (Float32) numbers.

**VB Syntax**

```vbnet
varray = fifo.DataAsFloat32(float32count)
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>varray</td>
<td>Variant</td>
<td>Variable to store the returned data.</td>
</tr>
<tr>
<td>fifo</td>
<td>A FIFO Object</td>
<td></td>
</tr>
<tr>
<td>float32count</td>
<td>(Long Integer)</td>
<td>Number of data points to read.</td>
</tr>
</tbody>
</table>

**Return Type**

Variant

**Default**

Not Applicable

**Examples**

```vbnet
value = fifo.DataAsFloat32(4096) 'Read
```

**C++ Syntax**

```cpp
HRESULT get_DataAsFloat32(long float32count, VARIANT* varray);
```

**Interface**

IFIFO2
## DataAsInt16 Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of 16-bit integers.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>varray = fifo.DataAsInt16(int16count)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>varray</code></td>
<td>(Variant) Variable to store the returned data.</td>
</tr>
<tr>
<td><code>fifo</code></td>
<td>A FIFO Object</td>
</tr>
<tr>
<td><code>int16count</code></td>
<td>(Long Integer) Number of data points to read.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = fifo.DataAsInt16(4096) 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_DataAsInt16(long int16count, VARIANT* varray);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFIFO2</td>
</tr>
</tbody>
</table>
## DataAsInt32 Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of 32-bit integers.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>varray = fifo.DataAsInt32(int32count)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td></td>
<td><code>varray</code> (Variant) Variable to store the returned data.</td>
</tr>
<tr>
<td></td>
<td><code>fifo</code> A FIFO Object</td>
</tr>
<tr>
<td></td>
<td><code>int32count</code> (Long Integer) Number of data points to read.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Variant</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = fifo.DataAsInt32(4096)</code> <code>Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_DataAsInt32(long int32count, VARIANT* varray);</td>
</tr>
<tr>
<td>Interface</td>
<td>IFIFO2</td>
</tr>
</tbody>
</table>


## DataBinCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of DFT points processed across the total RF span.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = sa.dft.DataBinCount</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>sa.dft</code></td>
<td>A <code>SpectrumAnalyzerDFT (object)</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(Long)</strong> Variable to store the returned value.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>value = sa.dft.DataBinCount 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_DataBinCount(long* val);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>ISpectrumAnalyzerDFT</code></td>
</tr>
</tbody>
</table>
## DataByteCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Reads the FIFO data buffer byte count.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = fifo.DataByteCount</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) Variable to store the returned data byte count.</td>
</tr>
<tr>
<td><code>fifo</code></td>
<td>A FIFO Object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = fifo.DataByteCount</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_DataByteCount(long* bytecount);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFIFO2</td>
</tr>
</tbody>
</table>
# DataByteSize Property

**Description**

Returns the byte size of the data to be exported in binary mode.

**Note:** Returned number can exceed the maximum integer number size. In that case, an error will be raised. For that reason, we provide an access to larger numbers with the same query and LSB or MSB suffixes.

**VB Syntax**

```vbnet
value = sa.dft.DataByteSize
```

**Variable (Type) - Description**

- `sa.dft` (object) - A SpectrumAnalyzerDFT
- `value` (Variant) - Variable to store the returned value.

**Return Type**

Variant

**Example**

```vbnet
value = sa.dft.DataByteSize 'Read
```

**C++ Syntax**

```cpp
HRESULT get_DataByteSize(VARIANT* val);
```

**Interface**

ISpectrumAnalyzerDFT
### DataByteSizeLOW Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the least significant bytes of the byte size of the data to be exported.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = sa.dft.DataByteSizeLOW</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>sa.dft</code></td>
<td>A SpectrumAnalyzerDFT <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long)</em> Variable to store the returned value.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Example</td>
<td><code>value = sa.dft.DataByteSizeLOW</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_DataByteSizeLOW(long* val);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ISpectrumAnalyzerDFT</td>
</tr>
</tbody>
</table>
## DataByteSizeHIGH Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the most significant bytes of the byte size of the data to be exported.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = sa.dft.DataByteSizeHIGH</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>sa.dft</code></td>
<td>A <code>SpectrumAnalyzerDFT</code> <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long)</em> Variable to store the returned value.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>value = sa.dft.DataByteSizeHIGH 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_DataByteSizeHIGH(long* val);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>ISpectrumAnalyzerDFT</code></td>
</tr>
</tbody>
</table>
### DataBytesPerBin Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the byte size of one data bin in binary mode.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = sa.dft.DataBytesPerBin</code></td>
</tr>
<tr>
<td><strong>Variable (Type)</strong> - Description</td>
<td></td>
</tr>
<tr>
<td><code>sa.dft</code></td>
<td>A <code>SpectrumAnalyzerDFT (object)</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) Variable to store the returned value.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>value = sa.dft.DataBytesPerBin 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_DataBytesPerBin(long* val);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISpectrumAnalyzerDFT</td>
</tr>
</tbody>
</table>
# DataCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the total number of data points in the FIFO buffer.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( value = fifo.\textbf{DataCount} )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td>( value )</td>
<td>(Long Integer) Variable to store the returned number of data points.</td>
</tr>
<tr>
<td>( fifo )</td>
<td>A FIFO Object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>( value = fifo.\textbf{DataCount} ) 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_DataCount(long *value)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFIFO</td>
</tr>
</tbody>
</table>
### DataExportMarkersEnabled Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables/disables adding marker data to the text file (*.txt) output.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>sa.dft.DataExportMarkersEnabled = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>sa.dft</strong> - A <code>SpectrumAnalyzerDFT</code> (object)</td>
</tr>
<tr>
<td></td>
<td><strong>value</strong> - (Boolean) Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <strong>OFF</strong> - Disable marker data to the file output.</td>
</tr>
<tr>
<td></td>
<td>1 - <strong>ON</strong> - Enable marker data to the file output.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>OFF</td>
</tr>
<tr>
<td>Examples</td>
<td><code>sa.dft.DataExportMarkersEnabled = ON</code> 'Write`</td>
</tr>
<tr>
<td></td>
<td><code>value = sa.dft.DataExportMarkersEnabled</code> 'Read`</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_DataExportMarkersEnabled(VARIANT_BOOL* enable)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_DataExportMarkersEnabled(VARIANT_BOOL enable)</td>
</tr>
<tr>
<td>Interface</td>
<td>ISpectrumAnalyzerDFT2</td>
</tr>
</tbody>
</table>
# DataExportWindowingFactor Property

**Description**
Returns the windowing factor for band power computation. This factor is derived from the window type (Gaussian, flat top, etc.). When doing the sum of linear power over a band, use this factor to compensate the side lobe effect of windowing to get an accurate band power value.

**VB Syntax**

```vbnet
value = sa.dft.DataExportWindowingFactor
```

**Variable**

- **Type**: Description
- **sa.dft**: A SpectrumAnalyzerDFT (object)
- **value**: (Double) Variable to store the returned value.

**Return Type**

Double

**Example**

```vbnet
value = sa.dft.DataExportWindowingFactor 'Read
```

**C++ Syntax**

```cpp
HRESULT get_DataExportWindowingFactor(double* WindowingFactor);
```

**Interface**

ISpectrumAnalyzerDFT2
### DataFirstRFBin Property

**Description**

Returns the frequency of the first RF bin.

**Note:** This value can differ slightly from the SA Sweep start frequency, the frequency of the first RF bin is aligned with the current DFT grid.

**VB Syntax**

```vbnet
value = sa.dft.DataFirstRFBin
```

**Variable**

*sa.dft* is a `SpectrumAnalyzerDFT` *(object)*

*value* is a *(Double)* Variable to store the returned value.

**Return Type**

`Double`

**Example**

```vbnet
value = sa.dft.DataFirstRFBin 'Read
```

**C++ Syntax**

```c++
HRESULT get_DataFirstRFBin(double* val);
```

**Interface**

`ISpectrumAnalyzerDFT`
DataFormat Property

Description  Sets and returns the data format.

VB Syntax  

\[ sa.dft.DataFormat = value \]

Variable  

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sa.dft</td>
<td>A SpectrumAnalyzerDFT (object)</td>
</tr>
<tr>
<td>value</td>
<td>(enum NASADataType) - Choose from:</td>
</tr>
</tbody>
</table>

0  naSAFloatMAGdb: Set data format to log magnitude in dBm.

1  naSAFloatAMPVolt: Set data format to linear magnitude in volts.

2  naSAPackedInt: Set data format to Packed Integers: a more compact (2 bytes) numeric representation for dBm. Each set of 2 bytes is a short number s, to get the dBm value compute \((s/200.0 - 36.165)\).

Return Type  Enum as NASADataType

Default  naSAFloatMAGdb

Examples  

\[ sa.dft.DataFormat = naSAFloatMAGdb \] 'Write

\[ FormatType = sa.dft.DataFormat \] 'Read

C++ Syntax  

HRESULT get_DataFormat(tagNASADataType* pVal)

HRESULT put_DataFormat(tagNASADataType newVal)

Interface  ISpectrumAnalyzerDFT
### DataInCompactForm Property

**Description**
Reads FIFO data the same as Data Property but returns the data in a more compact form of SAFEARRAY. This is significantly faster but it is not supported in all client environments.

**VB Syntax**

```vbnet
array = fifo.DataInCompactForm(count)
```

**Variable (Type) - Description**

- `array` (Variant) Variable to store the returned data
- `fifo` A FIFO Object
- `count` (Long Integer) Number of data points to read.

**Return Type**
Returns an array of 4 byte floating point numbers.

**Default**
Not Applicable

**Examples**

```vbnet
value = fifo.DataInCompactForm(500) 'Read
```

**C++ Syntax**

```
HRESULT get_DataInCompactForm(long count,VARIANT * data);
```

**Interface**
IFIFO
DataLevelThreshold Property

### Description
Sets and returns the threshold value (dBm). For text file output with verbose mode, only the frequencies with power greater than this threshold setting will be written to the file. This command can be used as a kind of simple spurious search.

### VB Syntax
```vbnet
sa.dft.DataLevelThreshold = value
```

### Variable
- **sa.dft** - Description
  - A SpectrumAnalyzerDFT (object)

### Return Type
Double

### Default
-60 dBm

### Examples
```vbnet
sa.dft.DataLevelThreshold = -5 dBm  'Write
value = sa.dft.DataLevelThreshold  'Read
```

### C++ Syntax
```csharp
HRESULT put_DataLevelThreshold(double threshold);
HRESULT get_DataLevelThreshold(double* threshold);
```

### Interface
ISpectrumAnalyzerDFT2
# DataLevelThresholdEnabled Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Enables/disables data level threshold mode. Set the threshold level using the DataLevelThreshold command.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>sa.dft.DataLevelThresholdEnabled = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td></td>
<td><strong>sa.dft</strong> A SpectrumAnalyzerDFT <em>(object)</em></td>
</tr>
<tr>
<td></td>
<td><strong>value</strong> <em>(Boolean)</em> Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>0 - OFF</strong> - Disable threshold mode.</td>
</tr>
<tr>
<td></td>
<td><strong>1 - ON</strong> - Enable threshold mode.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>OFF</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>sa.dft.DataLevelThresholdEnabled = ON</code> <em>Write</em></td>
</tr>
<tr>
<td></td>
<td><code>value = sa.dft.DataLevelThresholdEnabled</code> <em>Read</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_DataLevelThresholdEnabled(VARIANT_BOOL* enable)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_DataLevelThresholdEnabled(VARIANT_BOOL enable)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISpectrumAnalyzerDFT2</td>
</tr>
</tbody>
</table>
### DateTime Property

**Description**  
Returns the system date and time.

**VB Syntax**  
`datetime = app.DateTime`

**Variable**  
(Type) - Description

`datetime` Variant to store the date and time.

`app` An Application (object)

**Return Type**  
Variant of type VT_DATE.

**Default**  
Not Applicable

**Examples**  
`datetime = app.DateTime 'Read`

**C++ Syntax**  
`HRESULT get_DateTime(VARIANT *datetime);`

**Interface**  
IApplication22
### DCCorrection Property

**Description**
Sets and returns the correction ON/OFF state for a DC Meter and a DC Source.

**VB Syntax**
```
extDC.DCCorrection = value
```

**Variable**
- **(Type) - Description**
- `extDC` - An `ExternalDCDevice` *(object)*
- `value` - *(Boolean)* Correction ON/OFF state. Choose from:
  - **True** - Turn Correction ON
  - **False** - Turn Correction OFF

**Return Type**
Boolean

**Default**
False

**Examples**
```
extDC.DCCorrection = True 'Write
bool = extDC.DCCorrection 'Read
```

**C++ Syntax**
```
HRESULT get_DCCorrection (BOOL *pValue)
HRESULT put_DCCorrection (BOOL newVal)
```

**Interface**
IExternalDCDevice
**DCOffset Property**

**Description**
Sets and returns the offset correction value for an external DC Device which can be configured as either a DC Meter or a DC Source.

**VB Syntax**

```
extDC.DCOffset = value
```

**Variable**

- **extDC** (Type) - An ExternalDCDevice (object)
- **value** (Double) - DC offset value.

The VNA will display readings from a DC Meter as:

```
Display = (Meas'd value - Offset) * Scale
```

The VNA will adjust the output from a DC Source as:

```
Output  = (Set value - Offset) * Scale
```

**Return Type**
Double

**Default**
0

**Examples**

```
extDC.DCOffset = 4 'Write
offset = extDC.DCOffset 'Read
```

**C++ Syntax**

```
HRESULT get_DCOffset (double *pValue)

HRESULT put_DCOffset (double newVal)
```

**Interface**
IExternalDCDevice
DCOrder Property

Description Set and read the order for the specified DC source in the multidimensional sweep.

VB Syntax

```
md.DCOrder (name,port) = value
```

Variable (Type) - Description

- **name, port** (string) Name of the "DC source, port"
- **value** (long) Dimension order. Choose an integer value of 1 or higher.

Return Type long

Default 1

Examples

```
md.DCOrder ("AO1") = 2   'Write
value = md.DCOrder("MyDCSource,Port 1") 'Read
```

C++ Syntax

```
HRESULT put_DCOrder (BSTR name, long value);
HRESULT get_DCOrder (BSTR name, long* value);
```

Interface IMultiDimensionalSweep
## DCScale Property

**Description**
Sets and returns the scale correction value for an external DC Device which can be configured as either a DC Meter or a DC Source.

**VB Syntax**

```vbnet
extDC.DCScale = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>extDC</code></td>
<td>An ExternalDCDevice (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) DC scale value.</td>
</tr>
</tbody>
</table>

The VNA will display readings from a DC Meter as:

\[
\text{Display} = (\text{Meas'd value} - \text{Offset}) \times \text{Scale}
\]

The VNA will adjust the output from a DC Source as:

\[
\text{Output} = (\text{Set value} - \text{Offset}) \times \text{Scale}
\]

**Return Type**

Double

**Default**

1

**Examples**

```vbnet
extDC.DCScale = -2 'Write
slope = extDC.DCScale 'Read
```

**C++ Syntax**

```c++
HRESULT get_DCScale (double *pValue)
HRESULT put_DCScale (double newVal)
```

**Interface**

IExternalDCDevice
## DCSourcePointCount Property

**Description**  Set and read the number of steps the source will make across the specified source DC range. This setting is common to all sources.

**VB Syntax**  
```vbnet
sa.DCSourcePointCount = value
```

**Variable**  
- **sa**: A SpectrumAnalyzer (object)
- **value**: (Long) Point count. Choose an integer value of 1 or higher.

**Return Type**  Long

**Default**  1

**Examples**  
```vbnet
sa.DCSourcePointCount = 10 'Write
value = sa.DCSourcePointCount 'Read
```

**C++ Syntax**  
```c++
HRESULT put_DCSourcePointCount(long points);
HRESULT get_DCSourcePointCount(long* points);
```

**Interface**  ISpectrumAnalyzer
DCSourceSweepFirstDimension Property

Description Set and read the DC sweep order. The SA may be programmed to loop through a series of spectrum measurements at multiple RF source frequencies, multiple RF source powers, and multiple DC voltages. These settings determine whether the DC sources are swept before the RF power and frequencies are swept, or whether the DC sources are swept after the RF power and frequencies are swept.

VB Syntax

\[ \text{sa.DCSourceSweepFirstDimension(source) = value} \]

Variable (Type) - Description

- \( sa \): A SpectrumAnalyzer (object)
- \( value \): (Enum as NASADCSweepFirstTypes) Choose from:
  
  0 - naSADCFirst - Sweep through each DC voltage step first then sweep through the next frequency.
  
  1 - naSARFFirst - Sweep through each frequency step first then sweep through the next DC voltage.

Learn about these settings

Return Type Enum

Default \( \text{naSADCFirst} \)

Examples

\[ \text{sa.DCSourceSweepFirstDimension("Port 1").= naSADCFirst 'Write} \]

\[ \text{value = sa.DCSourceSweepFirstDimension("Port 1") 'Read} \]

C++ Syntax

HRESULT put_DCSourceSweepFirstDimension(tagNASADCSweepFirstTypes first);

HRESULT get_DCSourceSweepFirstDimension(tagNASADCSweepFirstTypes* first);

Interface ISpectrumAnalyzer
### DCSourceSweepState

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the ON/OFF state of the DC sources. If ON, the DC sources sweep between their start and stop voltages.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><strong>sa.DCSourceSweepState = value</strong></td>
</tr>
<tr>
<td><strong>Variable</strong> <em>(Type)</em> - Description</td>
<td><strong>sa</strong> A SpectrumAnalyzer <em>(object)</em></td>
</tr>
<tr>
<td><strong>value</strong> <em>(Boolean)</em> Choose from:</td>
<td><strong>0 - OFF</strong> - DC sweep OFF.</td>
</tr>
<tr>
<td></td>
<td><strong>1 - ON</strong> - DC sweep ON.</td>
</tr>
<tr>
<td><strong>Learn about these settings</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Return Type** Boolean

**Default** 0

**Examples**

<table>
<thead>
<tr>
<th><strong>Write</strong></th>
<th><strong>Write</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>sa.DCSourceSweepState = OFF</strong></td>
<td><strong>value = sa.DCSourceSweepState</strong></td>
</tr>
</tbody>
</table>

**C++ Syntax**

```cpp
HRESULT put_DCSourceSweepState(VARIANT_BOOL bEnable);

HRESULT get_DCSourceSweepState(VARIANT_BOOL* bEnable);
```

**Interface** ISpectrumAnalyzer
**DCState Property**

**Description**
Set and read the specified DC source’s ON/OFF state in the multi-dimensional sweep.

**VB Syntax**
```vbnet
md.DCState (name, port) = value
```

**Variable**
- `(Type)` - Description
  - `md` A `MultiDimensionalSweep` object which belongs to a SA channel.
  - `name, port` (string) Name of the "DC source, port"
  - `value` (Boolean) Choose from:
    - **0 - OFF** - Disable the specified DC source in multi-dimensional sweep.
    - **1 - ON** - Enable the specified DC source in multi-dimensional sweep.

**Return Type**
Boolean

**Default**
0

**Examples**
```vbnet
md.DCState ("AO1") = OFF   'Write
value = md.DCState ("MyDCSource,Port 1")   'Read
```

**C++ Syntax**
```csharp
HRESULT put_DCState (BSTR name, VARIANT_BOOL value);

HRESULT get_DCState (BSTR name, VARIANT_BOOL* value);
```

**Interface**
`IMultiDimensionalSweep`
**DCType Property**

**Description**  
Sets and returns the DC Type for an external DC Device which can be configured as either a DC Meter or a DC Source. This setting is used as the units for display on the VNA X-axis.

**VB Syntax**  
`extDC.DCType = value`

**Variable (Type) - Description**

- **extDC**  
  An ExternalDCDevice (object)
- **value**  
  (String) DC type. Choose from:
  
  "dBm", "A", "V", "W", "K", "F", "C"

**Return Type**  
String

**Default**  
"V"

**Examples**  
```
extDC.DCType = "A" 'Write
```
```
units = extDC.DCType 'Read
```

**C++ Syntax**  
```c++
HRESULT get_DCType (BSTR *pValue)
```

```c++
HRESULT put_DCType (BSTR newVal)
```

**Interface**  
IExternalDCDevice
# DefinedRoles Property

**Description**

This command replaces `GetSourceRoles Method`. Returns the roles for which sources can be used for the channel.

Use `RoleDevice` to assign a source to a role.

**VB Syntax**

```vbnet
value = chan.definedRoles
```

**Variable**

- `value` (Variant) - Variable to store the returned channel roles.
- `chan` A Channel (object)

**Return Type**

Variant

**Default**

Not Applicable

**Examples**

```vbnet
vRoles = chan.definedRoles  'Read
```

**C++ Syntax**

```c++
HRESULT get_DefinedRoles(Variant* roles);
```

**Interface**

IChannel22
### Delay (Segment Sweep) Property

**Description**
Sets or returns the segment delay property of the sweep segment. Enable delay using the [DelayOption Property](#) of Segments collection.

**VB Syntax**

```vbnet
seg.Delay = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>seg</code></td>
<td>A Segment <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Delay value in seconds.</td>
</tr>
</tbody>
</table>

**Return Type**

Double

**Default**

0

**Examples**

```vbnet
seg.Delay = 0.001 'Write
value = seg.Delay 'Read
```

**C++ Syntax**

```
HRESULT get_Delay(double* delay);

HRESULT put_Delay(double delay);
```

**Interface**

ISegment3
## Delay Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Returns the electrical delay value for the calibration standard.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>calstd.Delay = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>calstd</code></td>
<td>A CalStandard <em>(object)</em>. Use <code>calKit.GetCalStandard</code> to get a handle to the standard.</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(single) - Electrical delay in picoseconds</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Single</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>calstd.Delay = 12 'Write 12ps Delay</code></td>
</tr>
<tr>
<td></td>
<td><code>stdDelay = calstd.Delay 'Read the value of Delay</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_Delay(float *pVal)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_Delay(float newVal)</code></td>
</tr>
<tr>
<td>Interface</td>
<td><code>ICalStandard</code></td>
</tr>
</tbody>
</table>
**Delay (pulse) Property**

**Description**
Sets the pulse delay - the amount of time before a new pulse begins.

**VB Syntax**

```
pulse.Delay(n) = value
```

**Variable (Type) - Description**

- **pulse**
  A PulseGenerator (object)

- **n** (Integer) Pulse generator number. Choose from 0 to 4.
  
  Or use PulseGeneratorID to refer to an external pulse generator.

  0 is the generator that pulses the ADC.

- **value** (Double) Delay value in seconds. Choose a value from about 33ns to about 70 seconds.

**Return Type**
Double

**Default**
0

**Examples**

```
pulse.Delay(1) = 1ms 'Write
value = pulse.Delay(4) 'Read
```

**C++ Syntax**

```
HRESULT get_Delay(integer pulse, double* delay);
HRESULT put_Delay(integer pulse, double delay);
```

**Interface**
IPulseGenerator
**Delay Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the delay that should be applied by the VNA after the aux trigger input is received and before the acquisition is made.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong> Use on PNA-X ONLY. Other models do NOT have an Aux Input.</td>
<td></td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
auxTrig.Delay = value
```

**Variable**

- **Type** - Description
- `auxTrig` An `AuxTrigger` (object)
- `value` (double) - Delay value in seconds. Choose a value between 0 and 3.0 seconds.

**Return Type**

Double

**Default**

Not Applicable

**Examples**

```vbnet
auxTrig.Delay = 1.2 'Write 1.2s Delay
value = auxTrig.Delay 'Read the value
```

**C++ Syntax**

```c++
HRESULT get_Delay(double *val);
HRESULT put_Delay(double val);
```

**Interface**

IAuxTrigger
### DelayCalculationMethod Property

**Description**
Set and return the method of setting the delay through the calibration mixer.

To select Phase Reference Cal method for correcting an SMC+Phase measurement, use [ImportDataSet Method](#).

**VB Syntax**
```
smc.DelayCalculationMethod = value
```

**Variable**
- **Type** - Description
  - *smc* - An [SMCType](#) (object)
  - *value* - (Enum as NA\[DelayCalculationMethod](#))

**Return Type**
Enum

**Default**
0 - *na\[DelayCalculationMethod](#)_FixedDelay* - use a known delay value set with [FixedDelay Property](#)

1 - *na\[DelayCalculationMethod](#)_MixerCharacterizationFile* - use the S2P file set with [MixerCharacterizationFile Property](#)

**Example**
```
SMC.DelayCalculationMethod = na\[DelayCalculationMethod](#)_FixedDelay
```

**C++ Syntax**
```
HRESULT put_DelayCalculationMethod(tagNA\[DelayCalculationMethod](#) Value);

HRESULT get_DelayCalculationMethod(tagNA\[DelayCalculationMethod](#)* Value);
```

**Interface**
[SMCType5](#)
### DelayIncrement Property

**Description**
Sets the pulse delay increment. The delay increments with each pulse by the `<value>` amount.

For example, in this diagram the delay starts as 1. On the second pulse, delay=2. On the third pulse, delay=3.

**Important:** If $D + W$ is greater than $P$, then undefined VNA behavior results. There is NO error message or warning. Delay includes the incremented value.

This is useful for pulse profiling.

![Diagram](image-url)

**VB Syntax**

```vbnet
pulse.DelayIncrement(n) = value
```

**Variable (Type) - Description**

- **pulse**
  - A PulseGenerator (object)
- **n**
  - (Integer) Pulse generator number. Choose from 0 to 4.
  - Or use PulseGeneratorID to refer to an external pulse generator.
  - 0 is the generator that pulses the ADC.
- **value**
  - (Double) Delay increment value in seconds.

**Return Type**
Double

**Default**
0

**Examples**

```vbnet
pulse.DelayIncrement(1) = 1ms 'Write
value = pulse.DelayIncrement(4) 'Read
```

**C++ Syntax**

```csharp
HRESULT get_DelayIncrement(integer pulse, double* dIncre);
```

```csharp
HRESULT put_DelayIncrement(integer pulse, double dIncre);
```

**Interface**
IPulseGenerator
## DelayOption Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables or disables segment delay. Value is set using the Delay (Segment Sweep) Property on a Segment.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>segs.DelayOption = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>segs</code></td>
<td>A Segments collection (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(boolean)</td>
</tr>
<tr>
<td><code>True</code></td>
<td>Enable delay.</td>
</tr>
<tr>
<td><code>False</code></td>
<td>Disable delay.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>segs.DelayOption = True</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>delaystate = segs.DelayOption</code> 'Read'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_DelayOption(VARIANT_BOOL* pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_DelayOption(VARIANT_BOOL pVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISegments6</td>
</tr>
</tbody>
</table>
**DeltaFrequency Property**

**Description**
Sets and returns the fixed tone spacing value. Use with IMD sweep types:

- `naIMDToneCWSweep`
- `naIMDTonePowerSweep`
- `naIMDToneCenterFreqSweep`

**VB Syntax**

```
object.DeltaFrequency = value
```

**Variable (Type) - Description**

- `object` - A `SweptIMD` or `IMSpectrum` Object
- `value` - (Double) - Tone spacing frequency in Hz. Both the F1 and F2 tones MUST be within the frequency range of the VNA.

**Return Type**
Double

**Default**
1 MHz

**Examples**

```
imd.DeltaFrequency = 1e6 'Write
value = imd.DeltaFrequency 'Read
```

**C++ Syntax**

```
HRESULT get_DeltaFrequency(double *pVal)
HRESULT put_DeltaFrequency(double newVal)
```

**Interface**

- `ISweptIMD`
- `IIMSpectrum`
**DeltaFrequencyStart Property**

**Description**
Sets and returns the starting main tone separation for sweep type=naIMDDeltaFrequencySweep

**VB Syntax**
```vbnet
imd.DeltaFrequencyStart = value
```

**Variable (Type) - Description**
- `imd` (A SweptIMD Object)
- `value` (Double) - Starting tone separation between F1 and F2 in Hz. Both F1 and F2 tones MUST be within the frequency range of the VNA where:
  
  \[
  F1 \text{ (start)} = imd.FrequencyCenter - \frac{imd.DeltaFrequencyStart}{2}
  \]

  \[
  F2 \text{ (start)} = imd.FrequencyCenter + \frac{imd.DeltaFrequencyStart}{2}
  \]

**Return Type**
Double

**Default**
1 MHz

**Examples**
```vbnet
imd.DeltaFrequencyStart = 5e6 'Write
value = imd.DeltaFrequencyStart 'Read
```

**C++ Syntax**
```
HRESULT get_DeltaFrequencyStart(double *pVal))
HRESULT put_DeltaFrequencyStart(double newVal)
```

**Interface**
ISweptIMD
### DeltaFrequencyStop Property

**Description**  
Sets and returns the stop spacing of the main tones. Use with `sweep type=naIMDDeltaFrequencySweep`

**VB Syntax**  
`imd.DeltaFrequencyStop = value`

**Variable (Type) - Description**

- `imd` (A SweptIMD Object)
- `value` (Double) - Stopping tone separation between F1 and F2 in Hz. Both F1 and F2 tones MUST be within the frequency range of the VNA where:

  \[
  F1 \text{ (stop)} = \text{imd.FrequencyCenter} - \frac{\text{imd.DeltaFrequencyStop}}{2} \\
  F2 \text{ (stop)} = \text{imd.FrequencyCenter} + \frac{\text{imd.DeltaFrequencyStop}}{2}
  \]

**Return Type**  
Double

**Default**  
10 MHz

**Examples**

```vbnet
imd.DeltaFrequencyStop = 20e6 'Write
value = imd.DeltaFrequencyStop 'Read
```

**C++ Syntax**

```c++
HRESULT get_DeltaFrequencyStop(double *pVal))
HRESULT put_DeltaFrequencyStop(double newVal)
```

**Interface**  
ISweptIMD
### DeltaMarker Property

**Description**  
Sets a marker as a delta marker. The reference marker must already be turned ON. See \texttt{meas.ReferenceMarkerState}

**VB Syntax**  
\texttt{mark.DeltaMarker = state}

**Variable**  
\begin{itemize}
  \item \texttt{mark} \, \textbf{(Type)} - \textbf{Description}
  \item \texttt{state} \, \textbf{(boolean)} -
\end{itemize}

- **True** - marker is a delta marker
- **False** - marker is NOT a delta marker

**Return Type**  
Boolean

**Default**  
False

**Examples**
\begin{itemize}
  \item \texttt{mark.DeltaMarker = True} \textit{'Write}
  \item \texttt{delta = mark.DeltaMarker} \textit{'Read}
\end{itemize}

**C++ Syntax**

- \texttt{HRESULT get_DeltaMarker(VARIANT_BOOL bState)}
- \texttt{HRESULT put_DeltaMarker(VARIANT_BOOL *bState)}

**Interface**  
\texttt{IMarker}
### Description Property

Sets or returns the descriptive string assigned to the Cal Set. Change this string so that you can easily identify each Cal Set constructed.

**VB Syntax**

```
CalSet.Description = value
```

**Variable**

- **Type**: Description
- **CalSet** (object) - A Cal Set object
- **value** (string) – Description of the Cal Set

**Return Type**

String

**Default**

“CalSet_n” where n is an integer number.

**Examples**

```vbscript
CalSet.Description = "My Cal Set" 'Write
desc = CalSet.Description 'Read
```

**C++ Syntax**

```
HRESULT get_Description(BSTR *pVal)
HRESULT put_Description(BSTR newVal);
```

**Interface**

ICalSet
### DescriptiveText Property

**Description**
Write and read descriptive text associated with the configuration. This text is displayed in the path configuration dialog. Text is generally used to describe external connections that must be made manually to complete the configuration setup.

**VB Syntax**
```vbnet
pathConfig.DescriptiveText = text
```

**Variable**
- **(Type) - Description**
- **name** (String) Variable to store the returned configuration name.
- **pathConfig** A PathConfiguration (object)
- **text** (String) Descriptive text enclosed in quotes.

**Return Type** String

**Default** Not Applicable

**Examples**
```vbnet
pathConf.DescriptiveText "here are the instructions for connecting the device for this configuration"
```

**C++ Syntax**
```cpp
HRESULT get_DescriptionText(BSTR* pConnectionText );
HRESULT put_DescriptionText(BSTR connectionText );
```

**Interface** IPathConfiguration
## DetectorFunction Property

**Description**  
Set and read the detector type.

**VB Syntax**  
\( sa.DetectorFunction = value \)

**Variable**  
*(Type)* - Description |
---|---
\( sa \) | A `SpectrumAnalyzer (object)`
\( value \) | *(Enum as NASADetectorFunction)* Choose from:

- 0 - `naDTAverage`
- 1 - `naDTSample`
- 2 - `naDTPeak`
- 3 - `naDTNormal`
- 4 - `naDTNegPeak`
- 5 - `naDTPeakSample`
- 6 - `naDTPeakAverage`

Learn more about these settings.

**Return Type**  
Enum

**Default**  
2 - `naDTPeak`

**Examples**  
\( sa.DetectorFunction = naDTAverage \)  
\( value = sa.DetectorFunction \)

See an example program.

**C++ Syntax**  
\( \	ext{HRESULT put\_DetectorFunction(tagNASADetectorType type);} \)
\( \	ext{HRESULT get\_DetectorFunction(tagNASADetectorType* type);} \)

**Interface**  
`ISpectrumAnalyzer`
## DeviceInputPort Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the VNA port number to be used for the DUT input.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>port = obj.DeviceInputPort</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>port</code></td>
<td>(long) Variable to store the VNA port number of the DUT input.</td>
</tr>
<tr>
<td><code>mixer</code></td>
<td>A Mixer (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Default</td>
<td>1</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = mixer.DeviceInputPort</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_DeviceInputPort(long *pVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMixer8</td>
</tr>
</tbody>
</table>
## DeviceInputPort Property

**Description**
Read the VNA port number which is connected to the DUT input.

Use `SetPortMap` Method to change the port mapping.

**VB Syntax**

```vbnet
obj.DeviceInputPort
```

**Variable**

**`obj`**
A `Converter (object)` or
A `GainCompression (object)` or
A `SweptIMD (object)` or
An `IMSpectrum (object)`
A `NoiseFigure (object)`

**Return Type**
Integer

**Default**
1

**Examples**

```vbnet
inPort = gca.DeviceInputPort 'Read
```

**C++ Syntax**

```cpp
HRESULT get_DeviceInputPort(int* pVal)
```

**Interface**

- IGainCompression
- ISweptIMD
- IMSpectrum
- INoiseFigure6
- IConverter6
### DeviceNames Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a list of configured device names of the specified device type.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>value = extDevices.DeviceNames(devType)</td>
</tr>
<tr>
<td>Variable (Type) - Description</td>
<td>value (Variant) Variable to store the device names.</td>
</tr>
<tr>
<td>extDevices</td>
<td>An ExternalDevices (collection)</td>
</tr>
<tr>
<td>devType</td>
<td>(String) Device Type such as &quot;Source&quot; and &quot;Power Meter&quot;. See a complete list of valid device types.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Variant Array</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>devices = extDevices.DeviceNames(&quot;Power Meter&quot; ) 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_DeviceNames(VARIANT *names);</td>
</tr>
<tr>
<td>Interface</td>
<td>IExternalDevices2</td>
</tr>
</tbody>
</table>
**DeviceOutputPort Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the VNA port number to be used for the DUT output.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>port = mixer.DeviceOutputPort</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(<code>Type</code>) - Description</td>
</tr>
<tr>
<td><code>port</code></td>
<td>(<code>long</code>) Variable to store the VNA port number of the DUT output.</td>
</tr>
<tr>
<td><code>mixer</code></td>
<td>A Mixer (<code>object</code>)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>2</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = mixer.DeviceOutputPort</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_DeviceOutputPort(long *pVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMixer8</td>
</tr>
</tbody>
</table>
## InputLinearPowerLevel Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the input power at which Linear Gain and all S-parameters are measured.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>gca.InputLinearPowerLevel = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td></td>
<td><em>gca</em> A GainCompression <strong>(object)</strong></td>
</tr>
<tr>
<td></td>
<td><em>value</em> <strong>(double)</strong> Linear input power level in dBm. Choose a value from +30 to (-30).</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>- 25 dBm</td>
</tr>
<tr>
<td>Examples</td>
<td><code>gca.InputLinearPowerLevel = -10</code> <em>Write</em></td>
</tr>
<tr>
<td></td>
<td><code>LinPwr = gca.InputLinearPowerLevel</code> <em>Read</em></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_InputLinearPowerLevel(double* pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_InputLinearPowerLevel(double newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IGainCompression</td>
</tr>
</tbody>
</table>
**DeviceOutputPort Property**

**Description**
Read the VNA port number which is connected to the DUT Output.

Use **SetPortMap Method** to change the port mapping.

**VB Syntax**

```vb
obj.DeviceOutputPort
```

**Variable (Type) - Description**

- **obj** A Converter (object) or
- A GainCompression (object) or
- A SweptIMD (object) or
- An IMSpectrum (object)
- A NoiseFigure (object)

**Return Type**
Integer

**Default**
2

**Examples**

```vb
outPort = gca.DeviceOutputPort 'Read
```

**C++ Syntax**

```cpp
HRESULT get_DeviceOutputPort(int* pVal)
```

**Interface**
IGainCompression
ISweptIMD
IM Spectrum
INoiseFigure6
IConverter6
DeviceType Property

Description
Sets and returns the DeviceType for the external device.

VB Syntax
`extDevices.DeviceType = value`

Variable
(Type) - Description
`extDevices` An `ExternalDevice (object)`
`value` (String) Device Type.

"Source" - external source

"Power Meter" - power meter

"DC Meter" - DC voltmeter

"DC Source" - DC power supply

"Pulse Generator" - external pulse generator

Return Type
String

Default
"Unknown"

Examples
`extDevices.DeviceType = "source` Write
`value = extDevices.DeviceType` 'Read

See example program to configure PMAR device

See example program to configure External Source

C++ Syntax
`HRESULT get_DeviceType( BSTR* value);`

`HRESULT put_DeviceType( BSTR newVal);`

Interface
`IExternalDevices`
**DiffPortMatch_C Property**

**Description**
Sets the Capacitance value of the differential matching circuit.

**VB Syntax**
```vbnet
fixture.DiffPortMatch_C(portNum) = value
```

**Variable**

- **fixture** (Type) - A Fixturing (object)
- **portNum** (Integer) - Balanced (logical) port number. Choose from logical ports 1, 2, 3. Learn more about logical ports.
- **value** (Double) - Capacitance value in farads. Choose a value between \(-1E18\) to \(1E18\).

**Return Type**
Double

**Default**
0

**Examples**
```vbnet
fixture.DiffPortMatch_C(2) = 1e-6 'Write
value = fixture.DiffPortMatch_C(1) 'Read
```

**C++ Syntax**
```cpp
HRESULT get_DiffPortMatch_C( short portNum, double *pVal)
HRESULT put_DiffPortMatch_C( short portNum, double newVal)
```

**Interface**
IFixturing2
**DiffPortMatch_G Property**

**Description**: Sets the Conductance value of the differential matching circuit.

**VB Syntax**: `fixture.DiffPortMatch_G(portNum) = value`

**Variable**

- **Type** - Description
  - `fixture` A Fixturing (object)
  - `portNum` (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.
  - `value` (Double) Conductance value in siemens. Choose a value between -1E18 to 1E18.

**Return Type**: Double

**Default**: 0

**Examples**

- `fixture.DiffPortMatch_G(2) = 1e-3` 'Write
- `value = fixture.DiffPortMatch_G(1)` 'Read

**C++ Syntax**

- `HRESULT get_DiffPortMatch_G( short portNum, double *pVal)`
- `HRESULT put_DiffPortMatch_G( short portNum, double newVal)`

**Interface**: IFixturing2
## DiffPortMatch_L Property

Sets the Inductance value of the differential matching circuit.

### VB Syntax

```vb
fixture.DiffPortMatch_L(portNum) = value
```

### Variable (Type) - Description

- **fixture** (object)
- **portNum** (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.
- **value** (Double) Inductance value in henries. Choose a value between \(-1E18\) to \(1E18\).

### Return Type

Double

### Default

0

### Examples

```vb
fixture.DiffPortMatch_L(2) = 1e-3 'Write
value = fixture.DiffPortMatch_L(1) 'Read
```

### C++ Syntax

```cpp
HRESULT get_DiffPortMatch_L( short portNum, double *pVal)
HRESULT put_DiffPortMatch_L( short portNum, double newVal)
```

### Interface

IFixturing2
**DiffPortMatch_R Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the Resistance value of the differential matching circuit.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>fixture.DiffPortMatch_R(portNum) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>fixture</code></td>
<td>A Fixturing <em>(object)</em></td>
</tr>
<tr>
<td><code>portNum</code></td>
<td><em>(Integer)</em> Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Double)</em> Resistance value in ohms. Choose a value between (-1\times10^{18}) to (1\times10^{18}).</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
</tbody>
</table>
| **Examples** | `fixture.DiffPortMatch_R(2) = 1e3` *Write*  
`value = fixture.DiffPortMatch_R(1)` *Read* |
| **C++ Syntax** | `HRESULT get_DiffPortMatch_R(short portNum, double *pVal)`  
`HRESULT put_DiffPortMatch_R(short portNum, double newVal)` |
| **Interface** | IFixturing2 |
## DiffPortMatchMode Property

**Description**  
Sets the differential matching circuit type. To select a user-defined circuit, specify IN ADVANCE the 2-port touchstone filename with `DiffPortMatch_UserFilename Property`. If you do not specify the appropriate file and you select USER, an error occurs and `naNO_CIRCUIT` is automatically selected.

**VB Syntax**  
`fixture.DiffPortMatchMode(pNum) = value`

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fixture</code></td>
<td>A Fixturing (object)</td>
<td></td>
</tr>
<tr>
<td><code>pNum</code></td>
<td>(Integer)</td>
<td>Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Enum as NADiffPortMatchCircuitMode)</td>
<td>Choose from:</td>
</tr>
<tr>
<td>0 or <code>naSHUNT_L_SHUNT_C_CIRCUIT</code></td>
<td>Specifies the circuit that consists of shunt L and shunt C.</td>
<td></td>
</tr>
<tr>
<td>1 or <code>naUSER_FILE_CIRCUIT</code></td>
<td>Specifies the user-defined circuit.</td>
<td></td>
</tr>
<tr>
<td>2 or <code>naNO_CIRCUIT</code></td>
<td>Specifies no-circuit.</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**  
Enum

**Default**  
`naSHUNT_L_SHUNT_C_CIRCUIT`

**Examples**  
<table>
<thead>
<tr>
<th>Write</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fixture.DiffPortMatchMode(2) = naNO_CIRCUIT</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Read</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value = fixture.DiffPortMatchMode(1)</code></td>
</tr>
</tbody>
</table>

**C++ Syntax**  
```cpp
HRESULT get_DiffPortMatchMode( short port, tagNADiffPortMatchCircuitMode *eVal)

HRESULT put_DiffPortMatchMode( short port, tagNADiffPortMatchCircuitMode eVal)
```

**Interface**  
`IFixturing2`
**DiffPortMatchState Property**

**Description**
Turns ON or OFF 4-port differential port matching function. Must also set the fixture simulator function to ON using FixturingState Property.

**VB Syntax**
```
fixture.DiffPortMatchState = value
```

**Variable**
- **(Type)** - Description
  - `fixture` (A Fixturing (object))
  - `value` (Boolean)

**False** - Turns differential port matching OFF

**True** - Turns differential port matching ON

**Return Type**
Boolean

**Default**
False

**Examples**
```
fixture.DiffPortMatchState = False 'Write
value = fixture.DiffPortMatchState 'Read
```

**C++ Syntax**
```
HRESULT get_DiffPortMatchState( VARIANT_BOOL *pVal)
HRESULT put_DiffPortMatchState( VARIANT_BOOL newVal)
```

**Interface**
IFixturing2
**DiffPortMatchUserFilename Property**

**Description**  Specifies the 2-port touchstone file in which the information on the user-defined differential matching circuit is saved. Following this command, send **DiffPortMatchCircuit Property**. If the specified file does not exist, an error occurs when you set the type of differential matching circuit to USER.

**VB Syntax**  
```vbnet
fixture.DiffPortMatchUserFilename(pNum) = value
```

**Variable**  
- **fixture**  A Fixturing (object)
- **pNum**  (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.
- **value**  (String) Full path, file name, and extension (.s2P) of the de-embedding circuit.

Files are typically stored in "D:\".

**Return Type**  String

**Default**  Not Applicable

**Examples**  
```vbnet
fixture.DiffPortMatchUserFilename(2) = "D:\myFile.s4p"  'Write
value = fixture.DiffPortMatchUserFilename(1)  'Read
```

**C++ Syntax**  
```cpp
HRESULT get_DiffPortMatchUserFilename( short port, BSTR *bstrFile)

HRESULT put_DiffPortMatchUserFilename( short port, BSTR bstrFile)
```

**Interface**  IFixturing2
### DiffZConvPortImag Property

**Description**
Sets the imaginary part of the impedance value for the differential port impedance conversion function.

**VB Syntax**
```
fixture.DiffZConvPortImag(portNum) = value
```

**Variable (Type) - Description**
- **fixture** (A Fixturing object)
- **portNum** (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. [Learn more about logical ports.](#)
- **value** (Double) Imaginary part of the Impedance value. Choose a value between 0 and 1E18

**Return Type**
Double

**Default**
0

**Examples**
```
fixture.DiffZConvPortImag(2) = 75 'Write
value = fixture.DiffZConvPortImag(1) 'Read
```

**C++ Syntax**
```
HRESULT get_DiffZConvPortImag( short portNum, double *pVal)
HRESULT put_DiffZConvPortImag( short portNum, double newVal)
```

**Interface**
IFixturing2
**DiffZConvPortReal Property**

**Description**
Sets the real part of the impedance value for the differential port impedance conversion function.

**VB Syntax**
```
fixture.DiffZConvPortReal(portNum) = value
```

**Variable (Type) - Description**
- **fixture** (Fixturing (object))
- **portNum** (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.
- **value** (Double) Real part of the Impedance value. Choose a value between 0 and 1E18

**Return Type**
Double

**Default**
See Differential Port Z Conversion Default

**Examples**
```
fixture.DiffZConvPortReal(2) = 75 'Write
value = fixture.DiffZConvPortReal(1) 'Read
```

**C++ Syntax**
```
HRESULT get_DiffZConvPortReal( short portNum, double *pVal)
HRESULT put_DiffZConvPortReal( short portNum, double newVal)
```

**Interface**
IFixturing2
**DiffZConvPortZ0 Property**

**Description**
Sets the impedance value for the differential port impedance conversion function. Set either this single value or set the real and imaginary parts separately. The imaginary part is set to 0.0 using this command.

**VB Syntax**
```
fixture.DiffZConvPortZ0(portNum) = value
```

**Variable**
- (Type) - Description
- fixture (A Fixturing object)
- portNum (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.
- value (Double) Impedance value. Choose a value between 0 and 1E18

**Return Type**
Double

**Default**
See Differential Port Z Conversion Default

**Examples**
```
fixture.DiffZConvPortZ0(2) = 75 'Write
value = fixture.DiffZConvPortZ0(1) 'Read
```

**C++ Syntax**
```
HRESULT get_DiffZConvPortZ0( short portNum, double *pVal)
HRESULT put_DiffZConvPortZ0( short portNum, double newVal)
```

**Interface**
IFixturing2
**DiffZConvState Property**

**Description**
Turns ON or OFF 4-port differential impedance conversion function. Must also set the fixture simulator function to ON using `FixturingState Property`.

**VB Syntax**
```vbnet
fixture.DiffZConvState = value
```

**Variable**
- **(Type) - Description**
  - `fixture` - A `Fixturing` (object)
  - `value` - (Boolean)
    - **False** - Turns differential impedance conversion OFF.
    - **True** - Turns differential impedance conversion ON.

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
fixture.DiffZConvState = False 'Write
value = fixture.DiffZConvState 'Read
```

**C++ Syntax**
```cpp
HRESULT get_DiffZConvState( VARIANT_BOOL *pVal)
HRESULT put_DiffZConvState( VARIANT_BOOL newVal)
```

**Interface**
IFixturing2
### DimensionCatalog Property

**Description**  
Read the names of source domains in the multi-dimensional sweep whose state is ON and whose dimension order is the specified dimension order.

**VB Syntax**  
```vb
names = md.DimensionCatalog(dim)
```

**Variable**  
- **Type**: Variant  
- **Description**: Variable to store the returned source domain names.
- **md**: A `MultiDimensionalSweep (object)` which belongs to a SA channel.
- **dim**: A `long` Dimension order. Choose an integer value of 1 or higher.

**Return Type**  
String. Names are separated by commas.

**Default**  
Not Applicable

**Examples**  
```vb
names = md.DimensionCatalog(3) 'Read
```

**C++ Syntax**  
```c++
HRESULT get_DimensionCatalog (long dim, VARIANT * pValue);
```

**Interface**  
IMultiDimensionalSweep
DimensionCount Property

Description: Read the highest dimension order in the multi-dimensional sweep.

VB Syntax:
```
count = md.DimensionCount
```

Variable (Type) - Description:
- **count** (long): Variable to store the returned dimension order value.
- **md**: A MultiDimensionalSweep (object) which belongs to a SA channel.

Return Type: long

Default: 0

Examples:
```
count = md.DimensionCount 'Read
```

C++ Syntax:
```
HRESULT get_DimensionCount(long* count);
```

Interface: IMultiDimensionalSweep
DimensionPointCount Property

Description
Set and read the point count for the specified dimension order in the multi-dimensional sweep.

VB Syntax
```
md.DimensionPointCount (dim) = value
```

Variable (Type) - Description
- `md`: A `MultiDimensionalSweep (object)` which belongs to a SA channel.
- `dim`: (long) Dimension order. Choose an integer value of 1 or higher.
- `value`: (long) Point count. Choose an integer value of 1 or higher.

Return Type
long

Default
1

Examples
```
md.DimensionPointCount (2) = 3 'Write
value = md.DimensionPointCount(4) 'Read
```

C++ Syntax
```
HRESULT put_DimensionPointCount (long dim, long value);
HRESULT get_DimensionPointCount (long dim, long* value);
```

Interface
IMultiDimensionalSweep
### DimensionRepeatCount Property

**Description**
Set and read the repeat count for the specified dimension order in the multi-dimensional sweep.

**VB Syntax**
```
md.DimensionRepeatCount (dim) = value
```

**Variable (Type) - Description**
- **md** (MultiDimensionalSweep (object)) which belongs to a SA channel.
- **dim** (long) Dimension order. Choose an integer value of 1 or higher.
- **value** (long) Repeat count. Choose an integer value of 1 or higher.

**Return Type**
long

**Default**
1

**Examples**
```
md.DimensionRepeatCount (2) = 1  'Write
value = md.DimensionRepeatCount(4) 'Read
```

**C++ Syntax**
```
HRESULT put_DIMENSIONREPEATCOUNT (long dim, long value);

HRESULT get_DIMENSIONREPEATCOUNT (long dim, long* value);
```

**Interface**
IMultiDimensionalSweep
### DirectoryPath Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a the path name to the systems documents folders on the VNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = cap.DirectoryPath (enum)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><em>value</em></td>
<td>(String) - Variable to store the returned directory path.</td>
</tr>
<tr>
<td><em>cap</em></td>
<td>A Capabilities <em>(object)</em></td>
</tr>
<tr>
<td><em>enum</em></td>
<td>Type of file. Choose from:</td>
</tr>
<tr>
<td>0</td>
<td><strong>naDirectoryState</strong> - This is the location for the storage of state files.</td>
</tr>
<tr>
<td>1</td>
<td><strong>naDirectoryApplication</strong> - This is the location of the VNA firmware executable files.</td>
</tr>
<tr>
<td>2</td>
<td><strong>naDirectorySupport</strong> - This is the location of private support files for the VNA firmware.</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = cap.DirectoryPath(naDirectoryState)</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_DirectoryPath(enum DirectoryPathType directoryPathType,BSTR* path);</td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities10</td>
</tr>
</tbody>
</table>
## DiscreteFrequencies Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the list of discrete frequencies corresponding to the powers returned by <code>DiscreteGetMaxPowerArray</code> or <code>DiscreteGetMinPowerArray</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>powerRange.DiscreteFrequencies = discreteFreqs</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>powerRange</code></td>
<td>A <code>PowerRange (object)</code></td>
</tr>
<tr>
<td><code>discreteFreqs</code></td>
<td><em>(Variant array)</em> Variable to store the list of discrete frequencies.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant array</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><code>discreteFreqs = Array(1e9,2e9,3e9,4e9)</code></td>
<td><em>Write</em></td>
</tr>
<tr>
<td><code>powerRange.DiscreteFrequencies = discreteFreqs</code></td>
<td><em>Read</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_DiscreteFrequencies(VARIANT* Freq);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_DiscreteFrequencies(VARIANT Freq);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPowreRange</td>
</tr>
</tbody>
</table>
Format Property

Description: Sets or returns the display format of the measurement.

VB Syntax: `meas.Format = value`

Variable (Type) - Description:

- `meas`: A Measurement (object)
- `value`: (enum NADataFormat) - Choose from:
  - 0 - naDataFormat_LinMag
  - 1 - naDataFormat_LogMag
  - 2 - naDataFormat_Phase
  - 3 - naDataFormat_Polar
  - 4 - naDataFormat_Smith
  - 5 - naDataFormat_Delay
  - 6 - naDataFormat_Real
  - 7 - naDataFormat_Imaginary
  - 8 - naDataFormat_SWR
  - 9 - naDataFormat_PhaseUnwrapped
  - 10 - naDataFormat_InverseSmith
  - 11 - naDataFormat_Kelvin
  - 12 - naDataFormat_Fahrenheit
  - 13 - naDataFormat_Celsius
  - 14 - naDataFormat_PhasePositive

Return Type: Long Integer

Default: 1 - naDataFormat_LogMag

Examples:

- `meas.Format = naDataFormat_Real 'Write`
- `fmt = meas.Format 'Read`
C++ Syntax
HRESULT get_Format(tagDataFormat *pVal)
HRESULT put_Format(tagDataFormat newVal)

Interface
IMeasurement
## DisplayAutomationErrors Property

**Description**
Enables or disables automation error messages from being displayed on the screen.

**VB Syntax**
```vbnet
app.DisplayAutomationErrors = value
```

**Variable**
- `(Type)` - Description
  - `app` An Application (object)
  - `value` (Boolean)

- **True** allows error to show on display,
- **False** turns error off from display.

**Return Type**
Boolean

**Default**
True

**Examples**
```vbnet
Dim app As Application
Set app = New Application
app.DisplayAutomationErrors = False 'Turns off display
print app.DisplayAutomationErrors 'prints False
```

**C++ Syntax**
```cpp
HRESULT get_DisplayAutomationErrors(VARIANT_BOOL * Val);
HRESULT put_DisplayAutomationErrors(VARIANT_BOOL Val);
```

**Interface**
IApplication2
DisplayDomain Property

Description
Set and read the X-axis domain type.

VB Syntax
HotS22.DisplayDomain = value

Variable
(Type) - Description
HotS22 A ActiveParametersApp (object)
value (Enum as NAAxisDomainType) - Choose from:

- 0 - naAxisFrequency  Sets a linear frequency sweep that is displayed on a standard grid with ten equal horizontal divisions.
- 1 - naAxisPower  Sets a power sweep at a single specified frequency.
- 2 - naAxisPhase  Sets a phase sweep.
- 3 - naAxisDCValue  Sets value of dc source.
- 4 - naAxisPoints  Sets the number of phase points.
- 5 - TypeNum

Return Type
Enum

Default
naAxisFrequency

Examples
HotS22.DisplayDomain = naAxisFrequency 'Write
value = HotS22.DisplayDomain 'Read

C++ Syntax
HRESULT get_DisplayDomain(tagNAAxisDomainType* value)
HRESULT put_DisplayDomain(tagNAAxisDomainType value)

Interface
IAActiveChannelSettings
### DisplayGlobalPassFail Property

**Description**  
Shows or hides the dialog which displays global pass/fail results. Learn more about Global Pass/Fail.

**VB Syntax**  
```vbnet
app.DisplayGlobalPassFail = value
```

**Variable**  
(Type) - Description

- **app** An Application (object)
- **value** (Boolean)
  
  - **True** - displays the pass/fail dialog.
  - **False** - hides the pass/fail dialog.

**Return Type**  
Boolean

**Default**  
False

**Examples**
```vbnet
Dim app As Application
Set app = New Application
app.DisplayGlobalPassFail = true  'shows dialog
```

**C++ Syntax**  
```csharp
HRESULT get_DisplayGlobalPassFail(VARIANT_BOOL * Val);  
HRESULT put_DisplayGlobalPassFail(VARIANT_BOOL Val);
```

**Interface**  
IAplication6
## DisplayInputPower Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read a fixed trace input power level.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>HotS22.DisplayInputPower = level</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>HotS22</code></td>
<td>A <code>ActiveParametersApp (object)</code></td>
</tr>
<tr>
<td><code>level</code></td>
<td><em>(Double)</em> - Input power level. Choose a value from start power to stop power.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Start power in a power sweep</td>
</tr>
<tr>
<td>Examples</td>
<td><code>HotS22.DisplayInputPower = 0</code> <em>Write</em></td>
</tr>
<tr>
<td></td>
<td><code>level = HotS22.DisplayInputPower</code> <em>Read</em></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_DisplayInputPower(double* level)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_DisplayInputPower(double level)</td>
</tr>
<tr>
<td>Interface</td>
<td>IActiveChannelSettings</td>
</tr>
</tbody>
</table>
### DisplayInterpolationState Property

**Description**
Sets whether or not interpolation is on for display. Frequency, power, and phase X axis are supported for display. Interpolation may be applied in the trace.

**VB Syntax**

```
HotS22.DisplayInterpolationState = value
```

**Variable**
- **Type** - Description
  - *HotS22*
    - A `ActiveParametersApp (object)`
  - *value*
    - (boolean) - Choose from:
      - **True** - Turns interpolation ON
      - **False** - Turns interpolation OFF

**Return Type**
Boolean

**Default**
False

**Examples**

```
HotS22.DisplayInterpolationState = False
value = HotS22.DisplayInterpolationState 'Read
```

**C++ Syntax**

```
HRESULT get_DisplayInterpolationState(VARIANT_BOOL* value);
HRESULT put_DisplayInterpolationState(VARIANT_BOOL value);
```

**Interface**
IACTiveChannelSettings
## DisplayRange Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the range to be displayed on the VNA x-axis. All traces in the channel have this same x-axis scaling.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>FOM.DisplayRange = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(object)</code> - Description</td>
</tr>
<tr>
<td><code>object</code></td>
<td>An FOM (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(string) - Range to be displayed on the VNA x-axis. Case insensitive.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>&quot;Receivers&quot;</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>fom.DisplayRange = &quot;Source&quot;</code> 'sets the x-axis to the frequency range of &quot;source&quot;'</td>
</tr>
<tr>
<td><code>disprange = fom.DisplayRange</code> 'Read'</td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_DisplayRange(BSTR *pDspRange)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_DisplayRange(BSTR pDspRange)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFOM</td>
</tr>
</tbody>
</table>
**Distance Property**

**Description**  Set or query marker distance on a time domain trace.

The Write command moves the marker to the specified distance value. Once moved, you can read the Y axis value or read the X-axis time value. (Distance is calculated from the X-axis time value.)

The Read command reads the distance of the marker.

If the marker is set as delta, the WRITE and READ data is relative to the reference marker.

**VB Syntax**  

```vbnet
mark.Distance = value
```

**Variable**  

- **mark**  A Marker (object)
- **value**  (double) - Marker distance in the unit of measure specified with DistanceMarkerUnit Property

**Return Type**  Double

**Default**  Not Applicable

**Examples**  

```vbnet
mark.Distance = 3e9 'Write
XVal = mark.Distance 'Read
```

**C++ Syntax**  

```cpp
HRESULT get_Distance(double *pVal);
HRESULT put_Distance(double newVal);
```

**Interface**  IMarker2
About Distance Marker Settings

DistanceMarkerMode Property

Description: Specifies the measurement type in order to determine the correct marker distance.

- Select Auto for S-Parameter measurements.
- Select Reflection or Transmission for arbitrary ratio or unratioed measurements.

This setting affects the display of all markers for only the active measurement.

VB Syntax: `trans.DistanceMarkerMode = value`

Variable: (Type) - Description

- `trans`: A Transform (object)
- `value`: (enum As NADistanceMarkerMode) - Choose from:
  - 0 - naDistanceMarkerModeAuto
  - 1 - naDistanceMarkerModeReflection
  - 2 - naDistanceMarkerModeTransmission

Return Type: Enum

Default: 0 - naDistanceMarkerModeAuto

Examples: `trans.DistanceMarkerMode = naDistanceMarkerModeReflection` 'Write

`DMM = trans.DistanceMarkerMode` 'Read'

C++ Syntax: HRESULT get_DistanceMarkerMode(tagNADistanceMarkerMode *pVal);

HRESULT put_DistanceMarkerMode(tagNADistanceMarkerMode newVal);

Interface: ITransform2
DistanceMarkerUnit Property

Description: Specifies the unit of measure for the display of marker distance values. This setting affects the display of ALL markers for only the ACTIVE measurement (unless Distance Maker Units are coupled using CoupledParameters Property.

VB Syntax: `trans.DistanceMarkerUnit = value`

Variable (Type) - Description

- `trans`: A Transform (object)
- `value`: (Enum As NADistanceMarkerUnit) - Distance Marker Units. Choose from
  - 0 - naDistanceMarkerUnitMeter
  - 1 - naDistanceMarkerUnitFeet
  - 2 - naDistanceMarkerUnitInch

Return Type: Enum

Default: 0 - naDistanceMarkerUnitMeter

Examples:

- `trans.DistanceMarkerUnit = naDistanceMarkerUnitFeet` 'sets the
- `U = trans.DistanceMarkerUnit` 'Read

C++ Syntax:

- `HRESULT get_DistanceMarkerUnit(tagNADistanceMarkerUnit *pVal);`
- `HRESULT put_DistanceMarkerUnit(tagNADistanceMarkerUnit newVal);`

Interface: ITransform2
## Divisor Property

### Description
Sets and returns the Divisor value to be used when coupling this range to the primary range.

This setting is valid only if the specified range is coupled to the primary range.

### VB Syntax
```vbnet
FOMRange.Divisor = value
```

### Variable (Type) - Description
- **object**: An FOMRange
- **value**: (Double) - Divisor value.-(Unitless)

### Return Type
Double

### Default
0

### Examples
- ```vbnet
  fomRange.Divisor = 0.5 'Write
  Div = fomRange.Divisor 'Read
```

### C++ Syntax
- `HRESULT get_Divisor(double *pVal)`
- `HRESULT put_Divisor(double *pVal)`

### Interface
- IFOMRange
**Do1PortEcal Property - Superseded**

<table>
<thead>
<tr>
<th>Description</th>
<th>Note: This command is replaced with CalKitType which sets the ECal module or mechanical cal kit. Specify ECal or Mechanical calibration for the mixer characterization portion of a VMC calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>VMC.Do1PortEcal = bool</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>VMC</td>
<td>VMCType (object)</td>
</tr>
<tr>
<td>bool</td>
<td>(Boolean)</td>
</tr>
<tr>
<td>True</td>
<td>ECAL</td>
</tr>
<tr>
<td>False</td>
<td>Mechanical</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
<tr>
<td>Examples</td>
<td>value = VMC.Do1PortEcal</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT put_Do1PortEcal(VARIANT_BOOL bDoEcal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_Do1PortEcal(VARIANT_BOOL *bDoEcal);</td>
</tr>
<tr>
<td>Interface</td>
<td>VMCType</td>
</tr>
</tbody>
</table>
**Do2PortEcal Property - Superseded**

**Description**

*Note:* This command is replaced with `CalKitType` which sets the ECal module or mechanical cal kit.

Specify ECAL or Mechanical calibration. For VMC, this selection only applies to the 2-port calibration portion. For mixer characterization (VMC), use `Do1PortEcal` Property.

**VB Syntax**

```
object.Do2PortEcal = bool
```

**Variable**

`(Type) - Description`

- `object` SMCType (object) or VMCType (object)
- `bool` (Boolean)

  - `True` - ECAL
  - `False` - Mechanical

**Return Type**

Boolean

**Default**

False

**Examples**

```
value = VMC.Do2PortEcal
```

**C++ Syntax**

```
HRESULT put_Do2PortEcal(VARIANT_BOOL bDoEcal);

HRESULT get_Do2PortEcal(VARIANT_BOOL *bDoEcal);
```

**Interface**

SMCType

VMCType
Domain Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the domain (frequency, time, power, phase) of the measurement. To understand how this property is useful, see IMeasurement2 Interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = meas.Domain</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(Enum as NADomainType) - variable to store the returned value <code>0 - naDomainFrequency</code></td>
</tr>
<tr>
<td></td>
<td><code>1 - naDomainTime</code></td>
</tr>
<tr>
<td></td>
<td><code>2 - naDomainPower</code></td>
</tr>
<tr>
<td></td>
<td><code>4 - naDomainPhase</code></td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum as NADomainType</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><strong>Print</strong> meas.Domain <strong>'prints the value of the domain enum</strong>*</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_Domain(tagNADomainType * Val);</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
### DrawLimitLinesInRed Property

**Write/Read**

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return whether to draw limits lines in Red or the trace color.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pref.DrawLimitLinesInRed = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>pref</code></td>
<td>A Preferences (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean) - Choose from:</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>All Limit lines are drawn in Red.</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>Limit lines are drawn the same color as the trace.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pref.DrawLimitLinesInRed = True 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = pref.DrawLimitLinesInRed 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_DrawLimitLinesInRed (VARIANT_BOOL* preference);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_DrawLimitLinesInRed (VARIANT_BOOL val)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPreferences15</td>
</tr>
</tbody>
</table>
### Driver Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the external device driver (model).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>extDevices.Driver = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>extDevices</code></td>
<td>An ExternalDevice <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(String) External device driver (model). Choose from the following:</td>
</tr>
<tr>
<td></td>
<td>&quot;AGPM&quot; for all power meters.</td>
</tr>
<tr>
<td></td>
<td>&quot;AGPULSEGEN&quot; for supported pulse generators.</td>
</tr>
<tr>
<td></td>
<td>&quot;DCSource&quot; for all supported DC Sources</td>
</tr>
<tr>
<td></td>
<td>&quot;DCMeter&quot; for all supported DC Meters</td>
</tr>
</tbody>
</table>

See a list of supported external source drivers.

| **Return Type** | String |
| **Default** | Not Applicable |

**Examples**

```vbnet
extDevices.Driver = "AGESG" 'Write
value = extDevices.Driver 'Read
See example program to configure PMAR device
See example program to configure External Source
```

**C++ Syntax**

```cpp
HRESULT get_Driver( BSTR* value);
HRESULT put_Driver( BSTR newVal);
```

**Interface**

IExternalDevices
**DspFpgaRevision Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the DSP FPGA Revision number that is visible in the Help, About Network Analyzer dialog box. Learn more.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.DspFpgaRevision</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(String) - Variable to store the returned DSP revision number.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.DspFpgaRevision</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_DspFpgaRevision(BSTR *value);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities6</td>
</tr>
</tbody>
</table>
# DspRevision Property

Returns the DSP Revision number that is visible in the Help, About Network Analyzer dialog box. Learn more.

### VB Syntax

```
value = cap.DspRevision
```

### Variable (Type) - Description

| value  | (String) - Variable to store the returned DSP revision number. |
| cap    | A Capabilities (object) |

### Return Type

String

### Default

Not Applicable

### Examples

```
value = cap.DspRevision 'Read
```

### C++ Syntax

```
HRESULT get_DspRevision(BSTR *value);
```

### Interface

ICapabilities6
**DUTTopology Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the device type for the balanced measurement. To map the device type logical ports to the VNA physical ports, use the <code>SetCustomDUTTopology</code> command.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>balTopology.DUTTopology = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>balTopology</code></td>
<td>A BalancedTopology (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum NADUTTopology) - Choose either:</td>
</tr>
<tr>
<td></td>
<td>0 naSEBal: Single-Ended - Balanced topology</td>
</tr>
<tr>
<td></td>
<td>1 naSESEBal: Single-Ended - Single-Ended - Balanced topology</td>
</tr>
<tr>
<td></td>
<td>2 naBalBal: Balanced - Balanced topology</td>
</tr>
<tr>
<td></td>
<td>3 naBalSE: Balanced - Single-ended topology</td>
</tr>
<tr>
<td></td>
<td>4 naBal: Single ended balanced</td>
</tr>
<tr>
<td></td>
<td>5 naCustom: Custom</td>
</tr>
<tr>
<td></td>
<td>6 naBalSESE: Balanced - Single-ended - Single-ended topology</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum as NADUTTopology</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>naSEBal</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>balTop.DUTTopology = naSESEBal 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>DutTop = balTop.DUTTopology 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_DUTTopology(tagNADUTTopology* pVal)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_DUTTopology(tagNADUTTopology newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IBalancedTopology</td>
</tr>
</tbody>
</table>
Write/Read

DwellTime Property

Description: Sets or returns the dwell time at the start of each sweep point for all measurements in a channel. Dwell time is only available with `Chan.SweepGenerationMode = naSteppedSweep` (not `naAnalogSweep`).

Sets or returns the dwell time of a specified sweep segment.

VB Syntax: `object.DwellTime = value`

Variable (Type) - Description

- `object`: A Channel (object)
- `or` CalSet (object) - Read-only property
- `or` Segment (object)

- `value`: (double) - Dwell Time in seconds. Choose any number between 0 and 86400. Note that 0 is equivalent to no dwell time.

Return Type: Double

Default: 0

Examples:

- `chan.DwellTime = 3e-3 'sets the dwell time for the channel -Write`
- `segs(3).DwellTime = 5e-3 'sets the dwell time of segment 3 -Write`
- `dwell = chan.DwellTime 'Read`

C++ Syntax: `HRESULT get_DwellTime(double *pVal)`

- `HRESULT put_DwellTime(double newVal)`

Interface: IChannel

ISegment

|CalSet3
# DwellPerPoint Property

**Description**  
Sets and returns the amount of time the VNA should wait after for an external source to settle before making a measurement at each data point.

**VB Syntax**  
`extSource.DwellPerPoint = value`

**Variable**  
- `chan` An ExternalSource (object)
- `value` (Double) Dwell time in milliseconds.

**Return Type**  
Double

**Default**  
3

**Examples**  
- `extSource.DwellPerPoint = 10 'Write`
- `dpp = extSource.DwellPerPoint 'Read`

**C++ Syntax**  
- `HRESULT get_DwellPerPoint (tagNAExtDevDwellPerPoint *pValue)`
- `HRESULT put_DwellPerPoint (tagNAExtDevDwellPerPoint newVal)`

**Interface**  
IExternalSource
**Write/Read**

**ECALCharacterization Property - Superseded**

<table>
<thead>
<tr>
<th>Description</th>
<th>Note: This command is replaced with CalKitType which sets the ECal module and User Characterization.</th>
</tr>
</thead>
</table>

Specifies the characterization data within an ECal module to be used for the SMC calibration.

Learn more about ECal User Characterization.

**VB Syntax**

```
SMC.ECALCharacterization(mod) = value
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMC</td>
<td>SMCType (object)</td>
<td></td>
</tr>
<tr>
<td>module</td>
<td>1 - ECal module</td>
<td></td>
</tr>
<tr>
<td>value</td>
<td>(Long) – Characterization data within the ECal module to be used for ECal operations. Choose from:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0 – Factory Characterization</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 – UserCharacterization1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 – UserCharacterization2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 – UserCharacterization3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 – UserCharacterization4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 – UserCharacterization5</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Long</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>0 - Factory Characterization</td>
</tr>
</tbody>
</table>

**Examples**

```
SMC.ECALCharacterization(1) = 2
```

**C++ Syntax**

```c++
HRESULT put_ECALCharacterization( long moduleNumber, long characterization);

HRESULT get_ECALCharacterization( long moduleNumber, long* characterization);
```

**Interface**

ISMCType
### ECALCharacterization Property - Superseded

**Description**

Note: This command is replaced with `CalKitType` which sets the ECal module and User Characterization.

Specifies the characterization data within an ECal module to be used, and the portion of the VMC calibration.

Learn more about ECal User Characterization.

**VB Syntax**

```
VMC.ECALCharacterization (module, port) = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>VMC</code></td>
<td>VMCTYPE (object)</td>
</tr>
<tr>
<td><code>module</code></td>
<td>(long integer)</td>
</tr>
<tr>
<td>1</td>
<td>- ECAL module</td>
</tr>
<tr>
<td><code>port</code></td>
<td>(boolean)</td>
</tr>
<tr>
<td><code>True</code></td>
<td>- 2-port calibration portion of the VMC</td>
</tr>
<tr>
<td><code>False</code></td>
<td>- 1-port (mixer characterization portion of the VMC cal)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) – Characterization data within the ECal module to be used for ECal operations. Choose from:</td>
</tr>
<tr>
<td>0</td>
<td>– Factory Characterization</td>
</tr>
<tr>
<td>1</td>
<td>– UserCharacterization1</td>
</tr>
<tr>
<td>2</td>
<td>– UserCharacterization2</td>
</tr>
<tr>
<td>3</td>
<td>– UserCharacterization3</td>
</tr>
<tr>
<td>4</td>
<td>– UserCharacterization4</td>
</tr>
<tr>
<td>5</td>
<td>– UserCharacterization5</td>
</tr>
</tbody>
</table>

**Return Type**

Long

**Default**

0 - Factory Characterization

**Examples**

```
VMC.ECALCharacterization (1,True) = 4
```
C++ Syntax

HRESULT put_ECALCharacterization( long moduleNumber, long characterization);

HRESULT get_ECALCharacterization( long moduleNumber, long* characterization);

Interface

IVMCType
**ECALCharacterizationEx Property**

**Description**
This property replaces ECALCharacterization Property.

Specifies the characterization data within an ECal module to be used for the calibration.

Learn more about ECal User Characterization.

**VB Syntax**
```
cal.ECALCharacterizationEx (module) = value
```

**Variable**
- **cal** (Calibrator (object))
- **module** (long integer) Optional argument. ECal module.

Choose from modules 1 through 8

Use IsECALModuleFoundEx to determine the number of modules connected to the VNA

Use GetECALModuleInfoEx to returns the model and serial number of each module.

**value** (Long) – Characterization data within the ECal module to be used for ECal operations. Choose from:

- 0 – Factory Characterization
- 1 – UserCharacterization1
- 2 – UserCharacterization2
- ..and so forth up to...
- 12 – UserCharacterization12

**Return Type**
Long

**Default**
0 - Factory Characterization

**Examples**
```
cal.ECALCharacterizationEx (4) = 2
```

**C++ Syntax**
```
HRESULT put_ECALCharacterizationEx( long moduleNumber, long characterization);

HRESULT get_ECALCharacterizationEx( long moduleNumber, long* characterization);
```
Interface  ICalibrator4
## ECalID Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Selects the model and serial number of the ECal module to be characterized. This command does NOT set the model and serial number of the ECal module.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>userChar.ECalID = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>userChar</code></td>
<td>An ECalUserCharacterizer Object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(String) Model and serial number of the ECal module to be characterized.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>&quot;&quot; (Empty String)</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>userChar.ECalID = &quot;N4433A,00001&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT put_ECalID(BSTR id);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IECalUserCharacterizer</td>
</tr>
</tbody>
</table>
ECALIsolation Property

**Description**

Note: The inherent isolation of the VNA is better than that attained with this command. ONLY use this command when using an external test set, and ONLY using a 8509x ECal module.

Specifies whether the acquisition of the ECal calibration should include isolation or not.

**VB Syntax**

```vbnet
cal.ECALIsolation = value
```

**Variable**

- **cal** (Type) - Description
  - A Calibrator (object)
- **value** (boolean)
  - False - Exclude Isolation
  - True - Include Isolation

**Return Type**

Boolean

**Default**

False

**Examples**

```vbnet
Dim oPNA as AgilentPNA835x.Application
Dim oCal as Calibrator
Set oPNA = CreateObject("AgilentPNA835x.Application", "MachineName")
Set oCal = oPNA.ActiveChannel.Calibrator
' Uncomment the following line to have the cal include isolation
' oCal.ECALIsolation = True
' Uncomment the following line to have the cal omit isolation
' oCal.ECALIsolation = False
oCal.DoECAL2Port ' Do the cal
```

**C++ Syntax**

```cpp
HRESULT put_ECALIsolation ( VARIANT_BOOL bIsolationState );

HRESULT get_ECALIsolation ( VARIANT_BOOL *bIsolationState );
```

**Interface**

Calibrator
Read-only

**ECALModuleNumberList Property**

**Description**
Returns a list of index numbers to be used for referring to the ECal modules that are currently attached to the VNA.

**VB Syntax**

```vbnet
clist = cal.ECALModuleNumberList
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>clist</code></td>
<td>Variable to store the returned list of index numbers.</td>
</tr>
<tr>
<td><code>cal</code></td>
<td>Calibrator (object)</td>
</tr>
</tbody>
</table>

**Return Type**
Variant

**Default**
Not Applicable

**Examples**

```vbnet
clist = cal.ECALModuleNumberList
'If 2 modules are attached to the VNA
'then the returned list will be:
1,2
```

**C++ Syntax**

```csharp
HRESULT get_ECALModuleNumberList(VARIANT *modules);
```

**Interface**
ICalibrator6
EcalOrientation Property

Description
Specifies which port of the ECal module is connected to which port of the VNA when the AutoOrient property = False.

VB Syntax
SMC.EcalOrientation (mod) = value

Variable (Type) - Description
SMC SMCType (object)
mod (Long)

1 - Use ECAL Module for the calibration.

value (string) - Format this parameter in the following manner:
Aw,Bx,Cy,Dz

where

- A, B, C, and D are literal ports on the ECAL module
- w,x,y, and z are substituted for VNA port numbers to which the ECAL module port is connected.

Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECal module with

- port A connected to VNA port 2
- port B connected to VNA port 3
- port C not connected
- port D connected to VNA port 1

the string would be: A2,B3,D1

Return Type String
Default "A1,B2"

Examples
SMC.EcalOrientation (1) = "A2,B1"
**C++ Syntax**

HRESULT put_EcalOrientation(long lModuleNum, BSTR orientation);
HRESULT get_EcalOrientation(long lModuleNum, BSTR *orientation);

**Interface**

SMCType
Read/Write

**EcalOrientation1Port Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>For Mixer Characterization ONLY Specifies which port of the ECAL module is connected to which port of the VNA for the Do1PortECAL property when the AutoOrient property = False.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>VMC.EcalOrientation1Port (mod) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>VMC</code></td>
<td>VMCTYPE (object)</td>
</tr>
<tr>
<td><code>mod</code></td>
<td>(Long)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(string) - Choose from:</td>
</tr>
<tr>
<td></td>
<td>&quot;A1&quot; - ECAL module port A is connected to VNA port 1</td>
</tr>
<tr>
<td></td>
<td>&quot;B1&quot; - ECAL module port A is connected to VNA port 1</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>&quot;A1&quot;</td>
</tr>
<tr>
<td></td>
<td>If anything other than port 1 is specified, &quot;B1&quot; will be used. For example, if &quot;A2&quot; is specified, &quot;B1&quot; is used.</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>VMC.EcalOrientation1Port(1) = &quot;B1&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT put_EcalOrientation1Port(long lModuleNum, BSTR orientation);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_EcalOrientation1Port(long lModuleNum, BSTR)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>VMCTYPE</td>
</tr>
</tbody>
</table>
### EcalOrientation2Port Property

**Description**
Specifies which port of the ECal module is connected to which port of the VNA for the Do2PortECAL property when the AutoOrient property = False.

**VB Syntax**

```
VMC.EcalOrientation2Port (mod) = value
```

**Variable**

**VMC VMCType (object)**

**mod**
(Long) Module being used for the calibration.
Choose from 1 or 2.

**value**
(string) - Format this parameter in the following manner:

Aw,Bx,Cy,Dz

where

- A, B, C, and D are literal ports on the ECAL module
- w,x,y, and z are substituted for VNA port numbers to which the ECAL module port is connected.

Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECal module with

- port A connected to VNA port 2
- port B connected to VNA port 3
- port C not connected
- port D connected to VNA port 1

the string would be: A2,B3,D1

**Return Type**
String

**Default**
"A1,B2"

**Examples**

```
VMC.EcalOrientation1Port(1) = "A2,B1"
```
C++ Syntax
HRESULT put_EcalOrientation2Port(long lModuleNum, BSTR orientation);
HRESULT get_EcalOrientation2Port(long lModuleNum, BSTR *orientation);

Interface
VMCTYPE
**ECALPortMapEx Property**

**Description**

This property replaces ECALPortMap Property

Specifies which ports of the ECal module are connected to which ports of the VNA for the DoECAL1PortEx and DoECAL2PortEx methods when the OrientECALModule property = False.

This setting remains until the VNA is restarted or this command is sent again.

**Note:** For guided calibrations where Orient is OFF and the same ECal module is used in more than one Connection Step, you are not allowed to specify how the ECal module is connected. Instead, the VNA determines the orientation. The VNA does not verify that you made the connection properly.

This command, and OrientECALModule_Property, can be used to perform ECal orientation using the Guided Calibration interface.

**VB Syntax**

`cal.ECALPortMapEx (module) = value`

**Variable (Type) - Description**

- `cal` A Calibrator (object)
- `module` (long integer) Optional argument. ECal module.

Choose from modules 1 through 8

Use IsECALModuleFoundEx to determine the number of modules connected to the VNA

Use GetECALModuleInfoEx to return the model and serial number of each module.

- `value` (string) - Format this parameter in the following manner:

  Aw,Bx,Cy,Dz

  where

  - A, B, C, and D are literal ports on the ECAL module
  - w,x,y, and z are substituted for VNA port numbers to which the ECAL module port is connected.

  Ports of the module which are not used are omitted from the string.
For example, on a 4-port ECal module with

- port A connected to VNA port 2
- port B connected to VNA port 3
- port C not connected
- port D connected to VNA port 1

the string would be: A2,B3,D1

DoECAL1PortEx or DoECAL2PortEx methods will fail if the port numbers passed to those methods are not in the string of this property and OrientECALModule property = False.

**Return Type**  String

**Default**  Not Applicable

**Examples**

```vba
Dim cal As Calibrator
Dim sPortMap As String
Set cal = PNAapp.ActiveChannel.Calibrator
cal.ECALPortMapEx = "a2,b1" 'Write
sPortMap = cal.ECALPortMap 'Read
```

**C++ Syntax**

```cpp
HRESULT put_ECALPortMapEx( long moduleNumber, BSTR strPortMap);
HRESULT get_ECALPortMapEx( long moduleNumber, BSTR *strPortMap);
```

**Interface**  ICalibrator4
## ElecDelayMedium Property

**Description**
Sets or returns the electrical delay medium.

**VB Syntax**
`meas.ElecDelayMedium = value`

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>meas</em></td>
<td>A Measurement <em>(object)</em></td>
</tr>
<tr>
<td><em>value</em></td>
<td><em>(enum NACalStandardMedium)</em> choose from</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>naCoax</td>
</tr>
<tr>
<td>1</td>
<td>naWaveGuide</td>
</tr>
</tbody>
</table>

**Return Type**
NACalStandardMedium

**Default**
Not Applicable

**Examples**

```
Print meas.ElecDelayMedium 'prints the value of the electrical delay medium
```

**C++ Syntax**

```
HRESULT get_ElecDelayMedium(tagNACalStandardMedium *pVal);
HRESULT put_ElecDelayMedium(tagNACalStandardMedium newVal);
```

**Interface**
IMeasurement2
## ElecDistanceDelay Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the electrical delay in physical length (distance) for the selected measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>meas.ElecDistanceDelay = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td></td>
<td><em>meas</em> - A Measurement <strong>(object)</strong></td>
</tr>
<tr>
<td></td>
<td><em>value</em> - Electrical Delay in distance.</td>
</tr>
<tr>
<td></td>
<td>Set units using ElecDistanceDelayUnit.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
<tr>
<td>VB Examples</td>
<td><code>meas.ElecDistanceDelay = 1e-3</code> <em>Write</em></td>
</tr>
<tr>
<td>C# Examples</td>
<td><code>Meas.ElecDistanceDelay = 1e-3</code> <em>Write</em></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_ElecDistanceDelay(VARIANT *pVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement11</td>
</tr>
</tbody>
</table>

Examples

```vbnet
edelay = meas.ElecDistanceDelay 'Read
```

```csharp
// This property returns an object, and the object must be cast to a double to access the value.
Edelay = (double)meas.ElecDistanceDelay 'read
```

```cpp
HRESULT get_ElecDistanceDelay(VARIANT *pVal)
HRESULT put_ElecDistanceDelay(VARIANT newVal)
```

1624
# ElecDistanceDelayUnit Property

**Description**: Sets and returns the units for specifying electrical delay in physical length (distance).

**VB Syntax**

```vbnet
meas.ElecDistanceDelayUnit = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>value</td>
<td>(enum naDistanceUnit)</td>
</tr>
</tbody>
</table>

Choose from:

- 0 – `naDistanceUnit_Meter`
- 1 – `naDistanceUnit_Feet`
- 2 – `naDistanceUnit_Inch`

**Return Type**: Enum

**Default**: 0 – `naDistanceUnit_Meter`

**Examples**

```vbnet
meas.ElecDistanceDelayUnit = naDistanceUnit_Meter  'Write
edelay = meas.ElecDistanceDelayUnit  'Read
```

**C++ Syntax**

```csharp
HRESULT get_ElecDistanceDelayUnit(tagNADistanceUnit *pVal)
HRESULT put_ElecDistanceDelayUnit(tagNADistanceUnit newVal)
```

**Interface**: IMeasurement11
# ElectricalDelay Property

**Description**
Sets the Electrical Delay for the active channel.

**VB Syntax**
`meas.ElectricalDelay = value`

**Variable**
*meas* - A Measurement (object)

*value* - (double) - Electrical Delay in seconds. Choose any number between -9.99 and 9.99

**Return Type**
Double

**Default**
0

**Examples**
```
meas.ElectricalDelay = 1e-3  'Write
edelay = meas.ElectricalDelay  'Read
```

**C++ Syntax**
```cpp
HRESULT get_ElectricalDelay(double *pVal)
HRESULT put_ElectricalDelay(double newVal)
```

**Interface**
IMeasurement
## Element Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a handle to the specified PathElement object. Each element object contains a unique set of values. The Value Property is used to set the value for each element.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command is used to set both RF and IF Path Configuration.</td>
</tr>
<tr>
<td></td>
<td>• See RF Path Configuration (elements, value)</td>
</tr>
<tr>
<td></td>
<td>• See IF Path Configuration (elements, value)</td>
</tr>
<tr>
<td></td>
<td>• See Spectrum Analyzer IF Gain path settings.</td>
</tr>
<tr>
<td>VB Syntax</td>
<td>Set elem = <em>pathConfig.Element (element)</em></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td></td>
<td><em>elem</em> (Object) IPathElement</td>
</tr>
<tr>
<td></td>
<td><em>pathConfig</em> A PathConfiguration (object)</td>
</tr>
<tr>
<td></td>
<td><em>element</em> (String) Configurable element. Use <em>pathConfig.Elements</em> to return a list of configurable elements or see a list of configurable elements for various VNA models.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Object</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>See examples:</td>
</tr>
<tr>
<td></td>
<td><em>IF Path Config</em></td>
</tr>
<tr>
<td></td>
<td><em>RF Path Config</em></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT Element( BSTR elemName, IPathElement** ppElement);</td>
</tr>
<tr>
<td>Interface</td>
<td>IPathConfiguration</td>
</tr>
</tbody>
</table>
## Elements Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns an array containing the names of configurable elements. See a list of configurable elements and settings for various VNA models.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>values = pathConfig.Elements</code></td>
</tr>
<tr>
<td>Variable</td>
<td><code>values</code> (Variant array) Variable to store the array of configurable elements.</td>
</tr>
<tr>
<td>pathConfig</td>
<td>A PathConfiguration (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Variant array</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>elems=pathconfig.Elements</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT Elements( VARIANT* pElements );</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IPathConfiguration</td>
</tr>
</tbody>
</table>
### Embed4PortA Property

**Description**
Returns the VNA port number associated with 'a' based on the device topology.

To see 'a' for all topologies, and to specify the port connections, use Embed4PortList Property

Specify topology using Embed4PortTopology Property

**VB Syntax**
```
value = fixture.Embed4PortA
```

**Variable**
- **(Type)** - Description
  - **value** (Short Integer) Variable to store the returned VNA port number.
  - **fixture** A Fixturing (object)

**Return Type**
Integer

**Default**
Not Applicable

**Examples**
```
value = fixture.Embed4PortA 'Read
```

**C++ Syntax**
```
HRESULT get_Embed4PortA(short *portA );
```

**Interface**
IFixturing2
### Embed4PortB Property

**Description**
Returns the VNA port number associated with 'b' based on the device topology.

To see 'b' for all topologies, and to specify the port connections, use `Embed4PortList Property`.

Specify topology using `Embed4PortTopology Property`.

**VB Syntax**
```vbnet
value = fixture.Embed4PortB
```

**Variable (Type) - Description**
- **Value** (Short Integer) Variable to store the returned VNA port number.
- **fixture** A Fixturing (object)

**Return Type**
Integer

**Default**
Not Applicable

**Examples**
```vbnet
value = fixture.Embed4PortB 'Read
```

**C++ Syntax**
```c++
HRESULT get_Embed4PortB(short *portB);
```

**Interface**
IFixturing2
## Embed4PortC Property

**Description** Returns the VNA port number associated with 'c' based on the device topology.

To see 'c' for all topologies, and to specify the port connections, use Embed4PortList Property.

Specify topology using Embed4PortTopology Property.

**VB Syntax**

```vbnet
value = fixture.Eembed4PortC
```

**Variable (Type) - Description**

- **value** (Short Integer) Variable to store the returned VNA port number.
- **fixture** A Fixturing (object)

**Return Type** Integer

**Default** Not Applicable

**Examples**

```vbnet
value = fixture.Embed4PortC 'Read
```

**C++ Syntax**

```cpp
HRESULT get_Eembed4PortC(short *portC);
```

**Interface** IFixturing2
### Embed4PortD Property

**Description**  
Returns the VNA port number associated with 'd' based on the device topology.

To see 'd' for all topologies, and to specify the port connections, use Embed4PortList Property.

Specify topology using Embed4PortTopology Property

**VB Syntax**

```vbnet
value = fixture.Embed4PortD
```

**Variable**  
(value) - Description

- value (Short Integer) Variable to store the returned VNA port number.

- fixture A Fixturing (object)

**Return Type**  
Integer

**Default**  
Not Applicable

**Examples**

```vbnet
value = fixture.Embed4PortD 'Read
```

**C++ Syntax**

```cpp
HRESULT get_Embed4PortD(short *portD );
```

**Interface**  
IFixturing2
### Embed4PortList Property

**Description**
Specifies the VNA port connections for ALL topologies. The port assignment is dependent on the DUT topology. All four port numbers are required. However, for:

- Topology A, only the first two arguments are valid,
- Topology B, only the first three arguments are valid,
- Topology C, ALL arguments are valid.

Specify topology using `Embed4PortTopology Property`.

Read the port assignments using the following commands. A, B, C, and D, refer to the port; NOT the topology.

- `Embed4PortA Property`
- `Embed4PortB Property`
- `Embed4PortC Property`
- `Embed4PortD Property`

<table>
<thead>
<tr>
<th>Topology A</th>
<th>Topology B</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Topology A Diagram" /></td>
<td><img src="image2.png" alt="Topology B Diagram" /></td>
</tr>
</tbody>
</table>
VB Syntax  

\[ \text{fixture.Embed4PortList}(p1, p2, p3) = p4 \]

Variable (Type) - Description

- **fixture** A Fixturing (object)
  - \( p1 \) VNA Port number assigned to \( a \) in above graphic.
  - \( p2 \) VNA Port number assigned to \( b \) in above graphic.
  - \( p3 \) VNA Port number assigned to \( c \) in above graphic.
  - \( p4 \) VNA Port number assigned to \( d \) in above graphic.

Default 1,2,3,4

Examples

Port 1, 2, 3, 4 configuration:
\[ \text{fixture.Embed4PortList}(1,2,3) = 4 \]  'Write

Port 4, 3, 2, 1 configuration:
\[ \text{fixture.Embed4PortList}(4,3,2) = 1 \]  'Write

Port 2, 4, 1, 3 configuration:
\[ \text{fixture.Embed4PortList}(2,4,1) = 3 \]  'Write

C++ Syntax

```cpp
HRESULT put_Embed4PortList(short pPortA, short pPortB, short pPortC, short pPortD);
```

Interface IFixturing2
**Embed4PortNetworkFilename Property**

**Description**
Specifies the 4-port touchstone file (*.s4p) in which the network to embed or de-embed resides. If the specified file does not exist, an error occurs when type command is sent.

Following this command, send Embed4PortNetworkMode Property.

**Note:** This command affects ALL measurements on the channel.

**VB Syntax**

```vbnet
fixture.Embed4PortNetworkFilename(netNum) = value
```

**Variable** *(Type)* - Description

- **fixture** A Fixturing (object)
- **netNum** (Integer) Network position. Choose from 1 or 2. See Embed4PortTopology Property
- **value** (String) Full path, file name, and extension (.s4P) of the circuit.

Files are typically stored in "D:\".

**Return Type**
String

**Default**
Not Applicable

**Examples**

```vbnet
fixture.Embed4PortNetworkFilename(2) = "D:\myFile.s4p" 'Write

value = fixture.Embed4PortNetworkFilename(1) 'Read
```

**C++ Syntax**

```c
HRESULT get_Embed4PortNetworkFilename( short networkNum, BSTR *filename);

HRESULT put_Embed4PortNetworkFilename( short networkNum, BSTR filename);
```

**Interface**
IFixturing2
### Embed4PortNetworkMode Property

**Description**
Specify the type of processing to take place on the specified 4-port network. First specify the network filename with `FSim.Embed4PortNetworkFilename` Property.

**VB Syntax**

```
fixture.Embed4PortNetworkMode(netNum) = value
```

**Variable**

- **fixture**  
  A Fixturing (object)

- **netNum** (Integer) Network position. Choose from 1 or 2. See `Embed4PortTopology` Property

- **value** (Enum as `NA4PortEmbedNetworkMode`) Processing mode. Choose from:
  - 0 or `naNO_NETWORK` - The same as disabling.
  - 1 or `naEMBED_NETWORK` - Add Network circuit.
  - 2 or `naDEEMBED_NETWORK` - Remove Network circuit

**Return Type**

Enum

**Default**

`naNO_NETWORK`

**Examples**

```vbnet
fixture.Embed4PortNetworkMode(1) = naNO_NETWORK 'Write
tvalue = fixture.Embed4PortNetworkMode(2) 'Read
```

**C++ Syntax**

```
HRESULT get_Embed4PortNetworkMode( short networkNum, 
tagNA4PortEmbedNetworkMode *eVal );
HRESULT put_Embed4PortNetworkMode( short networkNum, 
tagNA4PortEmbedNetworkMode eVal );
```

**Interface**

IFixturing2
### Embed4PortState Property

**Description**
Turns ON or OFF 4-port Network embedding for all ports on the channel.

**VB Syntax**
```vbnet
fixture.Embed4PortState = value
```

**Variable**
- **(fixture)**: A Fixturing (object)
- **(value)**: (Boolean)

- **False**: Turns Embedding OFF
- **True**: Turns Embedding ON

**Return Type**
Boolean

**Default**
False (OFF)

**Examples**
```vbnet
fixture.Embed4PortState = False 'Write
value = fixture.Embed4PortState 'Read
```

**C++ Syntax**
```cpp
HRESULT get_Emb4PortState(VARIANT_BOOL *pVal)
HRESULT put_Emb4PortState(VARIANT_BOOL newVal)
```

**Interface**
IFixturing2
# Embed4PortTopology Property

## Description
Specifies the VNA / DUT topology. Learn more about these and other VNA/DUT configurations.

## VB Syntax
```
fixture.Embed4PortTopology = value
```

## Variable
**Type** - Description
- `fixture` A Fixturing (object)
- `value` (Enum as NA4PortEmbedTopology) VNA / DUT topology. Choose from:
  - 0 or `naTOPOLOGY_A` - 2 VNA/DUT Ports
  - 1 or `naTOPOLOGY_B` - 3 VNA/DUT Ports
  - 2 or `naTOPOLOGY_C` - 4 VNA/DUT Ports
  - 3 or `naTOPOLOGY_D` - >4 VNA/DUT Ports

## Return Type
Enum

## Default
`naTOPOLOGY_A` (2 VNA/DUT Ports)

## Examples
```
fixture.Embed4PortTopology = naTOPOLOGY_A 'Write
value = fixture.Embed4PortTopology 'Read
```

## C++ Syntax
```
HRESULT get_Embed4PortTopology( tagNA4PortEmbedTopology *eVal );
HRESULT put_Embed4PortTopology( tagNA4PortEmbedTopology eVal );
```

## Interface
IFixturing2
### Enable Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns ON / OFF the trigger output.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>auxTrig.Enable = state</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>auxTrig</code></td>
<td>An AuxTrigger (object)</td>
</tr>
<tr>
<td><code>state</code></td>
<td>(boolean)</td>
</tr>
</tbody>
</table>

- **True** - Trigger Output ON
- **False** - Trigger Output OFF

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Boolean</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
</tbody>
</table>

**Examples**

```
auxTrig.Enable = True 'Write
value = auxTrig.Enable 'Read
```

**C++ Syntax**

```c++
HRESULT get_Enable(VARIANT_BOOL * enable);
HRESULT put_Enable(VARIANT_BOOL enable);
```

**Interface**

IAuxTrigger
# EnableADCDither Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the ADC dither state.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>( sa.\text{EnableADCDither} = \text{value} )</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td></td>
<td>( sa ) A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td></td>
<td>( \text{value} ) (Boolean) Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - ADC dithering is disabled.</td>
</tr>
<tr>
<td></td>
<td>1 - ADC dithering is enabled.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
<tr>
<td>Examples</td>
<td>See an example program.</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT put_EnableADCDither(VARIANT_BOOL bEnable);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_EnableADCDither(VARIANT_BOOL* bEnable);</td>
</tr>
<tr>
<td>Interface</td>
<td>ISpectrumAnalyzer</td>
</tr>
</tbody>
</table>
EnableADCFilterAuto Property

Description
Set and read how the ADC filter is set. When set to 1, ADC filter is set based on the
ADC sampling frequency.

VB Syntax

\[ sa.\text{EnableADCFilterAuto} = \text{value} \]

Variable (Type) - Description
\( sa \) A SpectrumAnalyzer (object)
\( value \) (Boolean) Choose from:

0 - OFF - ADC filter is set manually using ADCFilter.

1 - ON - Automatically select ADC filter based on the ADC sampling frequency.

Learn about these settings.

Return Type
Boolean

Default 1

Examples

\[
\begin{align*}
\text{sa.EnableADCFilterAuto} = 0 & \quad \text{'Write} \\
\text{value} = \text{sa.EnableADCFilterAuto} & \quad \text{'Read}
\end{align*}
\]

See an example program.

C++ Syntax

\[
\begin{align*}
\text{HRESULT put_EnableADCFilterAuto(VARIANT_BOOL bEnable);} \\
\text{HRESULT get_EnableADCFilterAuto(VARIANT_BOOL* bEnable);} \\
\text{ISpectrumAnalyzer}
\end{align*}
\]
### EnableADCSampleRateAuto Property

**Description**  
Set and read how the ADC sample rate is set. When set to 1, ADC sample rate is set based on the ADC filter setting.

**VB Syntax**  
`sa.EnableADCSampleRateAuto = value`

**Variable**  
*sa* A `SpectrumAnalyzer (object)`

*value* (Boolean) Choose from:

- **0 - OFF** - ADC sampling rate is set manually using `ADCSampleRate`.
- **1 - ON** - Automatically select ADC sample rate based on the ADC filter setting.

**Learn about these settings.**

**Return Type**  
Boolean

**Default**  
1

**Examples**  
```vbnet
sa.EnableADCSampleRateAuto = 1 'Write
value = sa.EnableADCSampleRateAuto 'Read
```

**See an example program.**

**C++ Syntax**  
```cpp
HRESULT put_EnableADCSampleRateAuto(VARIANT_BOOL bEnable);
HRESULT get_EnableADCSampleRateAuto(VARIANT_BOOL* bEnable);
```

**Interface**  
`ISpectrumAnalyzer`
EnableAllOutput Property

Description
Sets and returns the ON / OFF state of all DC sources for the channel.

VB Syntax
```
dc.EnableAllOutput = state
```

Variable (Type) - Description
- `dc`: An DCStimulus (object)
- `state`: (boolean)
  - True - All DC sources ON
  - False - All DC sources OFF

Return Type
Boolean

Default
False

Examples
```
dc.EnableAllOutput = True 'Write
value = dc.EnableAllOutput 'Read
```

C++ Syntax
```
HRESULT get_EnableAllOutput(VARIANT_BOOL * pValue);
HRESULT put_EnableAllOutput(VARIANT_BOOL newValue);
```

Interface
IDCStimulus
Enabled Property

Description
Enables and disables (ON/OFF) the port mapping and control line output of the specified test set.

If the specified test set is not connected or not ON, then setting Enabled = True will report an error. All other properties can be set when the test set is not connected.

When this command is set to ON or OFF, then the display of the test set status bar (ShowProperties Property) is also set to ON or OFF.

VB Syntax
`tset.Enabled = value`

Variable (Type) - Description

`tset`  A TestsetControl object

OR

An E5091Testset object

`value`  (Boolean)

`True`  Enables test set control.

`False`  Disables test set control.

Return Type
Boolean

Default
False

Examples
See E5091A Example Program
See External Testset Program

C++ Syntax
HRESULT get_Enabled(VARIANT_BOOL *state);
HRESULT put_Enabled(VARIANT_BOOL state);

Interface
ITestsetControl
IE5091Testsets
EnableDetectorBypass Property

Description  Set and read the ON/OFF state of the detector bypass setting.

VB Syntax  

\[ sa.\text{EnableDetectorBypass} = value \]

Variable  

(sa) A SpectrumAnalyzer (object)

(value) (Boolean) Choose from:

0 - OFF - Detector bypass OFF.

1 - ON - Detector bypass ON.

Learn about these settings.

Return Type  Boolean

Default  0

Examples  

\[ sa.\text{EnableDetectorBypass} = \text{ON} \quad \text{'Write} \]
\[ \text{value} = sa.\text{EnableDetectorBypass} \quad \text{'Read} \]

See an example program.

C++ Syntax  

HRESULT put_EnableDetectorBypass(VARIANT_BOOL bEnable);

HRESULT get_EnableDetectorBypass(VARIANT_BOOL* bEnable);

Interface  ISpectrumAnalyzer
### EnableDitherFFTGridOrigin Property

**Description**
Set and read the FFT grid dither state.

**VB Syntax**

```vbnet
sa.EnableDitherFFTGridOrigin = value
```

**Variable** *(Type) - Description*

- **sa**
  A `SpectrumAnalyzer` (object)
- **value** *(Boolean) Choose from:*
  - **0 - OFF** - FFT grid dithering is disabled.
  - **1 - ON** - FFT grid dithering is enabled.

**Return Type**
Boolean

**Default**
0

**Examples**
See an example program.

**C++ Syntax**

```cpp
HRESULT put_EnableDitherFFTGridOrigin(VARIANT_BOOL bEnable);
HRESULT get_EnableDitherFFTGridOrigin(VARIANT_BOOL* bEnable);
```

**Interface**
ISpectrumAnalyzer
### EnableForceADCRecordSize Property

**Description**  
Set and read enable force ADC record size mode.

**VB Syntax**  
`sa.EnableForceADCRecordSize = value`

**Variable**  

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sa</code></td>
<td>A <code>SpectrumAnalyzer</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean) Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - OFF - ADC record size set automatically.</td>
</tr>
<tr>
<td></td>
<td>1 - ON - Manually set ADC record to specified size using <code>ForceADCRecordSize</code>.</td>
</tr>
</tbody>
</table>

Learn about these settings.

**Return Type**  
Boolean

**Default**  
0

**Examples**  
See an example program.

**C++ Syntax**  
`HRESULT put_EnableForceADCRecordSize(VARIANT_BOOL bEnable);`

`HRESULT get_EnableForceADCRecordSize(VARIANT_BOOL* bEnable);`

**Interface**  
`ISpectrumAnalyzer`
EnableForceLOToFrequency Property

Description  Set and read enable force LO to frequency mode.

VB Syntax  

```
sa.EnableForceLOToFrequency = value
```

Variable  

**(Type)** - Description

- `sa` A `SpectrumAnalyzer (object)`
- `value` (Boolean) Choose from:

  0 - **OFF** - Force LO to frequency is disabled.

  1 - **ON** - Force LO to frequency is set manually using `ForceLOToFrequency`.

Learn about these settings.

Return Type  Boolean

Default  0

```
sa.EnableForceLOToFrequency = AUTO  'Write
value = sa.EnableForceLOToFrequency  'Read
```

Examples  See an example program.

C++ Syntax  

```
HRESULT put_EnableForceLOToFrequency(VARIANT_BOOL bEnable);
HRESULT get_EnableForceLOToFrequency(VARIANT_BOOL* bEnable);
```

Interface  ISpectrumAnalyzer
## EnableImageRejectTraces Property

**Description**  
Set and read the show / hide state of the image reject traces in the measurement parameters dialog.

**VB Syntax**  
```vbnet
sa.EnableImageRejectTraces = value
```

**Variable**  
- **(Type)** - Description
  - `sa`: A `SpectrumAnalyzer` (object)
  - `value`: (Boolean) Choose from:
    - **0** - OFF - Disable image reject traces.
    - **1** - ON - Enable image reject traces and set mode using `ImageReject`.

Learn about these settings.

**Return Type**  
Boolean

**Default**  
0

**Examples**  
```vbnet
sa.EnableImageRejectTraces = 0  'Write
value = sa.EnableImageRejectTraces 'Read
```

See an example program.

**C++ Syntax**  
```cpp
HRESULT put_EnableImageRejectTraces(VARIANT_BOOL bEnable);
HRESULT get_EnableImageRejectTraces(VARIANT_BOOL* bEnable);
```

**Interface**  
`ISpectrumAnalyzer`
## EnableLOPowerCal Property

**Description**
Sets and returns whether or not the LO power cal step is included in the cal steps when a Cal is performed.

Set LO Power Level using

**VB Syntax**

```vb
obj.EnableLOPowerCal(n) = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>A SMCType (object)</td>
</tr>
<tr>
<td></td>
<td>A VMCType (object)</td>
</tr>
<tr>
<td></td>
<td>A SweptIMDCal (object)</td>
</tr>
<tr>
<td></td>
<td>A NoiseCal (object)</td>
</tr>
</tbody>
</table>

**n**
LO Stage. Choose 1. (Only single LO allowed)

**value**
(Boolean) Choose from:

**False** - Skips over the LO Power Cal when calibrating.

**True** - Includes a step for LO Power Cal when calibrating.

**Return Type**
Boolean

**Default**
**False**

**Examples**

```vb
imd.EnableLOPowerCal(1)= true 'Write
loPwrCal = imd.EnableLOPowerCal(1) 'Read
```

**C++ Syntax**

```c++
HRESULT get_EnableLOPowerCal (long stage, BOOL *enable)

HRESULT put_EnableLOPowerCal (long stage, BOOL enable)
```

**Interface**
ISMCType
IVMCType
ISweptIMD2
INoiseCal2
## EnableModulationControl Property

**Description**
Sets and reads the state of the modulation control. Modulation control must be ON to control the modulation of an external source.

**VB Syntax**
```vbnet
extSource.EnableModulationControl = value
```

**Variable**
- **Type**: An `ExternalSource Object` (object)
- **Description**: `value` (Boolean) Choose from:
  - 1 - Enable control of external modulation.
  - 0 - Disable control of external modulation.

**Return Type**
Boolean

**Default**
0

**Examples**
```vbnet
extSource.EnableModulationControl = 1 'Write
value = extSource.EnableModulationControl 'Read
```

**C++ Syntax**
```cpp
HRESULT put_EnableModulationControl(VARIANT_BOOL bEnable);
HRESULT get_EnableModulationControl(VARIANT_BOOL* bEnable);
```

**Interface**
IExternalSource2
# EnableOffsetDelays Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables/disables modulator and ADC delays for pulse measurements.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pulse.EnableOffsetDelays = value</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>pulse</code></td>
<td>A PulseGenerator (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean) Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>ON</strong> (or 1) - Enable delays.</td>
</tr>
<tr>
<td></td>
<td><strong>OFF</strong> (or 0) - Disable delays.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pulse.EnableOffsetDelays = True</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>bool = pulse.EnableOffsetDelays</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_EnableOffsetDelays(VARIANT_BOOL* enable);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_EnableOffsetDelays(VARIANT_BOOL enable);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPulseGenerator5</td>
</tr>
</tbody>
</table>
### EnablePhase Property

**Description**
Sets and returns the state of SMC Phase measurements.

In the User Interface, there are two "enable phase" checkboxes: in the Phase Settings dialog and in the Calibration Wizard. Checking one enables both. This single command also enables both.

**VB Syntax**

```vbnet
obj.EnablePhase = bool
```

**Variable (Type) - Description**

- **obj**: A Mixer Interface pointer to the Measurement (object)
  - A Converter (Object)
- **bool**: (Boolean) -
  - **True**: Include phase in SMC measurements
  - **False**: Do NOT include phase in SMC measurements

**Return Type**: Boolean

**Default**: False

**Examples**

```vbnet
mixer.EnablePhase = True
```

**C++ Syntax**

```cpp
HRESULT get_EnablePhase(VARIANT_BOOL * val);
HRESULT put_EnablePhase(VARIANT_BOOL val);
```

**Interface**: IMixer13
**EnablePowerCompensation Property**

**Description**
Adjusts the source power at the specified port by the combined amount of loss through ALL enabled fixturing operations. Use this function to set the power level at the DUT input. Learn more.

*Note:* This command affects ALL measurements on the specified channel.

**VB Syntax**
```
fixture.EnablePowerCompensation(port) = bool
```

**Variable**
- **(Type)** - Description
  - fixture
    - A Fixturing (object)
  - port
    - (Integer) Port number to receive power compensation.
  - bool
    - (Boolean)

  **True** - Compensate source power
  **False** - Do NOT compensate source power

**Return Type**
Boolean

**Default**
False

**Examples**
```
fixture.EnablePowerCompensation(1) = True
value = fixture.EnablePowerCompensation(2) 'Read
```

**C++ Syntax**
```
HRESULT get_EnablePowerCompensation(short port, VARIANT_BOOL *pState );
HRESULT put_EnablePowerCompensation(short port, VARIANT_BOOL bVal);
```

**Interface**
IFixturing5
**EnableSnPDataExtrapolation**

**Description**
Turns ON and OFF SNP file extrapolation for both 2-port and 4-port embedding/de-embedding.

**VB Syntax**

```vbnet
fixture.EnableSnPDataExtrapolation = bool
```

**Variable (Type) - Description**

- `fixture` A **Fixturing (object)**
- `bool` **True** - Turns Extrapolation ON
  **False** - Turns Extrapolation OFF

**Return Type**
Boolean

**Default**
False

**Examples**

```vbnet
fixture.EnableSnPDataExtrapolation = True
value = fixture.EnableSnPDataExtrapolation
```

**C++ Syntax**

```cpp
HRESULT get_EnableSnPDataExtrapolation(VARIANT_BOOL *pExtrap);
HRESULT put_EnableSnPDataExtrapolation(VARIANT_BOOL bExtrap);
```

**Interface**
IFixturing6
## EnableRandomizedLO Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the LO randomize state.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>sa.EnableRandomizedLO = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td></td>
<td>(Object) <code>sa</code> A SpectrumAnalyzer</td>
</tr>
<tr>
<td></td>
<td>(Boolean) <code>value</code> Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - OFF - LO Randomize is set to OFF.</td>
</tr>
<tr>
<td></td>
<td>1 - ON - LO Randomize is set to ON.</td>
</tr>
<tr>
<td>Learn about these settings.</td>
<td></td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>1</td>
</tr>
<tr>
<td>Examples</td>
<td><code>sa.EnableRandomizedLO = OFF</code> Write</td>
</tr>
<tr>
<td></td>
<td><code>value = sa.EnableRandomizedLO</code> Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT put_EnableRandomizedLO(VARIANT_BOOL bEnable);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_EnableRandomizedLO(VARIANT_BOOL* bEnable);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ISpectrumAnalyzer</td>
</tr>
</tbody>
</table>
EnableSourceUnleveledEvents Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies whether or not to report Source Unleveled errors as system events. These events can trigger an OnSystemEvent call. This setting will revert to the default (False) setting on Instrument Preset.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>pref.EnableSourceUnleveledEvents = bool</code></td>
</tr>
</tbody>
</table>
| Variable    | `pref` - A Preferences (object)  
`bool` - Choose from:  
**False** - Do NOT report Source Unleveled Errors.  
**True** - Report Source Unleveled Errors. |
| Return Type | Boolean                                                                                                                                                                                      |
| Default     | False                                                                                                                                                                                        |
| Examples    | **Write**  
`pref.EnableSourceUnleveledEvents = False`  
**Read**  
`prefer = pref.EnableSourceUnleveledEvents` |
| C++ Syntax  | `HRESULT put_EnableSourceUnleveledEvents( VARIANT_BOOL bsourcUnlEnable)`  
`HRESULT get_EnableSourceUnleveledEvents( VARIANT_BOOL *bsourcUnlEnable)` |
| Interface   | IPreferences3                                                                                                                                                                               |
EndOfSweepOperation Property

Description
Set and read the action which should be taken at the end of the last frequency or power sweep in the measurement. This setting is used to protect a sensitive device from too much power during the sweep retrace.

VB Syntax
```vbnet
gca.EndOfSweepOperation = value
```

Variable
- **gca** - A GainCompression (object)
- **value** - (NAGCAEndOfSweepOperation)

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>naStandard (0)</td>
<td>Use the default VNA method. <a href="#">Learn more</a></td>
</tr>
<tr>
<td>naSetToStartPower (1)</td>
<td>Sweep Start power</td>
</tr>
<tr>
<td>naSetToStopPower (2)</td>
<td>Sweep Stop power.</td>
</tr>
<tr>
<td>naSetRFOff (3)</td>
<td>Always turn power OFF while waiting.</td>
</tr>
</tbody>
</table>

Return Type
Enum

Default
naStandard

Examples
```vbnet
gca.EndOfSweepOperation = naSetToStartPower 'Write
```
```vbnet
eos = gca.EndOfSweepOperation 'Read
```

C++ Syntax
```c++
HRESULT get_EndOfSweepOperation(tagNAGCAEndOfSweepOperation* pVal)

HRESULT put_EndOfSweepOperation(tagNAGCAEndOfSweepOperation newVal)
```

Interface
IGainCompression
# ENRFile Property

Sets and returns the name of the ENR file associated with the noise source.

## VB Syntax

```vbnet
noise.ENRFile = value
```

## Variable (Type) - Description

- **noise** (A NoiseCal object)
- **value** (string) Full path and ENR filename.

## Return Type

String

## Default

Not Applicable

## Examples

```vbnet
noise.ENRFile = "c:/ProgramFiles/Keysight/Network Analyzer/Documents/ENR/346C.enr" 'Write

ENR = noise.ENRFile 'Read
```

## C++ Syntax

```cpp
HRESULT get_ENRFile(BSTR* pValue)

HRESULT put_ENRFile(BSTR pNewValue)
```

## Interface

INoiseCal
### ENRID Property

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and returns ID of ENR table.</td>
</tr>
</tbody>
</table>

**VB Syntax**

```
enr.ENRSN = ID
```

**Variable** *(Type) - Description*

<table>
<thead>
<tr>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>enr</code></td>
</tr>
<tr>
<td>An <code>ENRFile</code> <em>(object)</em></td>
</tr>
</tbody>
</table>

| `ID` |
| Identifier for the ENR table |

**Return Type**

String

**Default**

Not Applicable

**Examples**

*See example program*

**C++ Syntax**

```
HRESULT get_ENRID(BSTR *Val);
HRESULT put_ENRID(BSTR Val);
```

**Interface**

IENRFile
# ENRSN Property

**Description**  
Sets and returns the serial number of the noise source for which the ENR file applies.

**VB Syntax**  
```
enr.ENRSN = serialNumber
```

**Variable**  
- `enr` **(Type)**: ENRFile **(object)**
- `serialNumber` **(String)**: Serial number of the noise source.

**Return Type**  
String

**Default**  
Not Applicable

**Examples**  
See example program

**C++ Syntax**  
```
HRESULT get_ENRSN(BSTR *Val);
HRESULT put_ENRSN(BSTR Val);
```

**Interface**  
IENRFile
**EqualTonePower Property - Superseded**

**Description**

*Note: This command is replaced with LevelingMethod Property*

Sets and returns the ON | OFF state of Equal Tone Power for the Swept IMD or IMSpectrum measurement.

**VB Syntax**

```vbnet
object.EqualTonePower = value
```

**Variable (Type) - Description**

- **object** - A SweptIMD or IMSpectrum Object
- **value** - (Boolean) - Choose from:
  - **True** - Set equal tone power.
  - **False** - Do NOT set equal tone power.

**Return Type**

Boolean

**Default**

False

**Examples**

```vbnet
ims.EqualTonePower = true 'Write
value = imd.EqualTonePower 'Read
```

**C++ Syntax**

```cpp
HRESULT get_EqualTonePower(VARIANT_BOOL* val)
```

```cpp
HRESULT put_EqualTonePower(VARIANT_BOOL val)
```

**Interface**

ISweptIMD2

IIMSSpectrum2
FastProcessing Property

Description
Set and return equation editor trace update delay. This property delays updating the equation editor trace until all trace references have finished updating to ensure that all data is present.

**Note:** This property does not work in application channels. In addition, this property does not work with the standard channel when the channel is in HOLD and then SINGLE sweeps are sent.

VB Syntax
`
equation.FastProcessing = value
`

Variable (Type) - Description
- `equation` (Type) - Description
  - A MeasurementEquation (object)
- `value` (Boolean) - Choose from:
  - **True** - Delay equation editor trace update.
  - **False** - Do not delay equation editor trace update.

Return Type
Boolean

Default
**True**

Examples
`
equation.FastProcessing = 1 'Write
fast = equation.FastProcessing 'Read
`

C++ Syntax
```
HRESULT get_FastProcessing(VARIANT_BOOL * pref);
HRESULT put_FastProcessing(VARIANT_BOOL pref);
```

Interface
IMeasurementEquation3
## ErrorCorrection Property

**Description**
Sets (or returns) error correction ON or OFF for the measurement.

**VB Syntax**
```
meas.ErrorCorrection = value
```

**Variable**
(Type) - Description

| meas   | A Measurement (object) |
| value  | (boolean)              |

- **False** - Turns error correction OFF
- **True** - Turns error correction ON

**Return Type**
Boolean

**Default**
See Error Correction

**Examples**
```
meas.ErrorCorrection = True  'Write
errcorr = meas.ErrorCorrection  'Read
```

**C++ Syntax**
```
HRESULT put_ErrorCorrection (VARIANT_BOOL bState)
HRESULT get_ErrorCorrection (VARIANT_BOOL *bState)
```

**Interface**
IMeasurement
## ErrorCorrection (Channel) Property

**Description**
Attempts to set error correction ON or OFF for all of the measurements on the channel. This setting may not be successful for some measurements because the Cal Set currently in place may not contain the appropriate calibration data. To read the error correction state for a measurement, use Error Correction Property.

**VB Syntax**
```vbnet
chan.ErrorCorrection = value
```

**Variable**
- **Type** - Description
- **chan** - A Channel (object)
- **value** - (boolean)

- **False** - Turns error correction OFF
- **True** - Turns error correction ON

**Return Type**
Boolean

**Default**
About Error Correction

**Examples**
```vbnet
chan.ErrorCorrection = True
```

**C++ Syntax**
```cpp
HRESULT put_ErrorCorrection (VARIANT_BOOL bState)
```

**Interface**
IChannel7
**ErrorCorrectionIndicator Property**

*Description*
Returns the error correction state for the measurement.

*VB Syntax*
```vbnet
value = meas.ErrorCorrectionIndicator
```

*Variable (Type) - Description*

- `value` (Enum) Error correction state.
  
  - **0 - naErrorCorrectionIndicator_None** - No error correction
  
  - **1 - naErrorCorrectionIndicator_Master** - Original error correction terms applied.
  
  - **2 - naErrorCorrectionIndicator_Interpolated** - Error terms are interpolated.
    Learn more
  
  - **3 - naErrorCorrectionIndicator_Delta** - Delta Match calibration terms. Learn more
  
  - **4 - naErrorCorrectionIndicator_Invalid** - Error terms are not valid.

*meas* A Measurement *(object)*

*Return Type* Enum as NAErrorCorrectionIndicator

*Default* See Error Correction

*Examples*
```vbnet
errcorr = meas.ErrorCorrectionIndicator 'Read
```

*C++ Syntax*
```cpp
HRESULT get_ErrorCorrectionIndicator (enum NAErrorCorrectionIndicator *pIndicator);
```

*Interface* IMeasurement14
**ErrorQuery Property**

**Description**
Sets and returns the Error Query command for an external DC Source and an external DC Meter.

**VB Syntax**
```vbnet
extDC.ErrorQuery = value
```

**Variable** *(Type) - Description*
- `extDC` A `ExternalDCDevice` *(object)*
- `value` *(String)* The SCPI command for returning DC Source and DC Meter errors.

**Return Type**
String

**Default**
"SYST:ERR?"

**Examples**
```vbnet
extDC.ErrorQuery = "SYST:ERR?" 'Write
value = extDC.ErrorQuery 'Read
```

**C++ Syntax**
```cpp
HRESULT get_ErrorQuery( BSTR* cmd);
HRESULT put_ErrorQuery( BSTR cmd);
```

**Interface**
IExternalDCDevice3
### ErrorTermUncertainty Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns whether the uncertainties associated with the correction error terms are being included in the uncertainty values for the measurement trace.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code> uncert.ErrorTermUncertainty = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code> uncert</code></td>
<td>An <strong>Uncertainty (object)</strong></td>
</tr>
<tr>
<td><code> value</code></td>
<td>(Boolean) Choose from:</td>
</tr>
<tr>
<td>False</td>
<td>Error Terms are NOT included.</td>
</tr>
<tr>
<td>True</td>
<td>Error Terms are included.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>True</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code> uncert.ErrorTermUncertainty = True</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_ErrorTermUncertainty(VARIANT_BOOL *pState);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_ErrorTermUncertainty(VARIANT_BOOL state);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IUncertainty</td>
</tr>
</tbody>
</table>
### ExitCmd Property

**Description**
Sets and returns the **Disable I/O command** for an external DC Source and an external DC Meter.

**VB Syntax**
```vbnet
extDC.ExitCmd = value
```

**Variable**
- **(Type)** - Description
  - `extDC` A `ExternalDCDevice` *(object)*
  - `value` *(String)* The SCPI command used to disable the DC Source and DC Meter.

**Return Type**
String

**Default**
" " Empty String

**Examples**
```vbnet
extDC.ExitCmd = "OUTP OFF" 'Write
value = extDC.ExitCmd 'Read
```

**C++ Syntax**
```cpp
HRESULT get_ExitCmd( BSTR* cmd);

HRESULT put_ExitCmd( BSTR cmd);
```

**Interface**
`IExternalDCDevice3`
## ExportReceiverCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of currently exported receivers.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = sa.dft.ExportReceiverCount</code></td>
</tr>
<tr>
<td><strong>Variable</strong> (Type)</td>
<td>(Long) Variable to store the returned value.</td>
</tr>
<tr>
<td><code>sa.dft</code></td>
<td>A SpectrumAnalyzerDFT (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) Variable to store the returned value.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>value = sa.dft.ExportReceiverCount 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_ExportReceiverCount(long* val);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISpectrumAnalyzerDFT</td>
</tr>
</tbody>
</table>
# ExportReceiverList Property

**Description**
Returns the currently exported receiver list.

**Note:** The list is set with `ExportReceiverSetList` can contain more receivers, this query will only return the ones that are currently measured and that are in the receiver list.

**VB Syntax**
```vbnet
list = sa.dft.ExportReceiverList
```

**Variable**
- **(Type) - Description**
  - `sa.dft` A `SpectrumAnalyzerDFT (object)`
  - `list` (String) Variable to store the returned list.

**Return Type**
String

**Example**
```
list = sa.dft.ExportReceiverList 'Read
```

**C++ Syntax**
```cpp
HRESULT get_ExportReceiverList(BSTR* ReceiverList);
```

**Interface**
`ISpectrumAnalyzerDFT`
### ExportReceiverSetList Property

**Description**
Sets and returns the list of exported receivers.

**Note:** This list can contain receivers that are not currently measured in the channel. However, this is not an issue. To get the current list of receivers that export data, query `ExportReceiverList`.

#### VB Syntax

```vbnet
sa.dft.ExportReceiverSetList = list
```

#### Variable (Type) - Description

- **sa.dft** A `SpectrumAnalyzerDFT` (object)
- **list** (String) Variable to store the returned list.

#### Return Type

String

#### Example

```vbnet
sa.dft.ExportReceiverSetList = "B" 'Write
list = sa.dft.ExportReceiverSetList 'Read
```

#### C++ Syntax

```c++
HRESULT get_ExportReceiverSetList(BSTR* ReceiverList);
HRESULT put_ExportReceiverSetList(BSTR ReceiverList);
```

#### Interface

`ISpectrumAnalyzerDFT`
## ExtDCDeviceCorrectionScale Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the correction scaling for an external DC Source.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = extDC.ExtDCDeviceCorrectionScale</code></td>
</tr>
<tr>
<td>Variable</td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>extDC</code></td>
<td>An <code>ExternalDCDevice (object)</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Variable to store the returned correction scaling value.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
<tr>
<td>Example</td>
<td><code>value = extDC.ExtDCDeviceCorrectionScale 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_ExtDCDeviceCorrectionScale(double *value)</td>
</tr>
<tr>
<td>Interface</td>
<td>IExternalDCDevice3</td>
</tr>
</tbody>
</table>
## ExtendedProperties Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Provides access to the custom properties and methods of an external device.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>Set <code>PSG = ExtDev.ExtendedProperties</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>PSG</code></td>
<td><em>(Object) Variable to store the returned handle to an external device.</em></td>
</tr>
<tr>
<td><code>ExtDev</code></td>
<td>An <code>ExternalDevice (object)</code></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Object</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>See Example</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT getExtendedProperties(IDispatch** ppObject);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IExternalDevice</td>
</tr>
</tbody>
</table>
### ExternalALC Property

**Description**
Sets or returns the source of the analyzer leveling control.

**VB Syntax**
```vbnet
app.ExternalALC = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
</tbody>
</table>
| `value` | (boolean) - Choose from:  
| True   | - Leveling control supplied through the rear panel. |
| False  | - Leveling control supplied inside the analyzer |

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
app.ExternalALC = True 'Write
extALC = app.ExternalALC 'Read
```

**C++ Syntax**

```c++
HRESULT get_ExternalALC(VARIANT_BOOL *pVal)
HRESULT put_ExternalALC(VARIANT_BOOL newVal)
```

**Interface**
IApplication
### ExternalDeviceDeActivatePolicy Property

**Description**
Set and return whether External Devices remain activated or are de-activated when the
VNA is Preset or when a Instrument State is recalled.

This setting remains until changed again using this command, or until the hard drive is
changed or reformatted.

See the ExternalDevices collection.

**VB Syntax**

```vbnet
pref.ExternalDeviceDeActivatePolicy = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pref</td>
<td>A Preferences (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Boolean) - Choose from:</td>
</tr>
</tbody>
</table>

- **True** - External device are **de-activated** when the VNA is Preset or when a Instrument State is recalled.
- **False** - External devices **remain active** when the VNA is Preset or when a Instrument State is recalled.

**Return Type**
Boolean

**Default**
**True**

**Examples**

```vbnet
pref.ExternalDeviceDeActivatePolicy = 1 'Write
dDevPolicy = pref.ExternalDeviceDeActivatePolicy 'Read
```

**C++ Syntax**

```csharp
HRESULT get_ExternalDeviceDeActivatePolicy(VARIANT_BOOL * pref);
HRESULT put_ExternalDeviceDeActivatePolicy(VARIANT_BOOL pref);
```

**Interface**
IPreferences10
ExternalTriggerConnectionBehavior Property

Description
Configures the external triggering signal for the VNA and PNA-X.

- TriggerSource Property is automatically set to External when ExternalTriggerConnectionBehavior is sent.
- Edge triggering is only available on some VNA models.
- For more information, see External Triggering.

VB Syntax
trigsetup. ExternalTriggerConnectionBehavior (conn) = value

Variable
Type - Description
trigsetup A TriggerSetup (object)
conn (enum NATriggerConnection) Rear Panel connector to send or receive trigger signals. Choose from:

Only one of the input connectors is active at a time. When a command is sent to one, the VNA automatically makes the other INACTIVE.

0 - naTriggerConnectionAUXT Trigger IN from rear-panel AUX IO connector Pin 19 (No longer supported.

1 - naTriggerConnectionBNC1 Trigger IN

- MEAS TRIG IN on models

2 - naTriggerConnectionBNC2 Trigger OUT. Only useful in point sweep mode.

- AUX TRIG 1 OUT on models

3 - naTriggerConnectionMATH Trigger IN from rear-panel Material Handler connector Pin 18

4 - naTriggerConnectionBypassPulse3 - Internal routing of pulse 3 output to the MEAS TRIG IN on the rear panel.

Value (enum NAExternalTriggerBehavior) -

0 - naTriggerInactive - Disables the specified connector.
Choose from 1 through 4 when `<conn>` is set to `naTriggerConnectionBNC1`

1 - `naTriggerInEdgeNegative` - Triggers the VNA when receiving a negative going signal

2 - `naTriggerInEdgePositive` - Triggers the VNA when receiving a positive going signal

3 - `naTriggerInLevelLow` - Triggers the VNA when receiving a low level signal

4 - `naTriggerInLevelHigh` - Triggers the VNA when receiving a High-level signal

Choose from 5 through 8 when `<conn>` is set to `naTriggerConnectionBNC2`.

In addition to sending this command, you must also use `TriggerOutputEnabled Property` to enable the BNC2 output.

5 - `naTriggerOutPulsePositiveAfter` - Sends a POSITIVE going TTL pulse at the END of each point during the sweep.

6 - `naTriggerOutPulsePositiveBefore` - Sends a POSITIVE going TTL pulse at the START of each point during the sweep.

7 - `naTriggerOutPulseNegativeAfter` - Sends a NEGATIVE going TTL pulse at the END of each point during the sweep.

8 - `naTriggerOutPulseNegativeBefore` - Sends a NEGATIVE going TTL pulse at the START of each point during the sweep.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
</table>
| BNC1 = `naTriggerInactive` | BNC2 = `naTriggerInactive` 

When `Output is enabled`

BNC1 = `naTriggerInactive`

BNC2 = `naTriggerOutPulsePositiveAfter`
Examples

```c
trigsetup.ExternalTriggerConnectionBehavior (naTriggerConnectionBNC1) = naTriggerInLevelLow
```

```c
trigBehav = trigsetup.ExternalTriggerConnectionBehavior
               (naTriggerConnectionAUXT)
```

C++ Syntax

```c
HRESULT get_ExternalTriggerConnectionBehavior(tagNATriggerConnection connection,tagNAExternalTriggerBehavior *trigger);

HRESULT put_ExternalTriggerConnectionBehavior(tagNATriggerConnection connection,tagNAExternalTriggerBehavior trigger);
```

Interface

ITriggerSetup
### ExternalTriggerDelay Property

**Description**
Sets and reads the trigger delay for all measurements in the CHANNEL. This delay is only applied while in app.Source = naTriggerSourceExternal and trigsetup.Scope = naChannelTrigger. After an external trigger is applied, the start of the sweep is delayed for the specified delay value plus any inherent latency.

To apply a trigger delay for all channels (Global), use TriggerDelay Property.

**VB Syntax**

```vbnet
chan.ExternalTriggerDelay = value
```

**Variable**

- **chan** - A Channel (object)
- **value** - Double - Trigger delay value in seconds. Range is from 0 to 3 seconds.

**Return Type**

Double

**Default**

0

**Examples**

```vbnet
chan.ExternalTriggerDelay = .003 'Write
delay = chan.ExternalTriggerDelay 'Read
```

**C++ Syntax**

```c++
HRESULT get_ExternalTriggerDelay(double *delay);
HRESULT put_ExternalTriggerDelay(double delay)
```

**Interface**

IChannel6
**ExtractionToneMode Property**

**Description**
Set and read the tuning tone mode. The tuning tone is the source at the output port to extract the X parameters.

**VB Syntax**

```
HotS22.ExtractionToneMode = value
```

**Variable**

*HotS22*(ActiveParametersApp (object))

*value* (Enum as NAExtractionToneMode) - Choose from:

- **0 - naRelative** Tone power is a dBc power relative to the input power.
- **1 - naAbsolute** Tone power is an absolute power.

**Return Type**

Enum

**Default**
naAbsolute

**Examples**

```
HotS22.ExtractionToneMode = naAbsolute 'Write
value = HotS22.ExtractionToneMode 'Read
```

**C++ Syntax**

```
HRESULT get_ExtractionToneMode(tagNAExtractionToneMode* value)
HRESULT put_ExtractionToneMode(tagNAExtractionToneMode value)
```

**Interface**
IActiveChannelSettings
### F1Frequency Property

**Description**  
Sets and returns the frequency of the F1 tone. Use with IMD sweep types:

- `naIMDToneCWSweep`
- `naIMDTonePowerSweep`

**VB Syntax**  
`object.F1Frequency = value`

**Variable**  
- **object** A SweptIMD or IMSpectrum Object
- **value** (Double) F1 tone frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

**Return Type**  
Double

**Default**  
.9995 GHz

**Examples**  
```vbnet
imd.F1Frequency = 100e6 'Write
value = imd.F1Frequency 'Read
```

**C++ Syntax**  
```c++
HRESULT get_F1Frequency(double *pVal)
HRESULT put_F1Frequency(double newVal)
```

**Interface**  
ISweptIMD

IIMSpectrum
# F2Frequency Property

**Description**  
Sets and returns the frequency of the F2 tone. Use with IMD sweep types:

- `naIMDToneCWSweep`
- `naIMDTonePowerSweep`

**VB Syntax**  
`object.F2Frequency = value`

**Variable**  
*Type* - Description

- **object**: A `SweptIMD` or `IMSpectrum` Object
- **value**: (Double) F2 tone frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

**Return Type**  
Double

**Default**  
1.0005 GHz

**Examples**  
```
imd.F2Frequency = 200e9  'Write
value = imd.F2Frequency  'Read
```

**C++ Syntax**  
```cpp
HRESULT get_F2Frequency(double *pVal)
HRESULT put_F2Frequency(double newVal)
```

**Interface**  
ISweptIMD

IMSpectrum
FailedTraces Property

Description  Set and return the limit line color of failed traces or failure indicators (dots) and the word Fail.

VB Syntax  \textit{colors.FailedTraces} = \textit{value}

Variable  (\textbf{Type}) - Description
\begin{itemize}
  \item \textit{colors} \hspace{1cm} \textbf{A ComColors \textbf{(object)}}
  \item \textit{value} \hspace{1cm} \textbf{(Long Integer)} - RGB color of the FailedTraces pen.
\end{itemize}

Convert the three RGB colors to an integer as follows:

\begin{equation}
\text{RGB} = R + (G \times 2^8) + (B \times 2^{16})
\end{equation}

To find the three RGB values from the Display Colors dialog, click Change Color, then Define Custom Color.

Return Type  Long

Default  Display = 255,20,20

Print = 255,20,20

Examples  \begin{itemize}
  \item \textbf{R} = 10
  \item \textbf{G} = 10
  \item \textbf{B} = 10
  \item \textbf{RGB} = R + (G \times 2^8) + (B \times 2^{16})
  \item \textit{colors.FailedTraces} = \text{RGB} \quad \textit{\textbf{Write}}
  \item \textit{color} = \textit{colors.FailedTraces} \quad \textit{\textbf{Read}}
\end{itemize}

C++ Syntax  \begin{itemize}
  \item HRESULT \texttt{get\_FailedTraces(long* pVal)};
  \item HRESULT \texttt{put\_FailedTraces(long newVal)};
\end{itemize}

Interface  IColors
FastCWPointCount Property

**Description**

Enables Fast CW sweep and sets the number of data points for the channel. *Sweep Type* must already be set to CWTime and FIFO must already be enabled.

**See Also**

FIFO and other Antenna Features

FIFO Object

Example program

N5264B Measurement Receiver

**VB Syntax**

```vb
chan.FastCWPointCount = value
```

**Variable (Type) - Description**

- **chan**
  - A Channel Object

- **value**
  - (Long Integer) Number of data points to measure in Fast CW mode. This setting overwrites the standard number of points setting for the channel. This setting overwrites the standard number of points setting for the channel. The minimum value is 1. The maximum value is $2^{32} - 1 = 2,147,483,647$. The "-1" indicates infinite point count (i.e., go forever). Any other value will produce invalid results.

If the data acquisition rate exceeds 400,000 points per second, the upper limit on the number of points is 11e6. The following are conditions that can cause the higher data rate:

- IFBW's $\geq 1$ MHz and internally triggered.

- fastCW sweeps that are externally triggered at a rate faster than 400,000 points per second.

Set to 0 to disable Fast CW.

**Return Type**

Long Integer

**Default**

0

**Examples**

```vb
chan.FastCWPointCount = 1e3 'Write
value = chan.FastCWPointCount 'Read
```
<table>
<thead>
<tr>
<th><strong>C++ Syntax</strong></th>
<th>HRESULT get_FastCWPPointCount(long *value)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HRESULT put_FastCWPPointCount(long value)</td>
</tr>
</tbody>
</table>

**Interface**

ICloneStream16
## FastMode Property

**Description**  
Sets and returns the state of a separate IFBW setting for leveling sweeps. ON allows a higher (faster) IFBW than the measurement sweep. It also causes leveling sweeps to be noisier.

**VB Syntax**  
```vb
RxLevel.FastMode(srcPort) = value
```

**Variable**  
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RxLevel</td>
<td>A ReceiverLeveling Object</td>
</tr>
<tr>
<td>srcPort</td>
<td>(Long Integer) Source port for which to set Fast Mode for Receiver Leveling.</td>
</tr>
</tbody>
</table>

**Value**  
(Boolean) Separate IFBW setting state. Choose from:

- **True** - Use separate IFBW setting. Specify IFBW using `LevelingIFBW`.
- **False** - Use same IFBW as the measurement sweep. Specify IFBW using `IF Bandwidth`.

**Return Type**  
Variant Boolean

**Default**  
True

**Examples**  
```vbnet
rxLevel.FastMode (1) = True ' Write
value = rxLevel.FastMode 2' Read
```

**C++ Syntax**  
```cpp
HRESULT get_FastMode(long port, VARIANT_BOOL* pVal);
HRESULT put_FastMode(long port, VARIANT_BOOL newVal);
```

**Interface**  
IReceiverLevelingConfiguration
**FFTResolution Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the FFT resolution.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>sa.FFTResolution = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>sa</code></td>
<td>A SpectrumAnalyzer <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Double) Variable to store the returned FFT resolution.</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>N/A</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = sa.FFTResolution 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_FFTResolution(double* resolution);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISpectrumAnalyzer</td>
</tr>
</tbody>
</table>
**FIFOEnabled Property**

**Description**
Enables/disables exporting data to the FIFO (First-IN, First-OUT) data buffer. FIFO is a circular buffer that allows very fast Read-Write access.

*Note:* FIFO commands are under the FIFO Object, and a new set of commands has been added here for binary data.

**VB Syntax**
```
sa.dft.FIFOEnabled = value
```

**Variable (Type) - Description**
- `sa.dft` A `SpectrumAnalyzerDFT (object)`
- `value` (Boolean) Choose from:
  - 0 - OFF - Export data to FIFO disabled.
  - 1 - ON - Export data to FIFO enabled.

**Return Type**
Boolean

**Default**
OFF

**Examples**
```
sa.dft.FIFOEnabled = ON 'Write
value = sa.dft.FIFOEnabled 'Read
```

**C++ Syntax**
```
HRESULT get_FIFOEnabled(VARIANT_BOOL* enable)
HRESULT put_FIFOEnabled(VARIANT_BOOL enable)
```

**Interface**
ISpectrumAnalyzerDFT
FileEraseEachSweep Property

Description
Enables/disables erasing output data files after each sweep. When disabled, data is appended to the output file after each sweep which could lead to very large file sizes (and eventually fill the disk).

VB Syntax
\[ \text{sa.dft.FileEraseEachSweep} = \text{value} \]

Variable
\( \text{sa.dft} \) - A SpectrumAnalyzerDFT (object)

\( \text{value} \) - (Boolean) Choose from:

0 - OFF - Erase data files after each sweep disabled.

1 - ON - Erase data files after each sweep enabled.

Return Type
Boolean

Default
ON

Examples
\[ \text{sa.dft.FileEraseEachSweep} = \text{ON} \ 'Write \]
\[ \text{value} = \text{sa.dft.FileEraseEachSweep} \ 'Read \]

C++ Syntax
HRESULT get_FileEraseEachSweep(VARIANT_BOOL* enable)
HRESULT put_FileEraseEachSweep(VARIANT_BOOL enable)

Interface
ISpectrumAnalyzerDFT
## FilePrefix Property

**Description**
Sets and returns the file name prefix for the data file. The receivers selected in `ExportReceiverSetList` will be appended to the specified prefix name with either "_X.txt" if a text file is exported (`TextFileEnabled`) or "_X.bin" if a binary file is exported (`BinaryFileEnabled`). X is the receiver name.

**VB Syntax**

```vbnet
sa.dft.FilePrefix = prefix
```

**Variable (Type) - Description**

- `sa.dft` *A SpectrumAnalyzerDFT (object)*
- `prefix` *(String) Specified prefix.*

**Return Type**

String

**Example**

```vbnet
sa.dft.FilePrefix = "C:\TEMP\SA_DATA_OUT"  'Write
value = sa.dft.FilePrefix  'Read
```

**C++ Syntax**

```cpp
HRESULT get_FilePrefix(BSTR* prefix);

HRESULT put_FilePrefix(BSTR prefix);
```

**Interface**

`ISpectrumAnalyzerDFT`
**FileVerboseEnabled Property**

**Description**
Enables/disables exporting frequency and data for text files. Data is not exported until the next new sweep occurs.

**VB Syntax**
```
sa.dft.FileVerboseEnabled = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type) - Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sa.dft</code></td>
<td>A <a href="#">SpectrumAnalyzerDFT</a> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean) Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - OFF - Disable verbose mode.</td>
</tr>
<tr>
<td></td>
<td>1 - ON - Enable verbose mode.</td>
</tr>
</tbody>
</table>

**Return Type**
Boolean

**Default**
OFF

**Examples**
```
sa.dft.FileVerboseEnabled = ON 'Write
value = sa.dft.FileVerboseEnabled 'Read
```

**C++ Syntax**
```
HRESULT get_FileVerboseEnabled(VARIANT_BOOL* enable)
HRESULT put_FileVerboseEnabled(VARIANT_BOOL enable)
```

**Interface**
ISpectrumAnalyzerDFT
## FilterBW Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the results of the SearchBandwidth method.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>filtBW = meas.FilterBW</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>filtBW</code></td>
<td>(single) - Variable to store bandwidth data</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Single</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>filterBW = meas.FilterBW</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_FilterBW(float* bw)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
**FilterCF Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Center Frequency result of the SearchBandwidth method.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>filtCF = meas.FilterCF</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>filtCF</code> (double) - Variable to store bandwidth CF data</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>filtCF = meas.FilterCF</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_FilterCF(double* centerFrequency)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
## FilterLoss Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Loss value of the SearchBandwidth method.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>filtLoss = meas.FilterLoss</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>filtLoss</code></td>
<td>(single) - Variable to store bandwidth Loss data</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Single</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>filterLoss = meas.FilterLoss 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_FiltLoss(float* loss)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
### FilterQ Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Q (quality factor) result of the SearchBandwidth method. The Q factor is the ratio of Center Frequency to Bandwidth (Center Frequency / Bandwidth).</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>filtQ = meas.FilterQ</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>filtQ</code></td>
<td>(single) - Variable to store bandwidth Q data</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Single</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>filtQ = meas.FilterQ</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_FilterQ(float* quality)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
# FilterErrors Property

## Description

Returns the error string associated with the digital filters. The return string has three fields separated by commas: "stage1 status, stage2 status, stage3 status"

Each of these fields can contain one or more of the following error codes:

- **NO ERROR**
- **NUMBER-OF-COEFFICIENTS** - the number of coefficients is excessive for that filter-stage
- **COEFFICIENT VALUE** - one or more coefficients are out of range for that filter-stage
- **SUM-OF-COEFFICIENTS** - the sum of all coefficients is excessive for that filter-stage,
- **FREQUENCY** - the frequency for Stage 1 is out of range (only applies stage1 field),
- **PARAMETER** - one or more parameters are out of range (only applies to stage 3 field).

## VB Syntax

```vbnet
value = spm4.FilterErrors
```

## Variable (Type) - Description

- **value**
  - Variable to store the returned errors.
- **spm4**
  - A `SignalProcessingModuleFour` object

## Return Type

String

## Default

Not Applicable

## Examples

```vbnet
mode = spm4.FilterErrors 'Read
'example return strings"

NO ERROR, NO ERROR, NO ERROR
indicates no errors,

*SUM-OF-COEFFICIENTS, NO ERROR, NO ERROR
indicates that the sum of all filter coefficients exceed the maximum value for the Stage-1 filter,

*COEFFICIENT *SUM-OF-COEFFICIENTS, NO ERROR, *PARAMETER
indicates a problems with Stage 1 coefficients and a problem with one or more of the parameters associated with the Stage 3 filter.
```
<table>
<thead>
<tr>
<th>C++ Syntax</th>
<th>HRESULT get_FilterErrors(BSTR* dspErrors);</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>ISignalProcessingModuleFour</td>
</tr>
</tbody>
</table>
**FilterMode Property**

**Description**
Sets and returns whether the VNA configures the 3-stage digital filter settings or they will be configured manually. When making manual settings, also send ADCCaptureMode Property which routes the IF through the 3-stage filter.

**VB Syntax**
```
spm4.FilterMode = value
```

**Variable**
*spm4 A SignalProcessingModuleFour (object)*

*value (enum as NAModes) Filter mode. Choose from:*

- **naAUTO**  VNA controls digital filter settings.
- **naMANUAL**  You control digital filter settings using other SignalProcessingModuleFour commands.

**Return Type**
Enum

**Default**
naAUTO

**Examples**
```
spm4.FilterMode = naAUTO 'Write
mode = spm4.FilterMode 'Read
```

**C++ Syntax**

```
HRESULT get_FilterMode(tagNAModes* dspMode);
HRESULT put_FilterMode(tagNAModes dspMode);
```

**Interface**
ISignalProcessingModuleFour
FirmwareMajorRevision Property

Description
Returns the major firmware revision number as an integer. For example, given a firmware revision number A.03.30, this command returns 3.

VB Syntax
value = cap.FirmwareMajorRevision

Variable (Type) - Description
value (Long) - Variable to store the returned integer value of the firmware revision number.

cap A Capabilities (object)

Return Type
Long

Default
Not Applicable

Examples
value = cap.FirmwareMajorRevision 'Read

C++ Syntax
HRESULT get_FirmwareMajorRevision(long * majorRev );

Interface
ICapabilities
# FirmwareMinorRevision Property

**Description**  
Returns the minor firmware revision number as an integer. For example, given a firmware revision number A.03.30, this command returns 30.

**VB Syntax**  
`value = cap.FirmwareMinorRevision`

**Variable (Type) - Description**  
- `value` (Long) - Variable to store the returned decimal value of the firmware revision number.
- `cap` A Capabilities (object)

**Return Type**  
Long

**Default**  
Not Applicable

**Examples**  
`value = cap.FirmwareMinorRevision`  
`value = cap.FirmwareMinorRevision`  

**C++ Syntax**  
`HRESULT get_FirmwareMinorRevision(long * minorRev );`

**Interface**  
ICapabilities
# FirmwareSeries Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the alpha portion of the firmware revision number. For example, given a firmware revision number A.03.30, this command returns A.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = cap.FirmwareSeries</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(String) - Variable to store the returned alpha value of the firmware revision number.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = cap.FirmwareSeries</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_FirmwareSeries(BSTR * series);</td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities</td>
</tr>
</tbody>
</table>
### FixedDelay Property

**Description**  
Set and return the known delay through the calibration mixer.

**VB Syntax**  
`smc.FixedDelay = value`

**Variable**  
**Type** - Description  
- `smc` An SMCType (object)  
- `value` (Double) Known delay through the calibration mixer in seconds.

**Return Type**  
Double

**Default**  
0 seconds

**Example**  
```
SMC.FixedDelay = 12e-9  'Write

value = SMC.FixedDelay  'Read
```

**C++ Syntax**  
```
HRESULT put_FixedDelay(Double Value);

HRESULT get_FixedDelay(Double* Value);
```

**Interface**  
SMCType5
## FixedPhase Property

**Description**
Write and read the fixed phase value. Must not be logarithmic sweep type.

**VB Syntax**
`phase.FixedPhase(srcPort) = value`

**Variable**
- **(Type)** - Description
  - **phase** A PhaseControl Object
  - **srcPort** (Long Integer) Source port for which to make phase control settings.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port

**value** (Double) Phase value in degrees. Choose a value between -360 and 360.

**Return Type**
Double

**Default**
0 degrees

**Examples**
- `phase.FixedPhase 1 = 15` ' Write
- `value = phase.FixedPhase 2` ' Read

**C++ Syntax**
- `HRESULT get_FixedPhase(long port, double* pVal);`
- `HRESULT put_FixedPhase(long port, double newVal);`

**Interface**
IPhaseControl
# FixedRatioedPower Property

**Description**
Write and read the fixed power ratioed value. Must NOT be in power sweep to use this value during phase control.

**VB Syntax**
\[ phase.FixedRatioedPower(srcPort) = value \]

**Variable**
- **phase** (Type: A PhaseControl Object)
- **srcPort** (Long Integer) Source port for which to make phase control settings.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port

- **value** (Double) Fixed power ratio value in dBC within the allowable range of the VNA.

**Return Type**
Double

**Default**
0 dBC

**Examples**
- `phase.FixedRatioedPower 1 = -1` 'Write
- `value = phase.FixedRatioedPower 2` 'Read

**C++ Syntax**
- `HRESULT get_FixedRatioedPower(long port, double* pVal);`
- `HRESULT put_FixedRatioedPower(long port, double newVal);`

**Interface**
IPhaseControl
**FixturingState Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns all three fixturing functions (de-embedding, port matching, impedance conversion) ON or OFF for all ports on the specified channel. This does NOT affect port extensions.</th>
</tr>
</thead>
</table>

**VB Syntax**

```vbnet
fixture.FixturingState = value
```

**Variable**

- **fixture** *(Type)* - Description
  - A Fixturing *(object)*
- **value** *(boolean)*
  - **True** - Turns Fixturing ON
  - **False** - Turns Fixturing OFF

**Return Type**

Boolean

**Default**

False

**Examples**

```vbnet
fixture.FixturingState = True 'Write
value = fixture.FixturingState 'Read
```

**C++ Syntax**

```c
HRESULT get_FixturingState(VARIANT_BOOL *pVal)
HRESULT put_FixturingState(VARIANT_BOOL newVal)
```

**Interface**

IFixturing
## FootSwitch Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Reads the Footswitch Input (pin 20 of the AUX IO connector).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( value = AuxIO.Footswitch )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>( value )</td>
<td>(boolean) - Variable to store the returned value</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>foot switch is released</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>foot switch is depressed</td>
</tr>
<tr>
<td><strong>AuxIO</strong></td>
<td>(object) - A Hardware Aux I/O object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>True</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>( fs = aux.Footswitch )</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_FootSwitch ( VARIANT_BOOL* State );</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IHWAuxIO</td>
</tr>
</tbody>
</table>
FootswitchMode Property  **Obsolete**

<table>
<thead>
<tr>
<th>Description</th>
<th>Determines what occurs when the footswitch is pressed. For more information see the FootSwitch In pin description in the Auxiliary IO connector.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><strong>AuxIo.FootSwitchMode = value</strong></td>
</tr>
<tr>
<td><strong>Variable</strong> <em>(Type)</em> - Description</td>
<td><strong>value</strong> <em>(enum NAFootSwitchMode )</em></td>
</tr>
<tr>
<td>0 - naIgnoreFootswitch</td>
<td>Footswitch presses are ignored.</td>
</tr>
<tr>
<td>1 - naSweepTrigger</td>
<td>Footswitch presses trigger a sweep. The VNA must be in Manual Trigger Mode.</td>
</tr>
<tr>
<td>2 - naRecallNextState</td>
<td>Footswitch presses recall an instrument state. When more than one state is available, then each footswitch press recalls the next state, then starts over from the beginning. It is possible for a recalled state to override the current mode. If the recalled state is IGNore, then mode changes and additional footswitch presses are ignored.</td>
</tr>
<tr>
<td>3 - naRunMacro</td>
<td>Footswitch presses load and run a macro. When more than one macro is available, then each footswitch press loads and runs the next macro, then starts over from the beginning. It is possible for a Macro to override the current mode. If the macro contains a Preset, then the mode changes to the default setting IGNore and additional footswitch presses are ignored.</td>
</tr>
</tbody>
</table>

**AuxIO** *(object)* - A Hardware Aux I/O object

**Return Type**  NAFootSwitchMode

**Default**  0 - naIgnoreFootswitch

**Examples**  `auxIo.FootSwitchMode = naIgnoreFootSwitch`  'Write

**C++ Syntax**

```
HRESULT get_FootSwitchMode(NAFootSwitchMode *pFootSwitchMode )
HRESULT put_FootSwitchMode(NAFootSwitchMode newFootSwitchMode)
```

**Interface**  IHWAuxIO3
**ForceADCRecordSize Property**

**Description**
Set and read the ADC record size.

**VB Syntax**

```
sa.ForceADCRecordSize = value
```

**Variable (Type) - Description**

- `sa` A SpectrumAnalyzer (object)
- `value` (Long) Choose a value between 64 and 33554432

**Learn about these settings.**

**Return Type**
Long

**Default**
64

**Examples**

```
sa.ForceADCRecordSize = 256    'Write
value = sa.ForceADCRecordSize 'Read
```

See an example program.

**C++ Syntax**

```
HRESULT put_ForceADCRecordSize(long val);
HRESULT get_ForceADCRecordSize(long* val);
```

**Interface**
ISpectrumAnalyzer
**ForceDeEmbedENRAdapter Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the De-embedENRAdapter state. <a href="#">Learn more.</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>noiseCal.<strong>ForceDeEmbedENRAdapter</strong> = value</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>noiseCal</td>
<td>A <strong>NoiseCal</strong> (object)</td>
</tr>
<tr>
<td>value</td>
<td>(boolean) - ENR Adapter de-embed state.</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>Do not Force de-embedding.</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>Force de-embedding.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
<tr>
<td>Examples</td>
<td>noiseCal.<strong>ForceDeEmbedENRAdapter</strong> = False 'Write</td>
</tr>
<tr>
<td></td>
<td>AdapterDembed = noiseCal.<strong>ForceDeEmbedENRAdapter</strong> 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_ForceDeEmbedENRAdapter(VARIANT_BOOL* on);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_ForceDeEmbedENRAdapter(VARIANT_BOOL on);</td>
</tr>
<tr>
<td>Interface</td>
<td>INoiseCal2</td>
</tr>
</tbody>
</table>
### ForceDeEmbedSensorAdapter Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the state of power sensor adapter de-embedding. <a href="#">Learn more</a>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>noiseCal. ForceDeEmbedSensorAdapter = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td>noiseCal</td>
<td>A NoiseCal (<a href="#">object</a>)</td>
</tr>
<tr>
<td>value</td>
<td>(<a href="#">boolean</a>) - Power sensor adapter de-embed state.</td>
</tr>
<tr>
<td>False</td>
<td>Do not Force de-embedding.</td>
</tr>
<tr>
<td>True</td>
<td>Force de-embedding.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
<tr>
<td>Examples</td>
<td><code>noiseCal. ForceDeEmbedSensorAdapter = False</code> 'Write'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_ForceDeEmbedSensorAdapter(VARIANT_BOOL* on);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>INoiseCal2</td>
</tr>
<tr>
<td>AdapterDembed</td>
<td><code>noiseCal. ForceDeEmbedSensorAdapter</code> 'Read'</td>
</tr>
<tr>
<td>put_ForceDeEmbedSensorAdapter</td>
<td><code>HRESULT put_ForceDeEmbedSensorAdapter(VARIANT_BOOL on);</code></td>
</tr>
</tbody>
</table>
**ForceDeEmbedThruAdapter Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the state of Thru adapter de-embedding. <a href="#">Learn more</a>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>noiseCal.ForceDeEmbedThruAdapter = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>noiseCal</code></td>
<td>A NoiseCal (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(boolean) - Thru Adapter de-embed state.</td>
</tr>
<tr>
<td>False</td>
<td>Do not Force de-embedding.</td>
</tr>
<tr>
<td>True</td>
<td>Force de-embedding.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>noiseCal.ForceDeEmbedThruAdapter = False</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>AdapterDembed = noiseCal.ForceDeEmbedThruAdapter</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_ForceDeEmbedThruAdapter(VARIANT_BOOL* on);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_ForceDeEmbedThruAdapter(VARIANT_BOOL on);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>INoiseCal4</td>
</tr>
</tbody>
</table>
### ForceLOToFrequency Property

**Description**
Set and read the LO frequency.

**VB Syntax**
`sa.ForceLOToFrequency = value`

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sa</code></td>
<td>A <code>SpectrumAnalyzer</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Choose a value within the LO frequency range.</td>
</tr>
</tbody>
</table>

Learn about these settings.

**Return Type**
Double

**Default**
1 GHz

**Examples**

```vbnet
sa.ForceLOToFrequency = 1e9 'Write
value = sa.ForceLOToFrequency 'Read
```

See an example program.

**C++ Syntax**

```c++
HRESULT put_ForceLOToFrequency(double val);
HRESULT get_ForceLOToFrequency(double* val);
```

**Interface**
ISpectrumAnalyzer
### Format Property (marker)

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets (or returns) the format of the marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mark.Format = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><code>mark</code> A Marker (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum NAMarkerFormat) - Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naMarkerFormat_LinMag</td>
</tr>
<tr>
<td></td>
<td>1 - naMarkerFormat_LogMag</td>
</tr>
<tr>
<td></td>
<td>2 - naMarkerFormat_Phase</td>
</tr>
<tr>
<td></td>
<td>3 - naMarkerFormat_Delay</td>
</tr>
<tr>
<td></td>
<td>4 - naMarkerFormat_Real</td>
</tr>
<tr>
<td></td>
<td>5 - naMarkerFormat_Imaginary</td>
</tr>
<tr>
<td></td>
<td>6 - naMarkerFormat_SWR</td>
</tr>
<tr>
<td></td>
<td>7 - naMarkerFormat_LinMagPhase</td>
</tr>
<tr>
<td></td>
<td>8 - naMarkerFormat_LogMagPhase</td>
</tr>
<tr>
<td></td>
<td>9 - naMarkerFormat_RealImaginary</td>
</tr>
<tr>
<td></td>
<td>10 - naMarkerFormat_ComplexImpedance</td>
</tr>
<tr>
<td></td>
<td>11 - naMarkerFormat_ComplexAdmittance</td>
</tr>
<tr>
<td></td>
<td>12 - naMarkerFormat_Kelvin</td>
</tr>
<tr>
<td></td>
<td>13 - naMarkerFormat_Fahrenheit</td>
</tr>
<tr>
<td></td>
<td>14 - naMarkerFormat_Celsius</td>
</tr>
<tr>
<td></td>
<td>15 - naMarkerFormat_Default - the same format as the trace.</td>
</tr>
<tr>
<td></td>
<td>16 - naMarkerFormat_Noise - Available ONLY in IM Spectrum and SA measurement classes.</td>
</tr>
<tr>
<td>Return Type</td>
<td>NAMarkerFormat</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Default</td>
<td>15 -naMarkerFormat_Default</td>
</tr>
<tr>
<td>Examples</td>
<td>mark.Format = naMarkerFormat_SWR 'Write</td>
</tr>
<tr>
<td></td>
<td>fmt = mark.Format 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_Format(tagNAMarkerFormat *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Format(tagNAMarkerFormat newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker</td>
</tr>
</tbody>
</table>
**FormatUnit Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the units for the specified data format. Measurements with display formats other than those specified are not affected.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.FormatUnit (format) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><code>format</code></td>
<td>(enum NADataFormat) - Choose from:</td>
</tr>
<tr>
<td>0</td>
<td>naDataFormat_LinMag</td>
</tr>
<tr>
<td>1</td>
<td>naDataFormat_LogMag</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum naFormatUnit)</td>
</tr>
<tr>
<td>For LogMag, choose from:</td>
<td></td>
</tr>
<tr>
<td>0 - <strong>naFormatUnit_dBm</strong></td>
<td>Units are displayed in dBm. 0 dBm = 0.001 watt</td>
</tr>
</tbody>
</table>
| 1 - **naFormatUnit_dBmV** | Units are displayed in dBmV. 0 dBmV = 0.001 volt  
DBmV value depends on the reference impedance: dBmV = dBm + 30 + 10*log10(Z0) |
| 2 - **naFormatUnit_dBmA** | Units are displayed in dBmA. 0 dBmA = 0.001 Ampere |
| 6 - **naFormatUnit_dBuV** | Units are displayed in dBuV. 0 dBuV = 1 uV  
DBuV value depends on the reference impedance: dBuV = dBm + 90 + 10*log10(Z0) |
| For LinMag, choose from: |
| 3 - **naFormatUnit_W** | Watts |
| 4 - **naFormatUnit_V** | Volts |
| 5 - **naFormatUnit_A** | Amperes |
| **Return Type** | Enum |
| **Default** | `0 - naFormatUnit_dBm` |
### Examples

```csharp
meas.FormatUnit(1) = naFormatUnit_dBmV  // 'Write

units = meas.FormatUnit(1)           // 'Read
```

### C++ Syntax

```c++
HRESULT put_FormatUnit(tagDataFormat format, tagFormatUnit unit)

HRESULT get_FormatUnit(tagDataFormat format, tagFormatUnit* unit)
```

### Interface

IMeasurement9
## Frequency Property

**Description**
Sets group delay aperture using a fixed frequency range.

**VB Syntax**
`gdAperture.Frequency = value`

**Variable** *(Type)* - Description

- **gdAperture** A GroupDelayAperture *(object)*
- **value** *(Double)* Frequency range (in Hz) to use for the aperture setting.

**Return Type**
Double

**Default**
Frequency range that equates to 11 points.

This can be changed to two points with a preference setting.

**Examples**
```
gdAperture.Frequency = 1e6 'Write
aperture = gdAperture.Frequency 'Read
```

**C++ Syntax**
- `HRESULT get_Frequency(double Frequency *pVal)`
- `HRESULT put_Frequency(double Frequency newVal)`

**Interface**
IGroupDelayAperture
**Frequency Property**

| Description                                      | Sets or returns the frequency associated with a Power Sensor CalFactor Segment
|                                                 | or
|                                                 | Sets or returns the frequency associated with a Power Loss Segment.
| **VB Syntax**                                   | *object*.**Frequency** = *value*
| **Variable (Type) - Description**               | *object*
|                                                 | One of the following objects:
|                                                 | PowerSensorCalFactorSegment
|                                                 | PowerSensorCalFactorSegmentPMAR
|                                                 | PowerLossSegment
|                                                 | PowerLossSegmentPMAR
| **value** (double) – Frequency in units of Hz.  | This can be any non-negative value (limited by the maximum value of double).
| **Return Type**                                 | Double
| **Default**                                     | 0
| **Examples**                                    | `seg.Frequency = 6e9`  `Write`
|                                                 | `freq = seg.Frequency`  `Read`
| **C++ Syntax**                                  | HRESULT put_Frequency(double newVal);
|                                                 | HRESULT get_Frequency(double *pVal);
| **Interface**                                   | One of the above objects.
### FrequencyCenter Property

**Description**
Sets and returns the center frequency of the main tones. Use with IMD sweep types:

- `naIMDToneCWSweep`
- `naIMDTonePowerSweep`
- `naIMDDeltaFrequencySweep`

**VB Syntax**
```vbnet
object.FrequencyCenter = value
```

**Variable**
- **Object**
  - A SweptIMD or IMSpectrum Object
- **Value**
  - (Double) Tone center frequency in Hz. Both the F1 and F2 tones MUST be within the frequency range of the VNA.

**Return Type**
Double

**Default**
1.0 GHz

**Examples**
```vbnet
imd.FrequencyCenter = 2e9 'Write
value = imd.FrequencyCenter 'Read
```

**C++ Syntax**
```cpp
HRESULT get_FrequencyCenter(double *pVal)
HRESULT put_FrequencyCenter(double *pVal)
```

**Interface**
ISweptIMD

IIMSpectrum
## FrequencyCenterCenter Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the sweep center frequency when sweeping the main tones. Use with sweep type = naIMDToneCenterFreqSweep.</th>
</tr>
</thead>
</table>

### VB Syntax

```vbnet
imd.FrequencyCenterCenter = value
```

### Variable (Type) - Description

<table>
<thead>
<tr>
<th>imd</th>
<th>A SweptIMD Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(Double) Center frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.</td>
</tr>
</tbody>
</table>

### Return Type

Double

### Default

13.255 GHz

### Examples

```vbnet
imd.FrequencyCenterCenter = 10e9 'Write
value = imd.FrequencyCenterCenter 'Read
```

### C++ Syntax

```c++
HRESULT get_FrequencyCenterCenter(double *pVal)
HRESULT put_FrequencyCenterCenter(double newVal)
```

### Interface

ISweptIMD
# FrequencyCenterSpan Property

**Description**: Sets and returns the frequency span when sweeping the main tones. Use with sweep type `naIMDToneCenterFreqSweep`.

**VB Syntax**: `imd.FrequencyCenterSpan = value`

**Variable (Type) - Description**

- `imd` - A SweptIMD Object
- `value` - (Double) Frequency span in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

**Return Type**: Double

**Default**: 26.489 GHz

**Examples**:  
```
imd.FrequencyCenterSpan = 10e9 'Write
value = imd.FrequencyCenterSpan 'Read
```

**C++ Syntax**:  
```
HRESULT get_FrequencyCenterSpan(double *pVal)
HRESULT put_FrequencyCenterSpan(double newVal)
```

**Interface**: ISweptIMD
# FrequencyCenterStart Property

**Description**
Sets and returns the start frequency when sweeping the main tones. Use with sweep type = `naIMDToneCenterFreqSweep`.

**VB Syntax**
```vbnet
imd.FrequencyCenterStart = value
```

**Variable (Type) - Description**

- `imd` - A SweptIMD Object
- `value` - (Double) Start frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

**Return Type**
Double

**Default**
10.5 MHz

**Examples**
```vbnet
imd.FrequencyCenterStart = 20e6 'Write
value = imd.FrequencyCenterStart 'Read
```

**C++ Syntax**
```cpp
HRESULT get_FrequencyCenterStart(double *pVal)
HRESULT put_FrequencyCenterStart(double newVal)
```

**Interface**
ISweptIMD
## FrequencyCenterStop Property

**Description**
Sets and returns the stop frequency when sweeping the main tones. Use with sweep type = `naIMDToneCenterFreqSweep`.

**VB Syntax**
```
imd.FrequencyCenterStop = value
```

**Variable (Type) - Description**
- **imd**: A `SweptIMD` Object
- **value** (Double): Stop frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

**Return Type**
Double

**Default**
26.4995 MHz

**Examples**
```
imd.FrequencyCenterStop = 20e9  'Write
value = imd.FrequencyCenterStop  'Read
```

**C++ Syntax**
```
HRESULT get_FrequencyCenterStop(double *pVal)
HRESULT put_FrequencyCenterStop(double newVal)
```

**Interface**
ISweptIMD
# FrequencyOffsetDivisor Property **Superseded**

**Description**
This method is replaced by properties on the FOMRange Object. Specifies (along with FrequencyOffsetMultiplier) the value to multiply by the stimulus. See other Frequency Offset properties

**VB Syntax**
```
object.FrequencyOffsetDivisor = value
```

**Variable** *(Type)* - Description

- **object** Channel *(object)*
- or

- **CalSet (object)** - Read-only property
  
- **value** *(Double)* - Divisor value. Range is 1 to 1000

**Return Type** Double

**Default** 1

**Examples**
```
chan.FrequencyOffsetDivisor = 2 'Write
fOffsetDiv = chan.FrequencyOffsetDivisor 'Read
```

**C++ Syntax**
```
HRESULT get_FrequencyOffsetDivisor(double*pval)

HRESULT put_FrequencyOffsetDivisor(double newVal)
```

**Interface**
IChannel2

|CalSet3
**FrequencyOffsetFrequency Property Superseded**

**Description**
This method is replaced by properties on the FOMRange Object.

Specifies an absolute offset frequency in Hz. For mixer measurements, this would be the LO frequency. See other Frequency Offset properties.

**VB Syntax**

```vbnet
object.FrequencyOffsetFrequency = value
```

**Variable**

**(Type) - Description**

- `object` Channel *(object)*

  or

  CalSet *(object)* - Read-only property

**(value) - Description**

- *(Double)* - Offset value. Range is +/- 1000 GHz. (Offsets can be positive or negative.)

**Return Type**

- Double

**Default**

- 0 Hz

**Examples**

```vbnet
chan.FrequencyOffsetFrequency = 2 'Write
fOffsetFreq = chan.FrequencyOffsetFrequency 'Read
```

**C++ Syntax**

```c++
HRESULT get_FrequencyOffsetFrequency(double*pval)
HRESULT put_FrequencyOffsetFrequency(double newVal)
```

**Interface**

- IChannel2
- ICalSet3
**FrequencyOffsetMultiplier Property Superseded**

**Description**
This method is replaced by properties on the FOMRange Object.

Specifies (along with FrequencyOffsetDivisor) the value to multiply by the stimulus. See other Frequency Offset properties.

**VB Syntax**

```vbnet
object.FrequencyOffsetMultiplier = value
```

**Variable (Type) - Description**

- `object`: Channel `(object)`

  or

- `CalSet (object)`: Read-only property

  - `value` (Double) - Multiplier value. Range is 1 to 1000

**Return Type**
Double

**Default**
1

**Examples**

```vbnet
chan.FrequencyOffsetMultiplier = 2 'Write
fOffsetMult = chan.FrequencyOffsetMultiplier 'Read
```

**C++ Syntax**

```c++
HRESULT get_FrequencyOffsetMultiplier (double*pval);
HRESULT put_FrequencyOffsetMultiplier (double newVal);
```

**Interface**

`IChannel2`

`|CalSet3`
**Write/Read**

### FrequencyOffsetRangeForCalComputations Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the FOM frequency range to use when performing calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pref.FrequencyOffsetRangeForCalComputations = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>pref</code></td>
<td>A <code>Preferences</code> <code>object</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td><code>(Enum as NACalFOMRange)</code> - Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>0 - naCalFOMRangeAuto</strong> - All other calibration situations.</td>
</tr>
<tr>
<td></td>
<td><strong>1 - naCalFOMRangePrimary</strong> - Used for calibrating at the mmWave frequencies when NOT using a test set. <a href="#">Learn more.</a></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td><code>Enum</code></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td><code>naCalFOMRangeAuto</code></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pref.FrequencyOffsetRangeForCalComputations = naCalFOMRangePrimary</code></td>
</tr>
<tr>
<td></td>
<td><code>calPref = pref.FrequencyOffsetRangeForCalComputations</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_FrequencyOffsetRangeForCalComputations(tagNACalFOMRange * val);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_FrequencyOffsetRangeForCalComputations((tagNACalFOMRange val);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IPreferences10</code></td>
</tr>
</tbody>
</table>
### FrequencyOffsetCWOverride Property

**Superseded**

**Description**

This method is replaced by properties on the FOMRange Object.

Establishes a fixed (CW) stimulus frequency while measuring the Response over a swept frequency range. For example, a fixed-frequency VNA stimulus may be applied to the RF input of a mixer whose local oscillator (LO) is being swept. Because the IF output of the mixer will be swept, the VNA receivers must also be swept.

See other Frequency Offset properties.

**VB Syntax**

```vbnet
object.FrequencyOffsetCWOverride = value
```

**Variable (Type) - Description**

- `object` Channel (object)

  or

- CalSet (object) - Read-only property

  ```vbnet
  value (Enum as NaStates) - Choose from:
  naOFF (0) - Turns CW override OFF
  naON (1) - Turns CW override ON
  ```

**Return Type**

Enum

**Default**

0 Hz

**Examples**

```vbnet
chan.FrequencyOffsetCWOverride = 1 'Write
fOffsetOV = chan.FrequencyOffsetCWOverride 'Read
```

**C++ Syntax**

```c++
HRESULT get_FrequencyOffsetCWOverride (tagNAStates *pstate)
HRESULT put_FrequencyOffsetCWOverride (tagNAStates newState)
```

**Interface**

- IChannel2
- CalSet3

---

1730
**FrequencyOffsetState Property Superseded**

**Description**
This method is replaced by properties on the FOMRange Object.

Enables Frequency Offset on ALL measurements that are present on the active channel. This immediately causes the source and receiver to tune to separate frequencies. The receiver frequencies are specified with other channel and offset settings. To make the stimulus settings, use Channel Start, Stop Frequency properties. See other Frequency Offset properties.

Tip: To avoid unnecessary errors, first make other frequency offset settings. Then turn Frequency Offset ON.

**VB Syntax**

```vbnet
object.FrequencyOffsetState = value
```

**Variable**

**object** - Channel (**object**)

or

**CalSet (object)** - Read-only property

**value** - (Enum as NaStates) - Choose from:

- **naOFF** (0) - Turns Frequency Offset OFF
- **naON** (1) - Turns Frequency Offset ON

**Return Type**

Enum

**Default**

naOFF (0)

**Examples**

```vbnet
chan.FrequencyOffsetState = naON 'Write
Foffset = chan.FrequencyOffsetState 'Read
```

**C++ Syntax**

```cpp
HRESULT FrequencyOffsetState (tag NAStates *pState);
HRESULT FrequencyOffsetState (tag NAStates newState)
```

**Interface**

IChannel2

|CalSet3
## FrequencyStep Property

**Description**
Sets the frequency step size across the selected frequency range. This effectively sets the number of data points. Available ONLY when `SweepType = Linear`.

**VB Syntax**
```
object.FrequencyStep = value
```

**Variable**
- **object** (Type) - Description
  - A Channel (`object`)
- **value** (double) - Frequency step size in Hertz. Select any value up to the frequency range of the analyzer.

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```
chan.FrequencyStep = 1e9 'sets the frequency step size of a linear sweep for the channel object -Write
freqstep = chan.FrequencyStep 'Read
```

**C++ Syntax**
```
HRESULT get_FrequencyStep(double *pVal)
HRESULT put_FrequencyStep(double newVal)
```

**Interface**
IChannel24
**FrequencyType Property**

**Description**
Sets and returns the frequency range to use for receiver leveling. On the user interface, this is the "Receiver frequency is determined by:" setting.

**VB Syntax**

```vbnet
RxLevel.FrequencyType(srcPort) = value
```

**Variable**

- **(Type)** - Description
  - **RxLevel** - A ReceiverLeveling Object
  - **srcPort** - (Long Integer) Source port for Receiver Leveling.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number.
To learn more see [Remotely Specifying a Source Port](#).

**value**

- **naAutoFrequency** (0) - always uses the frequency range that is assigned to the measurement receiver.
- **naInputFrequency** (3) - Mixer/Converter input frequency range.
- **naOutputFrequency** (4) - Mixer/Converter input frequency range.
- **naReceiverFrequency** (1) - FOM Receiver frequency range.
- **naSourceFrequency** (2) - FOM Source frequency range

**Return Type**
Enum

**Default**
**naAutoFrequency** (0)

**Examples**

```vbnet
rxLevel.FrequencyType (1) = naAutoFrequency ' Write
value = rxLevel.FrequencyType 2' Read
```

**C++ Syntax**

```cpp
HRESULT get_FrequencyType(long port, tagNAFrequencyType* pVal);
HRESULT put_FrequencyType(long port, tagNAFrequencyType newVal);
```

**Interface**
IRxLevelingConfiguration3
### FullyCorrectedPorts Property

**Description**
Sets and returns the selected ports to include in a full NPort correction. All other ports are corrected with enhanced response calibration if available. [Learn more](#).

**Note:** The `CorrectionSubsettingState` must be set to ON to enable the full command.

**VB Syntax**
```
value = corrMethods.FullyCorrectedPorts
```

**Variable**
*(Type)* - Description

- **value** (Variant) Variable to store a comma separated list of ports to include in the full correction.

**corrMethods**  CorrectionMethods (object)

**Return Type**  Variant

**Default**  All ports included

**Example**  
16-port VNA with an active 16-port calibration
```
corrMethods CorrectionSubsettingState = True
portlist = Array(1,2,3)
portlist = corrMethods.FullyCorrectedPorts
```

Result: Full correction on ports 1, 2, and 3
All other port parameters are uncorrected

**C++ Syntax**
```
HRESULT get_FullyCorrectedPorts(VARIANT *portList);
HRESULT put_FullyCorrectedPorts(VARIANT portList);
```

**Interface**  ICorrectionMethods2
## Gain Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Gain result of the PNOP marker search.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pnop Gain</td>
<td>Pnop Out - Pnop In.</td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
gain = pnop.Gain
```

**Variable**

- `gain` *(double)* - Variable to store returned value
- `pnop` A PNOP *(object)*

**Return Type**

Double

**Default**

Not applicable

**Examples**

```vbnet
gain = pnop.Gain 'Read
```

See example program

**C++ Syntax**

```cpp
HRESULT get_Gain(double* pNewVal)
```

**Interface**

IPNOP
# GainLinear Property

This property returns the Linear Gain result of a PSat marker search.

**Description**

Gain Linear = Marker 1 - Y-axis value MINUS X-axis value

**VB Syntax**

```vbnet
gainLin = pSat.GainLinear
```

**Variable**

- `gainLin` *(double)* - Variable to store returned value
- `pSat` - A PSaturation (object)

**Return Type**

Double

**Default**

Not applicable

**Examples**

```vbnet
gainLin = pSat.GainLinear
```

See example program

**C++ Syntax**

```cpp
HRESULT get_GainLinear(double* pNewVal)
```

**Interface**

IPSaturation
## GainMax Property

**Description**
Returns the GainMax result of the PNOP or PSAT marker search.

\[
\text{Gain Max} = \text{PMax Out} - \text{PMax In}
\]

**VB Syntax**

\[
gainMax = pMarker.GainMax
\]

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gainMax</td>
<td>(double) - Variable to store returned value</td>
</tr>
<tr>
<td>pMarker</td>
<td>A PNOP (object) or PSaturation (Object)</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
Not applicable

**Examples**

\[
gainMax = pMarker.GainMax 'Read
\]

See example program

**C++ Syntax**

HRESULT get_GainMax(double* pNewVal)

**Interface**
IPNOP or IPSaturation
## GainSaturation Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the GainSaturation result of the PSAT marker search.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gain Sat</td>
<td>$\text{Gain Sat} = \text{Psat Out} - \text{Psat In}$</td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td>$\text{Gain Sat} = \text{pSat.GainSaturation}$</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>$\text{Gain Sat}$</td>
<td>(double) - Variable to store returned value</td>
</tr>
<tr>
<td>$\text{pSat}$</td>
<td>A PSaturation (Object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>$\text{Gain Sat} = \text{pSat.GainSaturation} \ 'Read'$</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_GainSaturation(double* pNewVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPSaturation</td>
</tr>
</tbody>
</table>
### GeneratedCalsets Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the cal set names that were produced by the cal all session.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>csets = calAll.GeneratedCalsets</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>csets</code> (Variant Array) Variable to store the returned cal set names.</td>
</tr>
<tr>
<td><code>calAll</code></td>
<td>A CalibrateAllChannels (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant Array</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>csets = calAll.GeneratedCalsets</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_GeneratedCalsets (VARIANT*, propNames);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>CalibrateAllChannels</td>
</tr>
</tbody>
</table>

See example program
# FrequencySpan Property

**Description**
Sets or returns the frequency span of the channel.

Sets or returns the frequency span of the segment.

**VB Syntax**
```vbnet
object.FrequencySpan = value
```

**Variable (Type) - Description**
- **object**
  - A Channel (object)
  - or
  - A Segment (object)

- **value**
  - (double) - Frequency span in Hertz. Choose any number between 70 (minimum) and maximum frequency span of the analyzer.

**Return Type**
Double

**Default**
Full frequency span of the analyzer

**Examples**
- `chan.FrequencySpan = 4.5e9 'sets the frequency span of a linear sweep for the channel object  -Write`
- `freqspan = chan.FrequencySpan 'Read`

**C++ Syntax**
```cpp
HRESULT get_FrequencySpan(double *pVal)
HRESULT put_FrequencySpan(double newVal)
```

**Interface**
- IChannel
- ISegment
## Shape Property

**Description**
Specifies the shape of the gate filter.

**VB Syntax**
`gat.Shape = value`

**Variable**
- **(Type)** - Description
  - `gat` - A Gating (object)
  - `value` - (enum NAGateShape) - Choose from:
    - 0 - naGateShapeMaximum
    - 1 - naGateShapeWide
    - 2 - naGateShapeNormal
    - 3 - naGateShapeMinimum

**Return Type**
NAGateShape

**Default**
2 - Normal

**Examples**
- `gat.Shape = naGateShapeMaximum` 'Write
- `filterShape = gat.Shape` 'Read

**C++ Syntax**
- HRESULT get_Shape(tagNAGateShape *pVal)
- HRESULT put_Shape(tagNAGateShape newVal)

**Interface**
IGating
# Type (gate) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the type of gate filter used.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>gat.Type = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>gat</code></td>
<td>A Gating (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum NAGateType) - Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <code>naGateTypeBandpass</code> - Includes (passes) the range between the start and stop times.</td>
</tr>
<tr>
<td></td>
<td>1 - <code>naGateTypeNotch</code> - Excludes (attenuates) the range between the start and stop times.</td>
</tr>
<tr>
<td>Return Type</td>
<td>NAGateType</td>
</tr>
<tr>
<td>Default</td>
<td>Bandpass</td>
</tr>
<tr>
<td>Examples</td>
<td><code>gate.Type = naGateTypeNotch 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>filterType = gate.Type 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_Type(tagNAGateType *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Type(tagNAGateType newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IGating</td>
</tr>
</tbody>
</table>
# GPIBAddress Property

**Description**
Sets and returns the VNA GPIB address on the talker/listener bus.

**VB Syntax**
```
app.GPIBAddress (bus) = value
```

**Variable**
- **app** (Type) - Description
  - **app** An Application (object)
- **bus** (Short Integer) GPIB bus. MUST be set to 0 - the talker/listener bus.
- **value** (Short Integer) GPIB Address on the VNA. Choose a value between 0 and 30.

**Return Type**
Short Integer

**Default**
16

**Examples**
```
address=app.GPIBAddress(0) 'Read

app.GPIBAddress(0)=16 'Write
```

**C++ Syntax**
```
HRESULT get_GPIBAddress(short busIndex, short* address);

HRESULT put_GPIBAddress(short busIndex,short address);
```

**Interface**
IApplication8
**GPIBMode Property**

**Description**
Changes the analyzer to a GPIB system controller or a talker/listener on the bus. The analyzer must be the controller if you want to use it to send commands to other instruments. The analyzer must be a talker/listener if you want to send it commands from another PC.

*Note:* This command has no affect in VNAs with dedicated Controller and Talker/Listener GPIB connectors.

**VB Syntax**
```vbnet
app.GPIBMode value
```

**Variable**
(Shape) - Description
- `app` - An Application (object)
- `value` - (enum `NAGPIBMode`) - Choose either:
  - `0` - `naTalkerListener` - the analyzer is a talker / listener
  - `1` - `naSystemController` - the analyzer is the system controller

**Return Type**
Long Integer

**Default**
`0` - `naTalkerListener`

**Examples**
```vbnet
app.GPIBMode = naTalkerListener 'Write
mode = app.GPIBMode 'Read
```

**C++ Syntax**
```cpp
HRESULT get_GPIBMode(tagGPIBModeEnum* eGpibMode)
HRESULT put_GPIBMode(tagGPIBModeEnum eGpibMode)
```

**Interface**
`IApplication`
## GPIBPortCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of GPIB ports that are present on the VNA rear-panel.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.GPIBPortCount</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) - Variable to store the returned integer value of the number of</td>
</tr>
<tr>
<td></td>
<td>GPIB ports.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.GPIBPortCount 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_GPIBPortCount(long * gpibPorts);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities3</td>
</tr>
</tbody>
</table>
## Grid Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return the inner lines of all grid in all windows, and the grid frame in inactive windows for the VNA display or hardcopy print.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>colors/Grid = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>colors</code></td>
<td>A <code>ComColors (object)</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long Integer) - RGB color of the Grid pen.</td>
</tr>
</tbody>
</table>

Convert the three RGB colors to an integer as follows:

\[
\text{RGB} = R + (G \times 2^8) + (B \times 2^{16})
\]

To find the three RGB values from the Display Colors dialog, click **Change Color**, then **Define Custom Color**.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Long</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Display = 175,175,175</td>
</tr>
<tr>
<td></td>
<td>Print = 0,0,0 (Black)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
<th>R = 10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>G = 10</td>
</tr>
<tr>
<td></td>
<td>B = 10</td>
</tr>
<tr>
<td>RGB</td>
<td>( R + (G \times 2^8) + (B \times 2^{16}) )</td>
</tr>
<tr>
<td><code>colors/Grid</code></td>
<td>RGB <code>Write</code></td>
</tr>
<tr>
<td><code>color</code></td>
<td>= <code>colors/Grid</code> <code>Read</code></td>
</tr>
</tbody>
</table>

### C++ Syntax

```cpp
HRESULT get_Grid(long* pVal);
HRESULT put_Grid(long newVal);
```

**Interface**

`IColors`
GridLineType Property

**Description**
Set and return whether the grid lines are displayed in solid or dotted lines for all open windows. Grid lines are returned to solid when the VNA is Preset.

**VB Syntax**
`app.GridLineType = value`

**Variable**
- `app` [Type] Application Object [object]
- `value` [Enum as naLineType] - Choose from:
  
  - 0 - naLineTypeSolid
  - 1 - naLineTypeDotted

**Return Type**
Enum

**Default**
aLineTypeSolid

**Examples**
```
app.GridLineType = naLineTypeSolid 'Write
grid = app.GridLineType 'Read
```

**C++ Syntax**
```
HRESULT get_GridLineType(tag naLineType* pVal);
```
```
HRESULT put_GridLineType(tag naLineType newVal);
```

**Interface**
IApplication
HandshakeEnable Property

Description

Turns handshake ON / OFF.

To enable handshake, the main trigger enable must ALSO be set using Enable.

When ON, VNA acquisition waits indefinitely for the input line to be asserted before continuing with the acquisition.

**Note:** Use on PNA-X ONLY. Other models do NOT have an Aux Input.

VB Syntax

```vbnet
auxTrig.HandshakeEnable = state
```

Variable (Type) - Description

- `auxTrig` An AuxTrigger (object)
- `state` (boolean) -
  - *True* - Handshake enabled
  - *False* - Handshake NOT enabled

Return Type

Boolean

Default

False

Examples

```vbnet
auxTrig.HandshakeEnable = True 'Write
value = auxTrig.HandshakeEnable 'Read
```

C++ Syntax

```c
HRESULT get_HandshakeEnable(VARIANT_BOOL * enable);
HRESULT put_HandshakeEnable(VARIANT_BOOL enable);
```

Interface

IAuxTrigger
### Has2ndOrderTrace Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the state of whether or not 2nd order parameters are measured. True indicates 2nd order parameters are measured.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = imd.Has2ndOrderTrace</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>imd</code></td>
<td>A SweptIMD Object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>Variable to store returned state.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_Has2ndOrderTrace(VARIANT_BOOL* val)</td>
</tr>
<tr>
<td>Interface</td>
<td>ISweptIMD4</td>
</tr>
</tbody>
</table>
### HasDirectReceiverAccess Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns whether or not the analyzer has direct receiver access (front-panel jumpers).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( \text{value} = \text{cap}.\text{HasDirectReceiverAccess} )</td>
</tr>
<tr>
<td><strong>Variable</strong> Class/Type/Description</td>
<td>(Boolean) - Variable to store the returned value</td>
</tr>
<tr>
<td><strong>Variable</strong> Description</td>
<td>Value to store the returned value</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>( \text{value} = \text{cap}.\text{HasDirectReceiverAccess} )</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_HasDirectReceiverAccess(VARIANT_BOOL *present);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities12</td>
</tr>
</tbody>
</table>
# HasLowFrequencyExtension Property

**Description**  
Returns whether or not the VNA has the low frequency extension (LFE) installed. Learn more.

**VB Syntax**  
```
value = chan.HasLowFrequencyExtension
```

**Variable**  
(Type) - Description  
```
value (boolean)
```

- **False** - LFE is not installed.
- **True** - LFE is installed.

**cap**  
Capabilities (object)

**Return Type**  
Not applicable

```
hasLFE = cap.HasLowFrequencyExtension 'Read
```

**C++ Syntax**  
```
HRESULT get_HasLowFrequencyExtension(VARIANT_BOOL* hasLFE);
```

**Interface**  
ICapabilities16
### HasItem Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a value indicating whether the specified external devices is configured.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = extDevices.HasItem(name)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean) Variable to store one of the following returned values.</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>Item (name) is NOT present in the External Devices collection.</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>Item (name) IS present in the External Devices collection.</td>
</tr>
<tr>
<td><strong>extDevices</strong></td>
<td>An <code>ExternalDevices (collection)</code></td>
</tr>
<tr>
<td><strong>name</strong></td>
<td>(String) Name of External Device for which to search the collection.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>has = extDevices.HasItem('mysource')</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_HasItem(VARIANT Index, BOOL *pVal);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IExternalDevices</code></td>
</tr>
</tbody>
</table>
## HighAmplitude Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the High amplitude (voltage) of the pulse generator.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>extPulseGen.HighAmplitude = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>extPulseGen</code></td>
<td>An <code>ExternalPulseGenerator (object)</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Double)</em> Pulse Generator high amplitude voltage.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>5</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>extPulseGen.HighAmplitude = 4</code> <em>'Write</em></td>
</tr>
<tr>
<td></td>
<td><code>hi = extPulseGen.HighAmplitude</code> <em>'Read</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_HighAmplitude (double *pValue)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_HighAmplitude (double newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IExternalPulseGenerator</code></td>
</tr>
</tbody>
</table>
# HighestOrderProduct Property

**Description** Returns the highest order product that can be measured by Swept IMD.

**VB Syntax**
```
value = imd.HighestOrderProduct
```

**Variable** *(Type)* - Description
- **value** (Long Integer) Variable in which to store the returned value.
- **imd** A Swept IMD Object

**Return Type** Integer

**Default** **Always returns 9**

**Examples**
```
value = imd.HighestOrderProduct 'Read
```

**C++ Syntax**
```
HRESULT get_HighestOrderProduct(long *pVal)
```

**Interface** ISweptIMD
### HighestOrderProductInUse Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Returns the highest order product measured by SweptIMD.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = imd.HighestOrderProductInUse</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long Integer) Variable in which to store the returned value.</td>
</tr>
<tr>
<td><code>imd</code></td>
<td>A SweptIMD Object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = imd.HighestOrderProductInUse 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_HighestOrderProductInUse(long *pVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISweptIMD</td>
</tr>
</tbody>
</table>
ID Property

Returns the test set ID number. For GPIB testsets, the ID is equivalent to the GPIB address. For testset I/O testsets, the ID is the base address of the testset (0 for the first testset, 1 for the second, and so on).

**VB Syntax**
```
value = tset.ID
```

**Variable (Type) - Description**
- `value` (Long) variable to store the returned information.
- `testsets(I)` A TestsetControl object.

OR

An E5091Testset object.

**Return Type** Long Integer

**Default** Not Applicable

**Examples**
```
value = testset1.ID
```

See E5091A Example Program

See External Testset Program

**C++ Syntax**
```
HRESULT get_ID(long *idNumber);
```

**Interface** ITestsetControl

IE5091Testset
### IDQuery Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the <strong>ID Query command</strong> for an external DC Source and an external DC Meter.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>extDC.IDQuery = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>extDC</code></td>
<td>A <strong>ExternalDCDevice (object)</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(String) The SCPI command for returning DC Source and DC Meter ID string.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td><code>&quot;*IDN?&quot;</code></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>extDC.IDQuery = &quot;*IDN?&quot; 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = extDC.IDQuery 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_IDQuery(BSTR* cmd);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_IDQuery(BSTR cmd);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IExternalDCDevice3</td>
</tr>
</tbody>
</table>
IDString Property

Description
Returns the ID of the analyzer, including the Model number, Serial Number, and the Software revision number.

Note: Beginning with Rev 6.01, this command now returns the software revision with 6 digits instead of 4. For example, A.06.01.02.

VB Syntax
value = app.IDString

Variable
app (Type) - Description
An Application (object)

value (string) - variable to contain the returned ID string

Return Type
String

Default
Not Applicable

Examples
id = app.IDString

C++ Syntax
HRESULT IDString(BSTR* IDString)

Interface
IApplication
IFBandwidthOption Property

**Description**
Enables the IFBandwidth to be set on individual sweep segments. This property must be set True **before** `seg.IFBandwidth = value` is sent. Otherwise, this command will be ignored.

**VB Syntax**
```
segs.IFBandwidthOption = value
```

**Variable**
- **Type**
  - `segs` A Segments collection *(object)*
  - `value` *(boolean)*

**Examples**
```
segs.IFBandwidthOption = True 'Write
IFOption = IFBandwidthOption 'Read
```

**C++ Syntax**
```
HRESULT get_IFBandwidthOption(VARIANT_BOOL *pVal)
HRESULT put_IFBandwidthOption(VARIANT_BOOL newVal)
```

**Interface**
ISegments
IFBandwidth Property

Description

Sets or returns the IF Bandwidth of the channel.

Sets or returns the IF Bandwidth of the segment.

Returns the IF Bandwidth used in the Cal Set.

VB Syntax

```
object.IFBandwidth = value
```

Variable (Type) - Description

- **object**
  - Channel (object)
  
  or
  
  Segment (object)
  
  or
  
  CalSet (object) - Read-only property

- **value**
  - (double) - IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the VNA model. (Click to see the lists.) If an invalid number is specified, the analyzer will round up to the closest valid number.

Return Type

Double

Default

Varies with VNA model.

Examples

```
chan.IFBandwidth = 3e3 'sets the IF Bandwidth of the channel object to 3 kHz. -Write
seg.IFBandwidth = 5 'sets the IF Bandwidth of the segment to 5 Hz. -Write
ifbw = chan.IFBandwidth -Read
```

C++ Syntax

```
HRESULT get_IFBandwidth(double *pVal);
```

```
HRESULT put_IFBandwidth(double newVal);
```

Interface

IChannel

ISegment

|CalSet3
## IFBW Property

**Description**  
Sets and returns the IFBW for a Cal All calibration. Learn more about this setting.

**VB Syntax**  
`calAll.IFBW = value`

**Variable (Type) - Description**

- **calAll**  
  A `CalibrateAllIFBW (object)`

- **value**  
  (Double) IFBW in Hz.

**Return Type**  
Variant

**Default**  
1 kHz

**Examples**

```vba
calAll.IFBW = 10e3 'sets IFBW to 10 kHz

ifbw = calAll.IFBW 'returns the ifbw setting
```

**C++ Syntax**

```cpp
HRESULT get_IFBW (Double *Val)

HRESULT put_IFBW (Double newVal)
```

**Interface**  
ICalibrateAllChannels
**IFBWList Property**

**Description**  Returns a list of supported IFBW values.

**VB Syntax**  

\[ \text{value} = \text{cap}.\text{IFBWList} \]

**Variable**  

- **(Type)** - Description
  - **value**  (Variant) - Variable to store the returned array of IFBW values
  - **cap**  A Capabilities (Object)

**Return Type**  Variant array

**Default**  Not Applicable

**Examples**  

'Read the supported IFBW values

```vbnet
Set app = CreateObject("AgilentPNA835x.Application")
Set cap = app.Capabilities
list=cap.IFBWList
For i = 0 To UBound(list)
    msg = msg & list(i) & vbCrLf
Next
MsgBox msg
```

**C++ Syntax**  

```csharp
HRESULT get_IFBWList(Variant *value);
```

**Interface**  ICapabilities8
# IFDenominator Property

**Description**  
Sets or returns the denominator value of the IF Fractional Multiplier. Only applies to 2 stage mixers.  
If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**  
```vbnet
conv.IFDenominator = value
```

**Variable**  
*conv*  
A Converter (object)

*value*  
(long) IF Denominator value.

**Return Type**  
Long

**Default**  
1

**Examples**  
```vbnet
Print mixer.IFDenominator 'prints the value of the IFDenominator
```

**C++ Syntax**  
```c++
HRESULT get_IFDenominator(long *pVal)

HRESULT put_IFDenominator(long newVal)
```

**Interface**  
IConverter4
### IFNumerator Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the numerator value of the IF Fractional Multiplier. Only applies to 2 stage mixers. If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>conv.IFNumerator = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>conv</code></td>
<td>A Converter (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) IF Numerator value.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>Print mixer.IFNumerator</code> 'prints the value of the IFNumerator'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_IFNumerator(long *pVal)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_IFNumerator(long newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IConverter4</td>
</tr>
</tbody>
</table>
### IFFrequency Property

**Description**
Sets and returns the IF frequency for ALL receiver paths being used for the specified channel. To set this frequency, **IFFrequencyModeProperty** must be set to OFF (Manual).

**VB Syntax**

```
IfConfig.IFFrequency = value
```

**Variable**

- **Type**: Description
- **Description**: An IFConfiguration (object)
- **Value**: (double) IF Frequency. Use **MaximumIFFrequency** and **MinimumIFFrequency** to determine the range of value for this command.

**Return Type**: Double

**Default**: 9 MHz

**Examples**

```
IfConfig.IFFrequency = 9.3e6
```

**C++ Syntax**

```
HRESULT get_IFFrequency (double *pVal);

HRESULT put_IFFrequency (double pVal);
```

**Interface**: IFConfiguration3
# IFFrequencyMode Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns method for specifying the way the IF Frequency is determined.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>IfConfig.IFFrequencyMode = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>IfConfig</strong> - An <strong>IFConfiguration</strong> <em>(object)</em></td>
</tr>
<tr>
<td></td>
<td><code>value</code> <em>(enum as NAModes)</em> IF Frequency mode. Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>0 - naAUTO</strong> The VNA determines the setting for the IF frequency. The IF</td>
</tr>
<tr>
<td></td>
<td>frequency is based on many VNA settings, including measurement frequency.</td>
</tr>
<tr>
<td></td>
<td>Therefore, it is NOT possible to read the IF frequency that is being used.</td>
</tr>
<tr>
<td></td>
<td><strong>1 - naMANUAL</strong> (use <strong>IFFrequency Property</strong> to manually set frequency.)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum</td>
</tr>
<tr>
<td>Default</td>
<td>0 naAuto</td>
</tr>
<tr>
<td>Examples</td>
<td><code>IfConfig.IFFrequencyMode = naMANUAL</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_IFFrequencyMode(tagNAModes* pdspMode);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_IFFrequencyMode(tagNAModes* pdspMode);</td>
</tr>
<tr>
<td>Interface</td>
<td>IFConfiguration3</td>
</tr>
</tbody>
</table>
Write/Read About Mixer Configuration

IFSideband Property

Description: When two LOs are used, sets or returns whether to select the sum or difference for the IF1 product. (Input + or - LO1 = IF1)

- This setting corresponds to the buttons on LO1.
- Also set OutputSideband to LOW or HIGH to determine the output frequency of the mixer.
- This setting is ignored when ONE LO is used.
- If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

Note: There is also an IFSideband_Property on the Mixer Interface.

VB Syntax: 
```
obj.IFSideband = value
```

Variable (Type) - Description

- **obj** - A Converter Object
- **value** - (enum as ConverterSideBand) - Choose from:
  - 0 or **naLowSide** Minus (-) on the Mixer setup dialog
  - 1 or **naHighSide** Plus (+) on the Mixer setup dialog

Return Type: Enum as ConverterSideBand

Default: 0 - **naLowSide**

Examples: 
```
conv.IFSideband = naLowSide
```

C++ Syntax:
```
HRESULT get_IFSideband(ConverterSideBand *pVal)

HRESULT put_IFSideband(ConverterSideBand newVal)
```

Interface: IConverter
**IFSideband Property**

**Description**

When two LOs are used, sets or returns whether to select the sum or difference for the IF1 product. (Input + or - LO1 = IF1)

- This setting corresponds to the buttons on LO1.
- Also set OutputSideband to LOW or HIGH to determine the output frequency of the mixer.
- This setting is ignored when ONE LO is used.
- If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**Note:** There is also an IFSideband_Property on the Converter Object.

**VB Syntax**

```vb
mixer.IFSideband = value
```

**Variable**

- **mixer** (Type) - Description
  - A Mixer (object)
- **value** (enum as FCASideBand) - Choose from:
  - 0 or LOW  Minus (-) on the Mixer setup dialog
  - 1 or HIGH  Plus (+) on the Mixer setup dialog

**Return Type**

Enum as FCASideBand

**Default**

0 - LOW

**Examples**

```vb
Print mixer.IFSideband  'prints the value of the IFSideband
```

**C++ Syntax**

```cpp
HRESULT get_IFSideband(FCASideBand *pVal)

HRESULT put_IFSideband(FCASideBand newVal)
```

**Interface**

IMixer
### IFStartFrequency Property

**Description**  
Sets or returns the start frequency value of the mixer IF frequency.

Only applies to 2 stage mixers.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**  
`mixer.IFStartFrequency = value`

**Variable (Type) - Description**

- `mixer`  
  A Converter (object)

- `value`  
  (double) - Frequency in Hertz.

**Return Type**  
Double

**Default**  
Not Applicable

**Examples**

```vbnet
Print mixer.IFStartFrequency 'prints the value of the IFStartFrequency
```

**C++ Syntax**

```c++
HRESULT get_IFStartFrequency(double *pVal)

HRESULT put_IFStartFrequency(double newVal)
```

**Interface**  
IMixer
## IFStopFrequency Property

**Description**  Sets or returns the stop frequency value of the mixer IF frequency.

Only applies to 2 stage mixers.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**  

```vbnet
mixer.IFStopFrequency = value
```

**Variable**  

- `mixer`  A `Converter` (object)
- `value`  (double) - IF stop frequency in Hertz.

**Return Type**  Double

**Default**  Not Applicable

**Examples**  

```vbnet
Print mixer.IFStopFrequency 'prints the value of the IFStopFrequency
```

**C++ Syntax**  

```csharp
HRESULT get_IFStopFrequency(double *pVal)

HRESULT put_IFStopFrequency(double newVal)
```

**Interface**  IMixer
### ImageRejectMethod Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the image reject value.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>sa.ImageRejectMethod = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td></td>
<td><code>sa</code> A SpectrumAnalyzer <em>(object)</em></td>
</tr>
<tr>
<td></td>
<td><code>value</code> <em>(Enum as NASAImageRejectMethod)</em> Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>0</strong> - naIRNoneHigh</td>
</tr>
<tr>
<td></td>
<td><strong>1</strong> - naIRNoneLow</td>
</tr>
<tr>
<td></td>
<td><strong>2</strong> - naIRMin</td>
</tr>
<tr>
<td></td>
<td><strong>3</strong> - naIRNormal</td>
</tr>
<tr>
<td></td>
<td><strong>4</strong> - naIBetter</td>
</tr>
<tr>
<td></td>
<td><strong>5</strong> - naIRMax</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum</td>
</tr>
<tr>
<td>Default</td>
<td><strong>3</strong> - naINormal</td>
</tr>
<tr>
<td>Examples</td>
<td><code>sa.ImageRejectMethod = naIRNormal    'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = sa.ImageRejectMethod 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT put_ImageReject(tagNASAImageRejectMethod algorithm);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_ImageReject(tagNASAImageRejectMethod* algorithm);</td>
</tr>
<tr>
<td>Interface</td>
<td>ISpectrumAnalyzer</td>
</tr>
</tbody>
</table>
ImageRejectStrength Property

Description
Sets and returns the image rejection strength. During the image rejection process, several LO acquisitions overlap at the same RF frequency. As a result, different RF signal values can be returned. This command sets the acceptable power differences between LOs in determining actual signals.

VB Syntax

```
sa.ImageRejectStrength = value
```

Variable (Type) - Description

- **sa**: A SpectrumAnalyzer (object)
- **value**: (enum NASAStrength) - Choose from:
  - 0: naSAWeak: 3 dB (approximate value, depends on RBW)
  - 1: naSANormal: 1 dB (approximate value, depends on RBW)
  - 2: naSAStrong: 0.5 dB (approximate value, depends on RBW)

Return Type
Enum as NASAStrength

Default
naSANormal

Examples

```
sa.ImageRejectStrength = naSAStrong 'Write
Strength = sa.ImageRejectStrength 'Read
```

C++ Syntax

```
HRESULT get_ImageRejectStrength(tagNASAStrength* pVal)
HRESULT put_ImageRejectStrength(tagNASAStrength newVal)
```

Interface
ISpectrumAnalyzer4
### ImpedanceStates Property

**Description**
Sets the number of impedance states to use during calibrated measurements.

**VB Syntax**

```vbnet
noise.ImpedanceStates = value
```

**Variable (Type) - Description**

- `noise` (A `NoiseFigure` object)
- `value` (double) - Impedance states. Choose between 4 and the maximum number allowed by the noise tuner device. If the specified number exceeds the capability of the device, the measurement will use the maximum number of states the device allows.

**Return Type**
Double

**Default**
4

**Examples**

- `noise.ImpedanceStates = 10` 'Write
- `AvgNoise = noise.ImpedanceStates` 'Read

**C++ Syntax**

```
HRESULT get_ImpedanceStates(double* pVal)
HRESULT put_ImpedanceStates(double newVal)
```

**Interface**
`INoiseFigure`
ImpulseWidth Property

Description: Sets or returns the Impulse Width of Time Domain transform windows.

VB Syntax: `trans.ImpulseWidth = value`

Variable (Type) - Description:
- `trans` A Transform (object)
- `value` (double) - Impulse Width in seconds. Range of settings depends on the frequency range of your analyzer.

Return Type: Double

Default: .98 / Default Span

Examples:
- `trans.ImpulseWidth = 200e-12` 'sets the Impulse width of a transform window -Write
- `IW = trans.ImpulseWidth` 'Read

C++ Syntax:
- `HRESULT get_ImpulseWidth(double *pVal)`
- `HRESULT put_ImpulseWidth(double newVal)`

Interface: ITransform
### IMToneIFBandwidth Property

**Description**
Sets and returns the IF Bandwidth for measurement of the intermodulation products.

**VB Syntax**
```
imd.IMToneIFBandwidth = value
```

**Variable (Type) - Description**
- **imd** (A SweptIMD Object)
- **value** (Double) IF Bandwidth in Hz. Choose from: 1 | 2 | 3 | 5 | 7 | 10 | 15 | 20 | 30 | 50 | 70 | 100 | 150 | 200 | 300 | 500 | 700 | 1k | 1.5k | 2k | 3k | 5k | 7k | 10k | 15k | 20k | 30k | 50k | 70k | 100k | 150k | 200k | 280k | 360k | 600k

Learn more about setting IFBW for IMD.

If an invalid number is specified, the analyzer will round up to the closest valid number.

**Return Type**
Double

**Default**
1 kHz

**Examples**
```
imd.IMToneIFBandwidth = 2e3 'Write
value = imd.IMToneIFBandwidth 'Read
```

**C++ Syntax**
- HRESULT get_IMToneIFBandwidth(double *pVal)
- HRESULT put_IMToneIFBandwidth(double newVal)

**Interface**
ISweptIMD
InactiveLabels Property

**Description**
Set and return the Inactive (not selected) Window Labels for the VNA display or hardcopy print.

**VB Syntax**
```
colors.InactiveLabels = value
```

**Variable**
- **(Type)** - Description
  - colors
    - A `ComColors (object)`
  - value
    - `(Long Integer)` - RGB color of the Inactive Labels pen.

Convert the three RGB colors to an integer as follows:

```
RGB = R+(G*2^8)+(B*2^16)
```

To find the three RGB values from the Display Colors dialog, click **Change Color**, then **Define Custom Color**.

**Return Type**
- Long

**Default**
- Display = 160,160,160
- Print = 0,0,0 (Black)

**Examples**
```
R = 10
G = 10
B = 10
RGB = R+(G*2^8)+(B*2^16)
colors.InactiveLabels = RGB 'Write
color = colors.InactiveLabels 'Read
```

**C++ Syntax**
```
HRESULT get_InactiveLabels(long* pVal);
HRESULT put_InactiveLabels(long newVal);
```

**Interface**
- `IColors`
### Include2ndOrderProduct Property

**Description**  
Sets and returns whether to include the second order products in the calibration. These frequencies of these products can be far from the main tones.

**VB Syntax**  
```vbnet
imd.Include2ndOrderProduct = value
```

**Variable**  
- `(Type)` - `Description`  
- `imd` - A `SweptIMDCal (object)`  
- `value` - (Boolean) Choose from:
  - **False** - Do NOT include 2nd order products  
  - **True** - Include 2nd order products

**Return Type**  
Boolean

**Default**  
**False** - Do NOT include 2nd order products

**Examples**  
```vbnet
imd.Include2ndOrderProduct = true 'Write
incl = imd.Include2ndOrderProduct 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_Include2ndOrderProduct(VARIANT_BOOL * Val)
HRESULT put_Include2ndOrderProduct(VARIANT_BOOL newVal)
```

**Interface**  
ISweptIMD
# IncludePort Property

Sets and returns the enable state for the specified port.

## Description
Sets and returns the enable state for the specified port.

## VB Syntax
```vb
phaseRef.IncludePort (n) = value
```

## Variable

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object</code></td>
<td><code>PhaseReferenceCalibration (object)</code></td>
</tr>
<tr>
<td><code>n</code></td>
<td>Port number to enable or disable.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>Port enable state. Choose from:</td>
</tr>
</tbody>
</table>

- `True` - Enable port <n>
- `False` - Disable port <n>

## Return Type
Boolean

## Default

- Ports 1 and 2 are enabled.
- Ports 3 and 4 (if present) are disabled

## Examples
```vb
phaseRef.IncludePort(3) = True
value = phaseRef.IncludePort(3) 'Read
```

## C++ Syntax
```cpp
HRESULT get_IncludePort(Long port, VARIANT_BOOL *value)

HRESULT put_IncludePort(Long port, VARIANT_BOOL value)
```

## Interface
IPhaseReference2
# IncludeReverseSweep Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets whether to include SC12 sweeps during measurements.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>obj.IncludeReverseSweep = bool</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td></td>
<td><code>obj</code> A Converter (object)</td>
</tr>
<tr>
<td></td>
<td><code>bool</code> (Boolean) -</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Include the SC12 (reverse) sweep.</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Do NOT Include the SC12 (reverse) sweep.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>True</td>
</tr>
<tr>
<td>Examples</td>
<td><code>obj.IncludeReverseSweep = True</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_IncludeReverseSweep(VARIANT_BOOL * val);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_IncludeReverseSweep(VARIANT_BOOL val);</td>
</tr>
<tr>
<td>Interface</td>
<td>IMixer12</td>
</tr>
</tbody>
</table>
### IncludeUnknownMixer Property

**Description**
Sets and returns the state of Unknown Mixer calibration.

**VB Syntax**

```vbnet
phaseRef.IncludeUnknownMixer = value
```

**Variable**

- **(Type)** - Description
  - `object` PhaseReferenceCalibration (object)
  - `value` (Boolean) Unknown Mixer cal state. Choose from:

  - **True** - Enable Unknown Mixer cal. The start frequency becomes 10 MHz and can NOT be changed.
  - **False** - Disable Unknown Mixer cal.

**Return Type**
Boolean

**Default**
False

**Examples**

- `phaseRef.IncludeUnknownMixer = True`
- `value = phaseRef.IncludeUnknownMixer 'Read`

**C++ Syntax**

- `HRESULT get_InvokeUnknownMixer(VARIANT_BOOL *value)`
- `HRESULT put_InvokeUnknownMixer(VARIANT_BOOL value)`

**Interface**
IPhaseReference2
## IndependentPowerCalibration Property

<table>
<thead>
<tr>
<th>Description</th>
<th>This property returns a handle to the <code>IndependentPowerCalibration</code> object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>calAll.IndependentPowerCalibration()</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>calAll</code></td>
<td>A <code>CalibrateAllChannels</code> <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td><code>IIndependentPowerCalibration</code></td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Example</td>
<td><code>CalibrateAllChannels.IndependentPowerCalibration()</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT IndependentPowerCalibration( IIndependentPowerCalibration **obj);</code></td>
</tr>
<tr>
<td>Interface</td>
<td><code>ICalibrateAllChannels2</code></td>
</tr>
</tbody>
</table>
**IndexState Property**

**Description**
Determines the control of Material Handler connector Pin 20.

**VB Syntax**
`handler.IndexState = value`

**Variable**
- **(Type)** - Description
- **handler** - A `Handler I/O` object
- **value** - A `boolean`
  
  - False - Pin 20 is controlled by Output Port B6
  
  - True - Pin 20 is controlled by the Index signal

**Return Type**
Boolean

**Default**
False

**Examples**

```
handler.IndexState = False 'Write
bState = handler.IndexState 'Read
```

**C++ Syntax**

```
HRESULT put_IndexState (BOOL *pVal);
HRESULT get_IndexState (BOOL newVal);
```

**Interface**
`IHWMaterialHandlerIO2`
**InitCmd Property**

**Description**
Sets and returns the Enable I/O command for an external DC Source and an external DC Meter.

**VB Syntax**
`extDC.InitCmd = value`

**Variable**
`extDC` (Type) - Description

`value` (String) The SCPI command used to enable the DC Source and DC Meter.

**Return Type**
String

**Default**
"" Empty String

**Examples**
```vbnet
extDC.InitCmd = "OUTP ON" 'Write
value = extDC.InitCmd 'Read
```

**C++ Syntax**
```cpp
HRESULT get_InitCmd( BSTR* cmd);
HRESULT put_InitCmd( BSTR cmd);
```

**Interface**
IExternalDCDevice3
# InputA Property - **Obsolete**

This property has NO replacement and no longer works correctly. (Sept. 2004)

Sets a Port Extension value for Receiver A

**VB Syntax**

```vbnet
portExt.InputA = value
```

**Variable**

- **(Type)** - Description
- **portExt** - A Port Extension **(object)**
- **value** - Port Extension value in seconds. Choose any number between -10 and 10

**Return Type**

Double

**Default**

0

**Examples**

```vbnet
portExt.InputA = 10e-6 'Write
inA = portExt.InputA 'Read
```

**C++ Syntax**

```cpp
HRESULT get_InputA(double *pVal)
HRESULT put_InputA(double newVal)
```

**Interface**

IPortExtension
### InputB Property - Obsolete

**Description**

This property has NO replacement and no longer works correctly. (Sept. 2004)

Sets the Port Extension value for Receiver B

**VB Syntax**

```vbnet
portExt.InputB = value
```

**Variable (Type) - Description**

- `portExt` (object) - A Port Extension
- `value` (double) - Port Extension value in seconds. Choose any number between -10 and 10

**Return Type**

Double

**Default**

0

**Examples**

```vbnet
portExt.InputB = 10e-6 'Write
inB = portExt.InputB 'Read
```

**C++ Syntax**

```cpp
HRESULT get_InputB(double *pVal)
HRESULT put_InputB(double newVal)
```

**Interface**

IPortExtension
InputC Property **Obsolete**

**Description**

*This property has NO replacement and no longer works correctly. (Sept. 2004)*

Sets the Port Extension value for Receiver C

**VB Syntax**

```vbnet
    portExt.InputC = value
```

**Variable**

*Type* - Description

- `portExt` - A Port Extension *(object)*
- `value` - Port Extension value in seconds. Choose any number between -10 and 10

**Return Type**

Double

**Default**

0

**Examples**

```vbnet
    portExt.InputC = 10e-6 'Write
    inc = portExt.InputC 'Read
```

**C++ Syntax**

```c
    HRESULT get_InputC(double *pVal)
    HRESULT put_InputC(double newVal)
```

**Interface**

IPortExtension
InputDenominator Property

Description
Sets or returns the denominator value of the Input Fractional Multiplier.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

VB Syntax

```
obj.InputDenominator = value
```

Variable

- **obj**
  - A Mixer Interface pointer to the Measurement (object)
  - Or
  - A Converter Object

- **value**
  - **(Long)** - Input denominator value.

Return Type

Long

Default

1

Examples

```
Print mixer.InputDenominator 'prints the value of the InputDenominator
```

C++ Syntax

```
HRESULT get_InputDenominator(long *pVal)

HRESULT put_InputDenominator(long newVal)
```

Interface

IMixer

IConverter
### InputFixedFrequency Property

**Description**
Sets or returns the mixer fixed Input frequency value.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**

```vbnet
obj.InputFixedFrequency = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>obj</code></td>
<td>A Mixer Interface pointer to the Measurement (object)</td>
</tr>
<tr>
<td>Or</td>
<td>A Converter Object</td>
</tr>
</tbody>
</table>

**value**

*(double)* - Input Fixed Frequency in Hertz.

**Return Type**
Double

**Default**
Not Applicable

**Examples**

```vbnet
mixer.InputFixedFrequency = 1e9
```

**C++ Syntax**

```cpp
HRESULT get_InputFixedFrequency(double *pVal)
HRESULT put_InputFixedFrequency(double newVal)
```

**Interface**

- IMixer6
- IConverter
IsInputGreaterThanLO Property

Description
Specifies whether to use the Input frequency that is greater than the LO or less than the LO. To learn more, see the mixer setup dialog box help.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

VB Syntax

```vbnet
obj.IsInputGreaterThanLO (LO) = bool
```

Variable (Type) - Description

- **obj** - A Mixer Interface pointer to the Measurement (object)
- Or
- A Converter Object
- **LO** - LO stage number
  - Choose from 1 (default) or 2
- **bool** -
  - **True** - Use the Input that is Greater than the specified LO.
  - **False** - Use the Input that is Less than the specified LO.

Return Type
Boolean

Default
True

Examples

```vbnet
mixer.IsInputGreaterThanLO (1) = True
```

C++ Syntax

```cpp
HRESULT get_IsInputGreaterThanLO(VARIANT_BOOL * val);
HRESULT put_IsInputGreaterThanLO(VARIANT_BOOL val);
```

Interface
IMixer2
IConverter
## InputNumerator Property

**Description**
Sets or returns the numerator value of the Input Fractional Multiplier.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the `Calculate` and `Apply` commands as you would do from the user interface.

**VB Syntax**

```vbnet
obj.InputNumerator = value
```

**Variable**

- **Type** - `obj` - A Mixer Interface pointer to the Measurement (object)
- Or
- A Converter Object

- **value** - `Long` - Input numerator value.

**Return Type**

`Long`

**Default**

1

**Examples**

```vbnet
Print mixer.InputNumerator 'prints the value of the InputNumerator
```

**C++ Syntax**

- `HRESULT get_InputNumerator(long *pVal)`
- `HRESULT put(InputNumerator(long newVal)`

**Interface**

- `IMixer`
- `IConverter`
## InputPower Property

**Description**
Sets or returns the value of the Input Power.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the `Calculate` and `Apply` commands as you would do from the user interface.

**VB Syntax**
```vbnet
obj.InputPower = value
```

**Variable (Type) - Description**

- `obj` A Mixer Interface pointer to the Measurement (object)
  
  Or

- A Converter Object

- `value` (double) - Input power in dBm.

**Return Type**
Double

**Default**
-15 dBm for IMixer
-20 dBm for IConverter

**Examples**
`Print mixer.InputPower 'prints the value of the InputPower`

**C++ Syntax**

```cpp
HRESULT get_InputPower(double *pVal)
HRESULT put_InputPower(double newVal)
```

**Interface**
IMixer
IConverter
### InputRangeMode Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the Input sweep mode.</th>
</tr>
</thead>
</table>

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**Note:** There is also a InputRangeMode Property on the Converter Object.

**VB Syntax**  
`obj.InputRangeMode = value`

**Variable (Type) - Description**

- `obj` A Mixer Interface pointer to the Measurement (object)
- `value` (Enum as MixerRangeMode) - Input sweep mode. Choose from:
  - `0 - mixSwept`
  - `1 - mixFixed`

**Return Type**  
Enum

**Default**  
0 - mixSwept

**Examples**  
`mixer.InputRangeMode = mixSwept`

**C++ Syntax**

- `HRESULT get_InputRangeMode(long *pVal)`
- `HRESULT put_InputRangeMode(long newVal)`

**Interface**  
IMixer6
### InputRangeMode Property

**Description**  
Sets or returns the Input sweep mode.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

**Note:** There is also a InputRangeMode on the Mixer Interface.

**VB Syntax**  
`obj.InputRangeMode = value`

**Variable**  
**(Type) - Description**

- `obj` : A Converter Object
- `value` : (Enum as NARangeMode) - Input sweep mode. Choose from:
  - `0 - naSwept`
  - `1 - naFixed`

**Return Type**  
Enum

**Default**  
0 - naSwept

**Examples**  
`conv.InputRangeMode = naSwept`

**C++ Syntax**  

```
HRESULT get_InputRangeMode(long LO, enum *pVal)

HRESULT put_InputRangeMode(long LO, enum newVal)
```

**Interface**  
IConverter
## InputStartFrequency Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the start frequency value of the mixer Input frequency.</th>
</tr>
</thead>
<tbody>
<tr>
<td>If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.</td>
<td></td>
</tr>
</tbody>
</table>

### VB Syntax

```
obj.InputStartFrequency = value
```

### Variable 

- **obj** (Type) - Description
  - A Mixer Interface pointer to the Measurement (object)
  - Or
    - A Converter Object

- **value** (double) - Input start frequency in Hertz.

### Return Type

Double

### Default

Start frequency of the VNA

### Examples

```
mixer.InputStartFrequency = Start_Freq
```

### C++ Syntax

```
HRESULT get_InputStartFrequency(double *pVal)
```

```
HRESULT put_InputStartFrequency(double newVal)
```

### Interface

- IMixer
- IConverter
**InputStartPower Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the Start Power value of the mixer Input Power.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.</td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
obj.InputStartPower = value
```

**Variable** *(Type)* - Description

- **obj** - A Converter Object
- **value** - (double) - Input start Power in dBm.

**Return Type** Double

**Default** Start Power of the VNA

**Examples**

```vbnet
mixer.InputStartPower = 0
```

**C++ Syntax**

```cpp
HRESULT get_InputStartPower(double *pVal)

HRESULT put_InputStartPower(double newVal)
```

**Interface** IConverter
### InputStopFrequency Property

**Description**
Sets and returns the stop frequency value of the mixer Input frequency.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**

```
obj.InputStopFrequency = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>A Mixer Interface pointer to the Measurement (object)</td>
</tr>
</tbody>
</table>

Or

A Converter Object

<table>
<thead>
<tr>
<th>value</th>
<th>(double) - Input stop frequency in Hertz.</th>
</tr>
</thead>
</table>

**Return Type**
Double

**Default**
Stop frequency of the VNA

**Examples**

```
mixer.InputStopFrequency = Stop_Freq
```

**C++ Syntax**

- HRESULT get_InputStopFrequency(double *pVal)
- HRESULT put_InputStopFrequency(double newVal)

**Interface**
IMixer
IConverter
### InputStopPower Property

**Description**
Sets and returns the Stop Power value of the mixer Input Power.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**

```vbnet
obj.InputStopPower = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>obj</code></td>
<td>A Converter Object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) - Input Stop Power in dBm.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
Stop Power of the VNA

**Examples**

```vbnet
mixer.InputStopPower = 0
```

**C++ Syntax**

```cpp
HRESULT get_InputStopPower(double *pVal)

HRESULT put_InputStopPower(double newVal)
```

**Interface**
IConverter
### InternalDCReceiverCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of internal DC receivers in the analyzer.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = cap.InternalDCReceiverCount</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) - Variable to store the returned value.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = cap.InternalDCReceiverCount</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_InternalDCReceiverCount(long *value);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities14</td>
</tr>
</tbody>
</table>
### InternalDCReceiverNames Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a list of names of the internal DC receivers.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = cap.InternalDCReceiverNames</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> Description</td>
</tr>
<tr>
<td></td>
<td><code>value</code> Variant containing one-dimensional array of string names.</td>
</tr>
<tr>
<td></td>
<td><code>cap</code> A <code>Capabilities (object)</code></td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = cap.InternalDCReceiverNames 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_InternalDCReceiverNames(Variant *names);</td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities14</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# InternalDCSourceCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of internal DC sources in the analyzer.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.InternalDCSourceCount</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) - Variable to store the returned value.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.InternalDCSourceCount 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_InternalDCSourceCount(long *value);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities14</td>
</tr>
<tr>
<td>Description</td>
<td>Returns a list of names of the internal DC sources.</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.InternalDCSourceNames</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td>Value Variant containing one-dimensional array of string names.</td>
</tr>
<tr>
<td><strong>cap</strong></td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.InternalDCSourceNames </code>Read`</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_InternalDCSourceNames(Variant *names);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities14</td>
</tr>
</tbody>
</table>
**InternalSourcePortCount Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of internal source ports.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.InternalSourcePortCount</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) - Variable to store the returned value.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.InternalSourcePortCount</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_InternalSourcePortCount(Long *value);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities13</td>
</tr>
</tbody>
</table>
InternalSourcePortNames Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a list of internal source port names.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>value = cap. InternalSourcePortNames</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>Variant containing one-dimensional array of string names.</td>
</tr>
<tr>
<td>cap</td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>value = cap. InternalSourcePortNames 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_InternalSourcePortNames(Variant *names);</td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities13</td>
</tr>
</tbody>
</table>
**InternalTestsetPortCount Property**

Description: Returns the number of VNA test ports. This does not include the ports on an external test set.

**VB Syntax**

```
value = cap.InternalTestsetPortCount
```

**Variable**

- **value** (Long) - Variable to store the returned number of VNA test ports.
- **cap** A Capabilities (object)

**Return Type** Long

**Default** Not Applicable

**Examples**

```
value = cap.InternalTestsetPortCount 'Read
```

**C++ Syntax**

```
HRESULT get_InternalTestsetPortCount(long *numPorts);
```

**Interface** ICapabilities
## InternalTestsetPortNames Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a list of internal test port names.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.InternalTestsetPortNames</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>Variant containing one-dimensional array of string names.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.InternalTestsetPortNames</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_InternalTestsetPortNames(Variant *names);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities13</td>
</tr>
</tbody>
</table>
Interpolate Correction Property

Description
Turns ON and OFF correction interpolation which calculates new error terms when stimulus values change after calibration.

When this property is ON and error correction is being applied, the calibration subsystem attempts to interpolate the error terms whenever the stimulus parameters are changed.

When this property is OFF under the same circumstances, error correction is turned OFF.

VB Syntax
```
meas.InterpolateCorrection = value
```

Variable (Type) - Description

- **meas**: A Measurement (object)
- **value**: (boolean) - Choose from:
  - **True**:Turns correction interpolation ON
  - **False**:Turns correction interpolation OFF

Return Type
Boolean

Default
True

Examples
```
meas.InterpolateCorrection = False
```

C++ Syntax
```
HRESULT get_InterpolateCorrection(boolean *pVal)
HRESULT put_InterpolateCorrection(boolean newVal)
```

Interface
IMeasurement
Write/Read

Interpolated Property

Description: Turns marker Interpolation ON and OFF. Marker interpolation enables X-axis resolution beyond the discrete data values. The analyzer will calculate the x and y-axis data values between discrete data points. Use meas.Interpolate to change interpolation of all markers in a measurement. This command will override the measurement setting.

**VB Syntax**

```vbnet
mark.Interpolated = value
```

**Variable (Type) - Description**

- **mark**: A Marker (object)
- **value**: (boolean)
  - **False**: Turns interpolation OFF
  - **True**: Turns interpolation ON

**Return Type**: Boolean

**Default**: True

**Examples**

```vbnet
mark.Interpolated = True 'Write
interpolate = mark.Interpolated 'Read
```

**C++ Syntax**

```cpp
HRESULT get_Interpolated(VARIANT_BOOL *pVal)
HRESULT put_Interpolated(VARIANT_BOOL newVal)
```

**Interface**: IMarker
InterpolateMemory Property

Description
Sets and reads the state of the memory data interpolation. Learn more.

VB Syntax
meas.InterpolateMemory = bool

Variable (Type) - Description
meas (measurement (object))
bool (boolean)
False - Turn memory data interpolation OFF.
True - Turn memory data interpolation ON.

Return Type
Boolean

Default
False

Examples
meas.InterpolateMemory = True 'Write
state = meas.InterpolateMemory 'Read

C++ Syntax
HRESULT get_InterpolateMemory(VARIANT_BOOL *intState)
HRESULT put_InterpolateMemory(VARIANT_BOOL intState)

Interface
IMeasurement18
## InterpolateMemoryIsDefault Property

**Description**  
Sets and reads the state of the memory data interpolation default preference. The PNA will return to the default interpolation state after a Preset, creating a new trace, or closing the PNA application. Learn more.

**VB Syntax**  
```vbnet
pref.InterpolateMemory = bool
```

**Variable**  
(`Type`) - Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pref</code></td>
<td>A Preferences (object)</td>
</tr>
<tr>
<td><code>bool</code></td>
<td>(boolean)</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Set memory interpolation to OFF as the default.</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Set memory interpolation to ON as the default.</td>
</tr>
</tbody>
</table>

**Return Type**  
Boolean

**Default**  
False

**Examples**  
```vbnet
pref.InterpolateMemoryIsDefault = True 'Write
prefer = pref.InterpolateMemory 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_InterpolateMemory(VARIANT_BOOL *intState)
HRESULT put_InterpolateMemory(VARIANT_BOOL intState)
```

**Interface**  
IPreferences
**InterpolateNormalization Property  Superseded**

**Description**  
*Note: This property is replaced by DoReceiverPowerCal Method.*

Turns ON and OFF normalization interpolation which calculates new divisor data when stimulus values change after normalization.

When this property is ON and normalization is being applied, the Normalization algorithm attempts to interpolate the divisor data whenever the stimulus parameters are changed.

When this property is OFF under the same circumstances, normalization is turned OFF.

Normalization is currently supported only on measurements of unratioed power for the purpose of performing a receiver power calibration.

**VB Syntax**  
`meas.InterpolateNormalization = value`

**Variable (Type) - Description**
- `meas` (object) - A Measurement object
- `value` (boolean)
  - False – Turns normalization interpolation OFF
  - True – Turns normalization interpolation ON

**Return Type**  
Boolean

**Default**  
False -OFF

**Examples**
- `meas.InterpolateNormalization = False 'Write`
- `normalized = meas.InterpolateNormalization 'Read`

**C++ Syntax**

- `HRESULT put_InterpolateNormalization(VARIANT_BOOL bState);`
- `HRESULT get_InterpolateNormalization(VARIANT_BOOL *bState);`

**Interface**  
IMeasurement
### Interrupt Property

**Description**
Reads the boolean that represents the state of the Interrupt In line (pin 13) on the external test set connector.

**VB Syntax**
```
value = ExtIO.Interrupt
```

**Variable**
- **value** (boolean) - Variable to store the returned data
- **ExtIO** (object) - An ExternalTestSetIO object

**Return Type**
Boolean

- **False** - indicates the line is being held at a TTL High
- **True** - indicates the line is being held at a TTL Low

**Default**
Not Applicable

**Examples**
```
value = ExtIO.Interrupt
```

**C++ Syntax**
```
HRESULT get_Interrupt( VARIANT_BOOL* bValue);
```

**Interface**
IHWExternalTestSetIO
### Invert Property

**Description**  
Sets whether to invert the polarity of the pulse.

**VB Syntax**  
`pulse.Invert (n) = value`

**Variable (Type) - Description**

- `pulse` A `PulseGenerator (object)`
- `n` (Integer) Pulse generator number. Choose from 0 to 4.
  
  Or use `PulseGeneratorID` to refer to an external pulse generator.

  0 is the generator that pulses the ADC.

- `value` (Boolean) - Invert the pulse generator polarity. This causes the pulse ON time to be active low and OFF be active high.
  
  **True** - Invert the pulse generator polarity.
  
  **False** - Do NOT Invert the pulse generator polarity.

**Return Type**  
Boolean

**Default**  
False

**Examples**

```markdown
pulse.Invert(1) = True 'Write
value = pulse.Invert(4) 'Read
```

**C++ Syntax**

```cpp
HRESULT get_Invert(integer pulse, VARIANT_BOOL *pVal);

HRESULT put_Invert(integer pulse, VARIANT_BOOL newVal);
```

**Interface**  
`IPulseGenerator4`
# IOConfiguration Property

**Description**
Sets and returns the method of communication and address for the external device.

**VB Syntax**
`extDevices.IOConfiguration = value`

**Variable**
- **(Type)**: `Description`  
  - `extDevices` (*An ExternalDevice* object)  
  - `value` (*String*) Configuration path. Any valid VISA resource shown in the IO Configuration field of the External Devices dialog, enclosed in quotes.

  Do NOT use the ID string of a PMAR USB power sensor as the resource string. The ID string is returned by the `USBPowerMeterCatalog Property`

**Return Type**
String

**Default**
" " Empty String

**Examples**
```vbnet
extDevices.IOConfiguration = "TCPIP::141.121.76.239::inst0::INSTR" 'Write
value = extDevices.IOConfiguration 'Read
```

See example program to configure PMAR device

See example program to configure External Source

**C++ Syntax**
- `HRESULT get_IOConfiguration( BSTR* value);`
- `HRESULT put_IOConfiguration( BSTR newVal);`

**Interface**
`IExternalDevices`
IOEnable Property

**Description**
Sets and returns whether an external device is enabled for IO communication with the VNA.

When disabled (False), the VNA will NOT attempt to connect to the external device regardless of the instrument state (Active). Therefore, no errors will be produced if the device is not connected.

This command is useful for debugging and testing states when the external device is not connected. This command is unnecessary in ordinary operation (when the device is connected).

**VB Syntax**
```
extDevices.IOEnable = value
```

**Variable (Type) - Description**
- **extDevices**: An ExternalDevice (object)
- **value**: (Boolean) Choose from:
  - **True**: Device is available for IO communication.
  - **False**: Device is NOT available for IO communication.

**Return Type**: Boolean
**Default**: True

**Examples**
```
extDevices.IOEnable = True 'Write
bool = extDevices.IOEnable 'Read
See example program to configure PMAR device
See example program to configure External Source
```

**C++ Syntax**
```
HRESULT get_IOEnable( VARIANT_BOOL* value);
HRESULT put_IOEnable( VARIANT_BOOL newVal);
```

**Interface**: IExternalDevices
### IsBalanced Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns whether or not the specified logical port number is balanced.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = chan.IsBalanced(logicalPort)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>balTopology</code></td>
<td>A BalancedTopology <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(boolean)</em> - Returned value is either:</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Logical port number is NOT balanced.</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Logical port number IS balanced.</td>
</tr>
<tr>
<td><code>logicalPort</code></td>
<td><em>(Long Integer)</em> Logical port number.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>*In the following example, 6 logical ports are mapped to 8 physical VNA ports.</td>
</tr>
<tr>
<td></td>
<td>*Logical port 3 is queried and &quot;True&quot; is returned.</td>
</tr>
<tr>
<td></td>
<td><code>Int portlist[] = {1,2,3,4,5,6,7,8}</code></td>
</tr>
<tr>
<td></td>
<td><code>balTopology.SetCustomDuTTopology (&quot;SSBSSB&quot;, portlist)</code></td>
</tr>
<tr>
<td></td>
<td><code>variable = balTopology.IsBalanced(3) 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_IsBalanced(long logicalPort, VARIANT_BOOL *bBalanced);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IBalancedTopology3</td>
</tr>
</tbody>
</table>
IsContinuous Property

Description
Returns whether or not a channel is in continuous mode. To set the channel to continuous mode, use Continuous Method.

VB Syntax

\[
value = chan.IsContinuous
\]

Variable

(Type) - Description

\[value\] (boolean) - Choose either:

- **False** - Channel trigger is NOT set to continuous.
- **True** - Channel trigger IS set to continuous.

\[chan\] Channel (object)

Return Type
Boolean

Default
Not Applicable

Examples

\[trig = chan.IsContinuous \]  

C++ Syntax

HR\RESULT get_IsContinuous ( VARIANT\_BOOL *bContinuous);

Interface
IChannel4
### IsDevicePresent Property

**Description**  
Returns whether the named device is present on the bus for which it is configured.

**VB Syntax**  
```vbnet
value = extDevices.IsDevicePresent(devName)
```

**Variable** *(Type) - Description*

- **value** *(Boolean)* Variable to store the returned value:
  
  - **False** - The device is not in the collection or the device fails to respond and times out when communication is attempted.
  
  - **True** - The device responds when communication is attempted.

- **extDevices** An **ExternalDevices (collection)**

- **devName** *(String)* Name of the device with which you want to communicate.

**Return Type**  
Boolean

**Default**  
Not Applicable

**Examples**  
```vbnet
present = extDevices.IsDevicePresent("MyPowerMeter") 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_IsDevicePresent(VARIANT_BOOL *present);
```

**Interface**  
IExternalDevices2
IsECALModuleFoundEx Property

Description
This property replaces IsECALModuleFound Property.

Returns true or false depending on whether communication was established between the VNA and the specified ECAL module.

VB Syntax

```vbnet
modFound = cal.IsECALModuleFoundEx(module)
```

Variable (Type) - Description

```vbnet
modFound (boolean) - Variable to store the returned test result.
```

**True** - The VNA identified the presence of the specified ECAL module.

**False** - The VNA did NOT identify the presence of the specified ECAL module.

```vbnet
cal (object) - A Calibrator object

module (long integer) - ECAL module.
```

Choose from modules 1 through 8.

Use GetECALModuleInfoEx to return the model and serial number of each module.

Return Type
Boolean

Default
Not applicable

Examples

```vbnet
Set cal = pna.ActiveChannel.Calibrator
moduleFound = cal.IsECALModuleFoundEx(1)
```

C++ Syntax

```cpp
HRESULT get_IsECALModuleFoundEx( long moduleNumber,
                                VARIANT_BOOL *bModuleFound);
```

Interface
ICalibrator4
### IsEnhancedModulationControl Property

**Description**
Sets and reads the state of modulation sources having the capability to download modulation files. Modulation distortion control must be ON to control the modulation of an external source.

**VB Syntax**
```
extSource.IsEnhancedModulationControl = value
```

**Variable**

- **extSource** (Type) - Description: An ExternalSource Object (object)
- **value** (Boolean) Choose from:
  - 1 - Enable control of external modulation.
  - 0 - Disable control of external modulation.

Learn about these settings and about adding an external source.

**Return Type**
Boolean

**Default**
0

**Examples**
```vbnet
eextSource.IsEnhancedModulationControl = 1 'Write
value = extSource.IsEnhancedModulationControl 'Read
```

**C++ Syntax**
```
HRESULT put_IsEnhancedModulationControl(VARIANT_BOOL bEnable);
HRESULT get_IsEnhancedModulationControl(VARIANT_BOOL* bEnable);
```

**Interface**
IEternalSource3
### IsFrequencyOffsetPresent Property

**Description**
Returns a value indicating the presence of Frequency Offset Option S93080A in the remote VNA.

**VB Syntax**
```
value = cap.IsFrequencyOffsetPresent
```

**Variable**
- **value** (Boolean) - Variable to store the returned value
  - True - Frequency Offset Option S93080A is present
  - False - Frequency Offset Option S93080A is not present

**cap**
A `Capabilities` (object)

**Return Type**
Boolean

**Default**
Not Applicable

**Examples**
```
value = cap.isFrequencyOffsetPresent(1) 'Read
```

**C++ Syntax**
```
HRESULT get_IsFrequencyOffsetPresent(VARIANT_BOOL * present);
```

**Interface**
ICapabilities
# IsHold Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns whether or not a channel is in hold mode. To set the channel to hold mode, use Hold Method.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = chan.IsHold</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(boolean)</strong> - Choose either:</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Channel trigger is NOT set to hold.</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Channel trigger IS set to hold.</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>Channel <strong>(object)</strong></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>trig = chan.IsHold 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_IsHold ( VARIANT_BOOL *bHold);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel4</td>
</tr>
</tbody>
</table>
## IsMarkerOn Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns whether or not a marker was used for the specified tuning sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = embedLODiag.IsMarkerOn(n)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td>value</td>
<td><em>(Boolean)</em> Variable to store the returned data.</td>
</tr>
<tr>
<td>embedLODiag</td>
<td>An EmbeddedLODiagnostic <em>(object)</em></td>
</tr>
<tr>
<td>n</td>
<td>Tuning sweep number. Use NumberOfSweeps to find the number of sweeps taken.</td>
</tr>
<tr>
<td>Return Type</td>
<td><em>(String)</em></td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>data = embedLO.IsMarkerOn 3 'read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT IsMarkerOn(long sweep, VARIANT_BOOL* markerOn);</td>
</tr>
<tr>
<td>Interface</td>
<td>IEmbededLODiagnostic</td>
</tr>
</tbody>
</table>
## IsolationIncrementAveraging Property

### Description
Value by which to increment (increase) the channel's averaging factor during measurement of isolation in an ECal calibration.

**Note:** If `<value>` is greater than 1 and the channel currently has averaging turned OFF, averaging will be turned ON only during the isolation measurements and with the averaging factor equal to `<value>`.

### VB Syntax
```
cal.IsolationIncrementAveraging = value
```

### Variable *(Type)* - Description
- **cal**
  - A **Calibrator** *(object)*
- **value**
  - A **Long** Incremental Averaging factor. The maximum averaging factor is 65536 ($2^{16}$).

### Return Type
Long Integer

### Default
8

### Examples
```vba
oCal.IsolationAveragingIncrement = 16 'Write
avgIncr = oCal.IsolationAveragingIncrement ' Read
```

### C++ Syntax
```c++
HRESULT get_IsolationAveragingIncrement(long *pVal);

HRESULT put_IsolationAveragingIncrement(long newVal);
```

### Interface
ICalibrator7
**IsOn Property**

| Description | Sets and returns the ON |OFF state of Embedded LO. |
|-------------|------------------------|
| VB Syntax   | *obj*.IsOn = *value*   |
| Variable    | (Type) - Description   |
| *obj*       | An EmbeddedLO *(object)* or |
|             | A ConverterEmbeddedLO *(object)* |
| *value*     | (Boolean)               |
|             | False - Turns Embedded LO OFF |
|             | True - Turns Embedded LO ON |
| Return Type | (Boolean)               |
| Default     | False (OFF)             |
| Examples    |                         |
|             | embedLO.IsOn = True 'write |
|             | data= embedLO.IsOn 'read |
| C++ Syntax  | HRESULT get_IsOn( VARIANT_BOOL* IsOn); |
|             | HRESULT put_IsOn( VARIANT_BOOL IsOn); |
| Interface   | IEmbeddedLO             |
IsReceiverStepAttenuatorPresent Property

Description
Returns a value indicating the presence of Receiver step attenuators in the remote VNA.

VB Syntax
value = cap.IsReceiverStepAttenuatorPresent(n)

Variable
- **value** (Boolean) - Variable to store the returned value.
  - True - Receiver step attenuators are present.
  - False - Receiver step attenuators are not present.

- **cap** - A Capabilities (object)

- **n** (Long) - port number to query for step attenuators

Return Type
Boolean

Default
Not Applicable

Examples
value = cap.IsReceiverStepAttenuatorPresent(1) 'Read

C++ Syntax
HRESULT get_IsReceiverStepAttenuatorPresent(long portNumber, 
VARIANT_BOOL * present);

Interface
ICapabilities
**IsReferenceBypassSwitchPresent Property**

**Description**
Returns a value indicating the presence of a Reference Bypass Switch in the remote VNA.

**VB Syntax**

```vbnet
value = cap.IsReferenceBypassSwitchPresent(n)
```

**Variable**

- `value` (Boolean) - Variable to store the returned value.
  - True - Reference Bypass Switch is present.
  - False - Reference Bypass Switch is not present.

- `cap` A `Capabilities` (object)

- `n` (Long) - port number to query for reference bypass switch

**Return Type**
Boolean

**Default**
Not Applicable

**Examples**

```vbnet
value = cap.IsReferenceBypassSwitchPresent(1) 'Read
```

**C++ Syntax**

```csharp
HRESULT get_IsReferenceBypassSwitchPresent(long portNumber, VARIANT_BOOL * present);
```

**Interface**
ICapabilities
IsSingleEnded Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns whether or not the specified logical port number is single ended.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = chan.IsSingleEnded(logicalPort)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>balTop</code></td>
<td>A <code>BalancedTopology (object)</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(boolean)</em> - Returned value is either:</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Logical port number is NOT single ended.</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Logical port number IS single ended.</td>
</tr>
<tr>
<td><code>logicalPort</code></td>
<td><em>(Long Integer)</em> Logical port number.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>'In the following example, 6 logical ports are mapped to 8 physical VNA ports.</td>
</tr>
<tr>
<td></td>
<td>'Logical port 4 is queried and &quot;True&quot; is returned.</td>
</tr>
<tr>
<td></td>
<td>Int portlist[] = {1,2,3,4,5,6,7,8}</td>
</tr>
<tr>
<td></td>
<td><code>balTop</code>.SetCustomDuTTopology (&quot;SSBSSB&quot;, portlist)</td>
</tr>
<tr>
<td></td>
<td><code>variable = balTop.IsSingleEnded(4)</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_IsSingleEnded(long logicalPort, VARIANT_BOOL *bSingleEnded);</td>
</tr>
<tr>
<td>Interface</td>
<td>IBalancedTopology3</td>
</tr>
</tbody>
</table>
**IsSPParameter Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns true if measurement represents an S-Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = meas.IsSPParameter</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean)</td>
</tr>
<tr>
<td></td>
<td>True - measurement is an S-Parameter</td>
</tr>
<tr>
<td></td>
<td>False - measurement is NOT an S-Parameter</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>True</td>
</tr>
<tr>
<td>Examples</td>
<td><code>print app.IsSPParameter</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT IsSPParameter( [out, retval] VARIANT_BOOL * bVal);</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
### Item Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Add or change a name-value pair in the Cal Set, or read the value associated with a name.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>After editing, <strong>Save</strong> the CalSet to the VNA.</td>
</tr>
<tr>
<td><strong>About Name-Value pairs</strong></td>
<td>A Cal Set name-value pair is a general purpose data structure that maps a name to a value. This property allows you to associate a name with a value. Then, using this same property, you can read the value using the name.</td>
</tr>
<tr>
<td></td>
<td>For example, one of the items added by the VNA firmware to every Cal Set is named 'Created By'. The value attached to this item is the name of the VNA App that created the Cal Set. When an SMC cal is performed, you can query the Cal Set for the 'Create By' item, and it will return 'Scalar Mixer/Converter'. The same query on an NFx channel returns 'Noise Figure Converters'.</td>
</tr>
<tr>
<td><strong>Warning</strong></td>
<td>Do NOT change the name or value of any Items that you did NOT create. Otherwise, the VNA firmware may behave unpredictably.</td>
</tr>
<tr>
<td><strong>See Also</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>EnumerateItems Method</strong></td>
</tr>
<tr>
<td></td>
<td><strong>RemoveItem Method</strong></td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td><em>CalSet.Item</em>(name) = value</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td></td>
<td><em>CalSet</em> <em>(object)</em> - A Cal Set object</td>
</tr>
<tr>
<td></td>
<td><em>name</em> <em>(String)</em> - Name of the name-value pair.</td>
</tr>
<tr>
<td></td>
<td><em>value</em> <em>(Variant)</em> - Can be an integer, float, double, string, or a single-dimensioned array of integer, float, double, string.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>'Create the VNA object'</td>
</tr>
<tr>
<td></td>
<td>Dim pna</td>
</tr>
<tr>
<td></td>
<td>Set pna = CreateObject(&quot;AgilentPNA835x.Application&quot;)</td>
</tr>
</tbody>
</table>
' Get a handle to the calsets collection
Dim calsets
Set calsets = pna.GetCalManager.calsets

' Get a handle to the cal set to be edited
Dim MyCalSet
Set MyCalSet = calsets.Item("CalSet_1")

' Add a name-value pair(item) to MyCalSet
MyCalSet.Item("MyItem")=15

' Save the edited Cal Set to the VNA
MyCalSet.Save

' Loop thru the name-value pairs in the Cal Set
Dim CSetItems
CSetItems = MyCalSet.EnumerateItems
for i=lbound(CSetItems) to Ubound(CSetItems)
    ' List the item names in MyCalSet
    Dim name
    name = CSetItems(i)
    wscript.echo name

    ' List the value for each item name
    Dim value
    value = MyCalSet.Item(name)
    wscript.echo value
Next

' Delete the new name-value pair
MyCalSet.RemoveItem("MyItem")

**C++ Syntax**

HRESULT get_Item( BSTR name, VARIANT *value);

HRESULT put_Item( BSTR name, VARIANT value);

**Interface**

ICalSet6
## Items Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a list of configured external devices in the collection.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>array = extDevices.Items</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(array) Variable to store the returned values. Each item in the array is in the format (name:driver), where:</td>
</tr>
<tr>
<td></td>
<td>- name = name of the external device</td>
</tr>
<tr>
<td></td>
<td>- driver = <a href="#">See a list of supported drivers.</a></td>
</tr>
<tr>
<td>extDevices</td>
<td>An ExternalDevices (collection)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Variant Array</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>array = extDevices.Items</code> Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_Items(VARIANT* array);</td>
</tr>
<tr>
<td>Interface</td>
<td>IExternalDevices</td>
</tr>
</tbody>
</table>
### IterationNumber Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the maximum leveling sweep iterations to be used in order to achieve the tolerance setting.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>RxLevel.IterationNumber(srcPort) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>RxLevel</code></td>
<td>A ReceiverLeveling Object</td>
</tr>
<tr>
<td><code>srcPort</code></td>
<td>(Long Integer) Source port for which to set the IterationNumber for Receiver Leveling.</td>
</tr>
</tbody>
</table>

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port.

| **value** | (Long Integer) Max iterations. Choose a value between 1 and 25. |
| **Return Type** | (Long Integer) |
| **Default** | 5 |

| **Examples** | `rxLevel.IterationNumber (1) = 10` 'Write' |
| *value = rxLevel.IterationNumber 2' 'Read* |

| **C++ Syntax** | `HRESULT get_IterationNumber(long port, long* pVal);` |
| | `HRESULT put_IterationNumber(long port, long newVal);` |

| **Interface** | IReceiverLevelingConfiguration |
IterationsTolerance Property

**Description**
This command, along with MaximumIterationsPerPoint Property deal with adjustments made to the source power.

Sets the maximum desired deviation from the sum of the test port power and the offset value. Power readings will continue to be made, and source power adjusted, until a reading is within this tolerance value or the max number of readings has been met. The last value to be read is the valid reading for that data point.

The following two commands allow for settling of power readings.

**ReadingsPerPoint Property**

**ReadingsTolerance Property**

**VB Syntax**

```
pwrCal.IterationsTolerance = value
```

**Variable**

(Variable) - Description

- `pwrCal` (object) - A SourcePowerCalibrator (object)
- `value` (Double) – Tolerance value in dBm. Choose any number between 0 and 5

**Return Type**
Double

**Default**
.05 dB

**Examples**

```
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.IterationsTolerance = .1 'Write
ReadTol = powerCalibrator.IterationsTolerance 'Read
```

**C++ Syntax**

```
HRESULT get_IterationsTolerance( double *pVal);

HRESULT put_IterationsTolerance( double newVal);
```

**Interface**
ISourcePowerCalibrator3
## KaiserBeta Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the Kaiser Beta of Time Domain transform windows</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>trans.KaiserBeta = value</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>trans</code></td>
<td>A Transform (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(single) - Kaiser Beta. Choose any number between 0 and 13.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Single</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>trans.KaiserBeta = 6 'sets the Kaiser Beta of a transform window</code></td>
</tr>
<tr>
<td></td>
<td><code>KB = trans.KaiserBeta 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_KaiserBeta(float *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_KaiserBeta(float newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ITransform</td>
</tr>
</tbody>
</table>
### L0 Property

**Description**
Sets and Returns the L0 (L-zero) value (the first inductance value) for the calibration standard.

To set the other inductance values, use L1, L2, L3.

**VB Syntax**
```vbnet
calstd.L0 = value
```

**Variable (Type) - Description**
- `value` (single): Value for L0 in femtohenries (1E-15)

**Return Type**
Single

**Default**
Not Applicable

**Examples**
```vbnet
calstd.L0 = 15 'Write the value of L0 = 15 femtohenries
Induct0 = calstd.L0 'Read the value of L0
```

**C++ Syntax**
```cpp
HRESULT get_L0(float *pVal)
HRESULT put_L0(float newVal)
```

**Interface**
ICalStandard
L1 Property

Description: Sets and Returns the L1 value (the second inductance value) for the calibration standard.

To set the other inductance values, use L0, L2, L3.

VB Syntax: `calsld.L1 = value`

Variable (Type) - Description:
- `value`: (single) - Value for L1.

Return Type: Single

Default: Not Applicable

Examples:
- `calsld.L1 = 15 'Write the value of L1`
- `Induct1 = calsld.L1 'Read the value of L1`

C++ Syntax:
- `HRESULT get_L1(float *pVal)`
- `HRESULT put_L1(float newVal)`

Interface: ICalStandard
L2 Property

Description  Sets and Returns the L2 value (the third inductance value) for the calibration standard.
To set the other inductance values, use L0, L1, L3.

VB Syntax  
\[ \text{calstd}.L2 = \text{value} \]

Variable  
- \(\text{calstd}\) (Type) - Description
- \(\text{value}\) (single) - Value for L2.

Return Type  Single

Default  Not Applicable

Examples  
\[ \text{calstd}.L2 = 15 \ 'Write the value of L2. \]
\[ \text{Induct2} = \text{calstd}.L2 \ 'Read the value of L2 \]

C++ Syntax  
- HRESULT get_L2(float *pVal)
- HRESULT put_L2(float newVal)

Interface  ICalStandard
## L3 Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Returns the L3 value (the third inductance value) for the calibration standard. To set the other inductance values, use L0, L1, L2.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>calstd.L3 = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>calstd</strong> (Type) - Description</td>
</tr>
<tr>
<td></td>
<td><code>value</code> - Value for L3.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Single</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>calstd.L3 = 15  'Write the value of L3.'</code></td>
</tr>
<tr>
<td></td>
<td><code>Induct3 = calstd.L3  'Read the value of L3'</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_L3(float *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_L3(float newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>ICalStandard</td>
</tr>
</tbody>
</table>
# Label Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Returns the label for the calibration standard. The label is used to prompt the user to connect the specified standard.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>calsdt.Label = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>calsdt</code></td>
<td>A CalStandard <em>(object)</em>. Use <code>calKit.GetCalStandard</code> to get a handle to the standard.</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(string)</em> - between 1 and 12 characters long. Cannot begin with a numeric.</td>
</tr>
<tr>
<td><strong>Return</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>calsdt.Label = &quot;Short&quot;</code> <em>(Write)</em></td>
</tr>
<tr>
<td></td>
<td><code>stdLabel = calsdt.Label</code> <em>(Read)</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Label(BSTR *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Label(BSTR newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalStandard</td>
</tr>
</tbody>
</table>
# Label Property

**Description**
Sets or gets the display label for a given channel/testset combination. The label appears in a status bar at the bottom of the VNA display when the `ShowProperties` property is set to TRUE.

**VB Syntax**
```vbnet
tset.Label(chNum) = value
```

**Variable**

- **tset**  
  A `TestsetControl` object. Obtained from the `ExternalTestsets` collection.

- **chNum**  
  (Integer) Channel number of the measurement.

- **value**  
  (String) The text of the label.

**Return Type**
String

**Default**
None

**Examples**

The following sets the label for channel 5 corresponding to a given testset object.

```
testset1.label(5) = 'High-power output'
```

**C++ Syntax**

```c++
HRESULT get_Label(long channelNum, BSTR *pLabel);
HRESULT put_Label(long channelNum, BSTR label);
```

**Interface**
`ITestsetControl`
LANConfiguration Property

Description
Returns information about the current status of the VNA’s computer networking configuration. This is the same set of information that is returned in an NA_IPConfiguration data structure by the GetIPConfigurationStruct method.

VB Syntax
value = app.LANConfiguration

Variable
(Type) - Description
value (String) Variable to contain the VNA’s LAN configuration string
app An Application (object)

Return Type
Comma-delimited string

Default
Not Applicable

Examples
networkConfigInfo = app.LANConfiguration

C++ Syntax
HRESULT get_LANConfiguration (BSTR * pStrConfig);

Interface
IApplication13
About Source Power Cal

LastCalPassedTolerance Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the pass / fail status of the tolerance limits of the target power from the most recent source power cal.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = pwrCal.LastCalPassedTolerance</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>pwrCal</strong> <em>(object)</em> – A <em>SourcePowerCalibrator</em> (object)</td>
</tr>
<tr>
<td>value</td>
<td><em>(boolean)</em> -</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> – Source power cal did NOT achieve the specified tolerance limits.</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> – Source power cal DID achieve the specified tolerance limits.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>status = powerCal.LastCalPassedTolerance</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_LastCalPassedTolerance(VARIANT_BOOL *bState);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ISourcePowerCalibrator7</td>
</tr>
</tbody>
</table>
### LastLevelingAsSPC Property

**Description**
Sets and returns the state of **Use Last Result for Source Power Cal**. When Leveling Mode is switched back to Internal, this feature turns Source Power Cal correction ON using the latest receiver leveling correction data.

**VB Syntax**
```vba
RxLevel.LastLevelingAsSPC(srcPort) = value
```

**Variable**

<table>
<thead>
<tr>
<th><strong>Type</strong></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RxLevel</strong></td>
<td>A ReceiverLeveling Object</td>
</tr>
<tr>
<td><strong>srcPort</strong></td>
<td>(Long Integer) Source port being used for Receiver Leveling.</td>
</tr>
</tbody>
</table>

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

**value**

- **True** - When Leveling Mode is switched back to Internal, Source Power Cal correction is turned ON using the latest receiver leveling correction data.
- **False** - When Leveling Mode is switched back to Internal, Source Power Cal correction is NOT turned ON.

**Return Type**
Variant Boolean

**Default**
False

**Examples**
```vba
rxLevel.LastLevelingAsSPC (1) = True ' Write
value = rxLevel.LastLevelingAsSPC 2 ' Read
```

**C++ Syntax**
```cpp
HRESULT get_LastLevelingAsSPC(long port, VARIANT_BOOL* pVal);

HRESULT put_LastLevelingAsSPC(long port, VARIANT_BOOL newVal);
```

**Interface**
IReceiverLevelingConfiguration2
### LastModified Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the time stamp of the last modification to this Cal Set. The time is returned in the local time zone setting of the VNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>value = Object.LastModified</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>object</td>
<td>Channel (object)</td>
</tr>
<tr>
<td>or</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CalSet (object) - Read-only property</td>
</tr>
<tr>
<td>value</td>
<td>(Variant) – Variable to store the time stamp.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>date = CalSet.LastModified 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_LastModified(VARIANT* datetime)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalSet3</td>
</tr>
</tbody>
</table>
### LegacyGroupDelayApertureMath Property

Sets the group delay aperture to use the legacy computation method.

**VB Syntax**

```vbnet
pref.LegacyGroupDelayApertureMath = value
```

**Variable** *(Type) - Description*

- **pref** - A Preferences *(object)*
- **value** - *(Boolean)* - Choose from:

  - True - Use legacy computation method.
  - False - Do not use legacy computation method.

**Return Type**

Boolean

**Default**

False

**Examples**

```vbnet
pref.LegacyGroupDelayApertureMath = True 'Write

gda = pref.LegacyGroupDelayApertureMath 'Read
```

**C++ Syntax**

```cpp
HRESULT get_LegacyGroupDelayApertureMath(VARIANT_BOOL* pVal);
HRESULT put_LegacyGroupDelayApertureMath(VARIANT_BOOL pVal);
```

**Interface**

IPreferences19
Write/Read  

About Receiver Leveling

**LevelingIFBW Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the IFBW to be used for leveling sweeps. Enable separate IFBW for leveling sweeps using <em>FastMode Property</em>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><em>RxLevel.LevelingIFBW(srcPort) = value</em></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><strong>RxLevel</strong></td>
<td>A <em>ReceiverLeveling</em> Object</td>
</tr>
<tr>
<td><strong>srcPort</strong></td>
<td>(Long Integer) Source port for which to set the LevelingIFBW for Receiver Leveling.</td>
</tr>
</tbody>
</table>

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use *chan.getPortNumber* to translate the string into a port number. To learn more see *Remotely Specifying a Source Port*.

<table>
<thead>
<tr>
<th><strong>value</strong></th>
<th>(Double) IFBW for leveling sweeps in Hz. The list of valid IF Bandwidths is different depending on the VNA model. See <em>list</em>. If an invalid number is specified, the VNA will round up to the closest valid number.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Return Type</strong></td>
<td>(Double)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>100 kHz</td>
</tr>
</tbody>
</table>
| **Examples** | *rxLevel.LevelingIFBW (1) = 1e3*  'Write'  
 *value = rxLevel.LevelingIFBW 2' Read* |
| **C++ Syntax** | HRESULT get_LevelingIFBW(long port, double* pVal);  
 HRESULT put_LevelingIFBW(long port, double newVal); |
| **Interface** | IReceiverLevelingConfiguration |
# Limit Property

Sets and returns the power limit for the specified port.

**VB Syntax**

```vbnet
gpl.Limit (port) = value
```

**Variable**

- **gpl**: A `GlobalPowerLimit` object
- **port**: `Long` - Port number for which power limit value is to be set.
- **value**: `Double` - Power limit value. Choose a value between -27 dBm (approximately) and the max settable power for the VNA.

**Return Type**

Double

**Default**

100 dBm for all ports

**Examples**

```vbnet
jgl.Limit(1) = 0 'Write
Limit = jgl.Limit(2) 'Read
```

**C++ Syntax**

```cpp
HRESULT get_Limit(long port, double *pVal)
HRESULT put_Limit(long port, double newVal)
```

**Interface**

`IGlobalPowerLimit`
## LimitCmd Property

**Description**  
Sets and returns a user-defined command string that is used to set the DC limit of the external DC source. The actual limit value is set using either the `VoltageLimit` (voltage) or `CurrentLimit` (current). The limit command is sent to the external DC source at the beginning of a sweep for each channel. The firmware automatically selects the current or voltage limit value depending on the external DC source type.

**VB Syntax**  
```
extDC.LimitCmd = value
```

**Variable (Type) - Description**

- **extDC**  
  An `ExternalDCDevice` (object)
- **value**  
  (String) User-defined command name: Include `{%f}` in the command string which will be substituted by the actual limit value.

**Return Type**  
String

**Default**  
Not Applicable

**Examples**

```vbnet
extDC.LimitCmd = "myDCDevice","Limit Command {%f}" 'Write

value = extDC.LimitCmd 'Read
```

**C++ Syntax**

```
HRESULT get_LimitCmd (BSTR* cmd)
HRESULT put_LimitCmd (BSTR cmd)
```

**Interface**  
`IExternalDCDevice4`
### LimitFrequency Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enable or disable the use of the power meter min and max frequencies.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pwrSensor.LimitFrequency = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>pwrSensor</code></td>
<td>A <code>PowerSensorAsReceiver</code> (Object) or a <code>PowerSensor</code> (Object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean) - State of min and max frequency use. Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Min and max frequencies disabled.</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Min and max frequencies enabled.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pwrSensor.LimitFrequency = False</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>limit = pwrSensor.LimitFrequency</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT put_LimitFrequency(VARIANT_BOOL newVal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_LimitFrequency(VARIANT_BOOL* pVal);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPowerSensorAsReceiver</td>
</tr>
<tr>
<td></td>
<td>IPowerSensor</td>
</tr>
</tbody>
</table>
### EndStimulus Property

<table>
<thead>
<tr>
<th>Description</th>
<th>When constructing a limit line, specifies the stimulus value for the end of the segment.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>limtseg.EndStimulus = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>limtseg</code></td>
<td>A LimitSegment <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(double)</em> - End Stimulus X-axis value. No units</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
<tr>
<td>Examples</td>
<td><strong>Set</strong> <code>limtseg = meas.LimitTest(1)</code></td>
</tr>
<tr>
<td></td>
<td><code>limtseg.EndStimulus = 8e9</code> <em>Write</em></td>
</tr>
<tr>
<td></td>
<td><code>EndStim = limtseg.EndStimulus</code> <em>Read</em></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_EndStimulus(double *pVal)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_EndStimulus(double newVal)</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ILimitSegment</td>
</tr>
</tbody>
</table>
## EndResponse Property

<table>
<thead>
<tr>
<th>Description</th>
<th>When constructing a limit line, specifies the amplitude value at the end of the limit segment.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>limtseg.EndResponse = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>limts</em></td>
</tr>
<tr>
<td></td>
<td><em>value</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>Set <code>limtseg = meas.LimitTest(1)</code>&lt;br&gt;<code>limtseg.EndResponse = 10</code> <em>(Write)</em>&lt;br&gt;<code>EndResp = limtseg.EndResponse</code> <em>(Read)</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT <code>get_EndResponse(double *pVal)</code>&lt;br&gt;HRESULT <code>put_EndResponse(double newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ILimitSegment</td>
</tr>
</tbody>
</table>
## LimitMax Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the Max DC limit value for a DC source.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>dc.LimitMax(deviceName) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>dc</code></td>
<td>A DCStimulus (object)</td>
</tr>
<tr>
<td><code>deviceName</code></td>
<td>(String) Name of the &quot;DC source&quot;</td>
</tr>
</tbody>
</table>

Use Source Property to read a list of configured DC source names.

- **value** (Double) Max DC limit value. Choose a value within the range of the DC source.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Double</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>10</td>
</tr>
</tbody>
</table>

**Examples**

```vbnet
dc.LimitMax("myDCSource") = 10 'Write
value = dc.LimitMax("myDCSource") 'Read
```

**C++ Syntax**

```cpp
HRESULT get_LimitMax(BSTR deviceName, double* value);
HRESULT put_LimitMax(BSTR deviceName, double value);
```

**Interface**

IDCStimulus2
## LimitMin Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the Min DC limit value for a DC source.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>dc.LimitMin(deviceName) = value</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>dc</code></td>
<td>A DCStimulus (object)</td>
</tr>
<tr>
<td><code>deviceName</code></td>
<td>(String) Name of the &quot;DC source&quot;</td>
</tr>
</tbody>
</table>

Use **Source Property** to read a list of configured DC source names.

<table>
<thead>
<tr>
<th><strong>value (Double)</strong></th>
<th>Min DC limit value. Choose a value within the range of the DC source.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>-10</td>
</tr>
</tbody>
</table>

### Examples

```
dc.LimitMin("myDCSource") = -10    'Write
value = dc.LimitMin("myDCSource")  'Read
```

### C++ Syntax

```cpp
HRESULT get_LimitMin(BSTR deviceName, double* value);
HRESULT put_LimitMin(BSTR deviceName, double value);
```

### Interface

`IDCStimulus2`
## LimitTestFailed Property

**Description**: Returns the results of limit testing for the measurement.

**VB Syntax**: `testFailed = meas.LimitTestFailed`

**Variable**

- **(Type)** - Description
  - `testFailed` (boolean): Variable to store the returned value
    - **False**: Limit Test Passed
    - **True**: Limit Test Failed

- **meas**: A Measurement (object)

**Return Type**: Boolean

**Default**: False returned if there is no testing in progress.

**Examples**

```vbnet
Dim testRes As Boolean
testRes = meas.LimitTestFailed
MsgBox (testRes)
```

**C++ Syntax**: `HRESULT get_LimitTestFailed(VARIANT_BOOL* trueIfFailed)`

**Interface**: IMeasurement
## LimitTestXPosition Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the X-axis position of the Limit Line Pass/Fail indicator on the VNA screen. The lower-left corner of the Pass/Fail indicator is the point of reference for positioning.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>win.LimitTestXPosition = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>win</code></td>
<td>A NAWindow (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Double)</em> X-axis position. Choose a value between 0 (far left) and 10 (far right).</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>7</td>
</tr>
<tr>
<td>Examples</td>
<td><code>win.LimitTestXPosition = 3</code> <em>(Write)</em></td>
</tr>
<tr>
<td></td>
<td><code>value = app.ActiveNAWindow.LimitTestXPosition</code> <em>(Read)</em></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_LimitTestXPosition(double *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_LimitTestXPosition(double newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>INAWindow4</td>
</tr>
</tbody>
</table>
## LimitTestYPosition Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the Y-axis position of the Limit Line Pass/Fail indicator on the VNA screen. The lower-left corner of the Pass/Fail indicator is the point of reference for positioning.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>win.LimitTestYPosition = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>win</code></td>
<td>A NAWindow (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Y-axis position. Choose a value between 0 (bottom) and 10 (top).</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>win.LimitTestYPosition = 3 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = app.ActiveNAWindow.LimitTestYPosition 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_LimitTestYPosition(double *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_LimitTestYPosition(double newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>INAWindow4</td>
</tr>
</tbody>
</table>
## Type (limit) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the Limit Line type.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>limt(index).Type = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td></td>
<td><code>limt</code> A LimitSegment (object)</td>
</tr>
<tr>
<td></td>
<td><code>index</code> (variant) - Limit line number in the LimitTest collection</td>
</tr>
<tr>
<td></td>
<td><code>value</code> (enum NALimitSegmentType) - Limit Line type. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <code>naLimitSegmentType_OFF</code> - turns limit line OFF</td>
</tr>
<tr>
<td></td>
<td>1 - <code>naLimitSegmentType_Maximum</code> - limit line fails with a data point ABOVE the line</td>
</tr>
<tr>
<td></td>
<td>2 - <code>naLimitSegmentType_Minimum</code> - limit line fails with a data point BELOW the line</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long Integer</td>
</tr>
<tr>
<td>Default</td>
<td>0 - OFF</td>
</tr>
<tr>
<td>Examples</td>
<td><code>Set limts = meas.LimitTest</code></td>
</tr>
<tr>
<td></td>
<td><code>limts.Type = naLimitSegmentType_Maximum</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>limitType = limts.Type</code> 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT put_Type(tagNALimitSegmentType *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_Type(tagNALimitSegmentType newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>ILimitSegment</td>
</tr>
</tbody>
</table>
### LineDisplay Property

**Description**
Turns the display of limit lines ON or OFF. To turn limit TESTING On and OFF, use State Property.

**Note:** Trace data must be ON to view limit lines

**VB Syntax**
```
limits.LineDisplay = state
```

**Variable**
- **Type** - Description
- **limitst** A LimitTest *(object)*
- **state** *(boolean)*
  - **False** - Turns the display of limit lines OFF
  - **True** - Turns the display of limit lines ON

**Return Type**
Long Integer

**Default**
**True**

**Examples**
```
LimitTest.LineDisplay = true 'Write
lineDsp = LimitTest.LineDisplay 'Read
```

**C++ Syntax**
- HRESULT get_LineDisplay(VARIANT_BOOL *pVal)
- HRESULT put_LineDisplay(VARIANT_BOOL newVal)

**Interface**
ILimitTest
ListData Property

Description
Sets and returns the DC stimulus values per point from the specified DC source.

This setting overrides the Start and Stop DC settings for the channel. Only the values that are set with this command can be read by this command. The read command does NOT read the values that are set using the Start and Stop settings.

VB Syntax

\[ DCStim.ListData(name, port) = values \]

Variable

- **DCStim**: A DCStimulus (object)
- **name, port**: (String) Name of the DC source and source port.

Use Source Property to read a list of configured DC source names.

To set the DC source to be always ON, do NOT specify a port.

- **value**: (Variant) DC stimulus value array. The size of the array must equal the sweep point number.

Return Type
Variant array

Default
Not Applicable

Examples

'The following shows how to set a DC value for each data point

```vba
dim data(2)
data(0) = 1
data(1) = 2
data(2) = 1
DC.ListData("AO1") = data
```

'Read

```vba
varData = dc.ListData("AO1")
for i=0 to 2
msgbox (varData (i))
next
```
C++ Syntax

HRESULT get_ListData(BSTR name, VARIANT * pValue);
HRESULT put_ListData(BSTR name, VARIANT  newValue);

Interface

IExternalDCSource
Read/Write

**LoadCharFromFile Property**

**Description**
Sets and returns whether a Mixer characterization file is to be loaded. Specify and load the filename with `CharFileName Property`.

**VB Syntax**

```vbnet
VMC.LoadCharFromFile = bool
```

**Variable**

- **VMC** - VMCType (object)
- **bool** - (Boolean)

  - **True** - Load from file
  - **False** - Perform mixer characterization

**Return Type**

Boolean

**Default**

False

**Examples**

```vbnet
value = VMC.LoadCharFromFile
```

**C++ Syntax**

```c++
HRESULT put_LoadCharFromFile(VARIANT_BOOL bLoadCharFromFile);
HRESULT get_LoadCharFromFile(VARIANT_BOOL *bLoadCharFromFile);
```

**Interface**

VMCType
**LoadImpedance Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the load impedance of the pulse generator.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>extPulseGen.LoadImpedance = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>extPulseGen</code></td>
<td>An <code>ExternalPulseGenerator</code> <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Double)</em> Pulse generator load impedance.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>50</td>
</tr>
<tr>
<td>Examples</td>
<td><code>extPulseGen.LoadImpedance = 50</code> <em>(Write)</em></td>
</tr>
<tr>
<td></td>
<td><code>load = extPulseGen.LoadImpedance</code> <em>(Read)</em></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_LoadImpedance (double *pValue)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_LoadImpedance (double newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IExternalPulseGenerator</td>
</tr>
</tbody>
</table>
**LoadPort Property**

**Description**
Returns the load port number associated with an S-parameter reflection measurement. If the measurement is not a reflection S-parameter, the number returned by this property will have no meaning.

**VB Syntax**

```vbnet
loadPort = meas.LoadPort
```

**Variable**

- **loadPort**
  - *(Type)* - Description
  - *(long integer)* - The reflection measurement’s load port number.
- **meas**
  - A Measurement *(object)*

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**

```vbnet
Set meas = pna.ActiveMeasurement
loadPort = meas.LoadPort
```

**C++ Syntax**

```cpp
HRESULT get_LoadPort(long *pPortNumber);
```

**Interface**
IMeasurement
Write/Read

**LocalLockoutState Property**

**Description**
Prevents use of the mouse, keyboard, and front panel while your program is running. Use of these controls while this property is set TRUE causes an error message on the VNA display. To prevent these messages, see About Error Messages.

**VB Syntax**
```
app.LocalLockoutState = bool
```

**Variable**
(Type) - Description
- `app` - An Application (object)
- `bool` - Choose either:
  - **False** - User Interface is NOT locked out.
  - **True** - User Interface IS locked out.

**Return Type**
Boolean

**Default**
False

**Examples**
```
app.LocalLockoutState = True 'Write
block = app.LocalLockoutState 'Read
```

**C++ Syntax**
```
HRESULT get_LocalLockoutState(VARIANT_BOOL *State);
HRESULT put_LocalLockoutState(VARIANT_BOOL *State);
```

**Interface**
IApplication4
Locator Property

**Description**
Specifies the location, address, or ID string of the power meter / sensor that is used during a source power calibration.

Use **Path Property** to specify the type of interface.

**VB Syntax**
`pwrMtrInterface.Locator = value`

**Variable**
(Type) - Description

- `pwrMtrInterface` (object) - A PowerMeterInterface (object)
- `value` (string) - Location of the power meter / sensor, depending on the type of interface (**Path Property**)

- For **naGPIB**, address of the power meter. Choose any integer between 0 and 30.
- For **naUSB**, the ID string of the power sensor. Use **USBPowerMeterCatalog Property** to see a list of ID strings of connected power sensors.
- For **naLAN**, the hostname or IP address of the power meter.
- For **naANY**, any VISA resource string or a VISA alias.

**Return Type**
String

**Default**
Not applicable

**Examples**
- `pwrMeterInterface.Locator = "13"` 'GPIB address'
- `pwrMeterInterface.Locator = "Keysight Technologies,U2000A,MY12345678"` 'USB ID string'
- `pwrMeterInterface.Locator = "mymeter.Keysight.com"` 'LAN'
- `pwrMeterInterface.Locator = "TCPIP0::mymeter.Keysight.com::5025::SOCKET"` 'Any'

**C++ Syntax**
`HRESULT put_Locator( BSTR pValue );`
`HRESULT get_Locator(BSTR* pValue );`
## Lock Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables or disables the ability to change the power limit values through the user interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>gpl.Lock = state</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td></td>
</tr>
<tr>
<td><code>gpl</code></td>
<td>A <code>GlobalPowerLimit</code> (object)</td>
</tr>
<tr>
<td><code>state</code></td>
<td>(boolean)</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>- Disables the ability to change the power limit values from the user interface.</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>- Enables the ability to change the power limit values from the user interface.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td></td>
</tr>
<tr>
<td><code>gpl.Lock = True</code></td>
<td>'Write'</td>
</tr>
<tr>
<td><code>UILock = gpl.Lock</code></td>
<td>'Read'</td>
</tr>
</tbody>
</table>
| **C++ Syntax** | ```
HRESULT get_Lock(BOOL *pVal)
HRESULT put_Lock(BOOL newVal)
``` |
| **Interface** | `IGlobalPowerLimit` |
## LODeltaFound Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the LO frequency delta from the specified tuning sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>( \text{value} = \text{embedLODiag.LODeltaFound}(n) )</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(Double) Variable to store the returned data.</td>
</tr>
<tr>
<td>embedLODiag</td>
<td>An EmbeddedLODiagnostic (object)</td>
</tr>
<tr>
<td>( n )</td>
<td>(Long) Tuning sweep number. Use NumberOfSweeps to find the number of sweeps taken.</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>\text{data} = \text{embedLO.LODeltaFound} 3 \ 'read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT LODeltaFound(long sweep, double* deltaLO);</td>
</tr>
<tr>
<td>Interface</td>
<td>IEmbededLODiagnostic</td>
</tr>
</tbody>
</table>
# LODenominator Property

**Description**
Sets or returns the denominator value of the LO Fractional Multiplier.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

**VB Syntax**
```
obj.LODenominator (n) = value
```

**Variable**
- `obj` (Type) - Description
  - `obj` A Mixer Interface pointer to the Measurement (object)
  - Or
  - A Converter Object

- `value` (Long) - LO denominator value
- `n` (Long) - LO stage number

Choose from 1 or 2

**Return Type** Long

**Default** 1

**Examples**
```
Print mixer.LODenominator(1) 'prints the value of the first LODenominator
```

**C++ Syntax**
```
HRESULT get_LODenominator(long *pVal)

HRESULT put_LODenominator(long newVal)
```

**Interface**
- IMixer
- IConverter
### LOFixedFrequency Property

**Description**  
Sets or returns the LO frequency value.  

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**  
`obj.LOFixedFrequency (n) = value`

**Variable**  
- `obj`  
A Mixer Interface pointer to the Measurement (object)

Or

A Converter Object

- `value`  
(double) - LO Frequency in Hertz.

- `n`  
(Long) - LO stage number  
Choose from 1 or 2

**Return Type**  
Double

**Default**  
0 Hz

**Examples**  
`Print mixer.LOFixedFrequency(1) 'prints the value of the first LO fixed frequency`

**C++ Syntax**  
`HRESULT get_LOFixedFrequency(double *pVal)`

`HRESULT put_LOFixedFrequency(double newVal)`

**Interface**  
IMixer

IConverter
**LOFrequencyDelta Property**

**Description**
Sets and returns LO Frequency Delta. There is usually no need to set this value. Read this value to determine the difference between the LO Frequency that is stated in the Mixer dialog box and the last measured LO Frequency.

**VB Syntax**
```vbnet
obj.LOFrequencyDelta = value
```

**Variable**
- **obj**
  - An `EmbeddedLO (object)` or
  - A `ConverterEmbeddedLO (object)`

- **value**
  - (Double) LO Frequency delta in Hertz.

**Return Type**
(Double)

**Default**
Not Applicable

**Examples**
```vbnet
embedLO.LOFrequencyDelta = 0 'write
value = embedLO.LOFrequencyDelta 'read
```

**C++ Syntax**
```cpp
HRESULT get_LOFrequencyDelta(double* val);

HRESULT put_LOFrequencyDelta(double val);
```

**Interface**
IEmbededLO
**LogMagnitudeOffset Property** Superseded

**Description**

*Note: This property is replaced by DoReceiverPowerCal Method.*

For **Receiver Calibration** - Sets or returns the value to offset the normalized unratioed power measurement data. The unratioed power measurement is effectively calibrated to the power level specified by the value of LogMagnitudeOffset as soon as the Normalization property is set to ON after calling the DataToDivisor method.

To offset the data trace magnitude a specified value, use **MagnitudeOffset Property**

**VB Syntax**

```vbnet
meas.LogMagnitudeOffset = value
```

**Variable**

- **meas** *(object)* - A Measurement object
- **value** *(double)* - Offset value in dBm.

**Return Type**

Double

**Default**

0

**Examples**

```vbnet
meas.LogMagnitudeOffset = -10 'Write (-10 dBm)
calpower = meas.LogMagnitudeOffset 'Read
meas.DataToDivisor 'Store meas data as measurement divisor
meas.Normalize = 1 'Measurement is now calibrated to -10 dBm
```

**C++ Syntax**

```cpp
HRESULT put_LogMagnitudeOffset(double newVal);
HRESULT get_LogMagnitudeOffset(double *pVal);
```

**Interface**

IMeasurement
**LOName Property**

**Description**
Sets or returns the name of the VNA internal source or external source to use as the LO in an FCA measurement.

**VB Syntax**

```vbnet
obj.LOName(n) = value
```

**Variable**

- `obj` (Type) - Description
  - A Mixer Interface pointer to the Measurement (object)
  - Or
  - A Converter Object

- `n` (Long) - LO stage number
  - Choose from 1 or 2

- `value` (string) - LO Source name. Use `SourcePortNames Property` to return a list of valid source ports. An external source must be configured and selected to be valid. Learn more about external source configuration.

**Note:** If the port is defined by a string name, such as an external source or one of the Source 2 outputs on the 2-port, 2-source VNA-x model, then you must use `cap.getPortNumber` to translate the string into a port number. To learn more see `[Remotely Specifying a Source Port](#)`.

**Return Type**
String

**Default**
"Not controlled"

**Examples**

```
mixer.LOName(1) = "MySource"
```

**C++ Syntax**

- HRESULT `get_LOName(string *pVal)`
- HRESULT `put_LOName(string newVal)`

**Interface**
MIXer
IConverter
### LONumerator Property

**Description**
Sets or returns the numerator value of the LO Fractional Multiplier.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**

```vbnet
obj.LONumerator (n) = value
```

**Variable**

- `obj` A Mixer Interface pointer to the Measurement (object)
  
  Or

- A Converter Object

- `value` (Long) - LO denominator value

- `n` (Long) - LO stage number

Choose from 1 or 2

**Return Type**

Long

**Default**

1

**Examples**

```vbnet
Print mixer.LONumerator(1) 'prints the value of the first LO Numerator
```

**C++ Syntax**

```cpp
HRESULT get_LONumerator(long *pVal)

HRESULT put_LONumerator(long newVal)
```

**Interface**

IMixer

IConverter
## LOPower Property

**Description**: Sets or returns the value of LO Power.

**VB Syntax**: `obj.LOPower(n) = value`

**Variable** *(Type) - Description*

- **obj**: A Mixer Interface pointer to the Measurement (object)
  
  Or

- A Converter Object

- **n**: (Long) - LO stage number
  
  Choose from 1 or 2

- **value**: (double) - LO Power in dBm.

**Return Type**: Double

**Default**: -10dBm

**Examples**: `Print mixer.LOPower(1)` 'prints the value of the LO Power'

**C++ Syntax**

- `HRESULT get_LOPower(double *pVal)`
  
  `HRESULT put_LOPower(double newVal)`

**Interface**: IMixer

IConverter
### LORangeMode Property

**Description**
Sets or returns the LO sweep mode.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**Note:** There is also a LORangeMode on the Converter Object.

**VB Syntax**

```
obj.LORangeMode (n) = value
```

**Variable (Type) - Description**

- **obj** A Mixer Interface pointer to the Measurement (object)
- **n** (Long) - LO stage number.

Choose from 1 or 2

- **value** (Enum as MixerRangeMode) - LO sweep mode. Choose from:
  - 0 - mixSwept
  - 1 - mixFixed

**Return Type**
Enum

**Default**
0 - mixSwept

**Examples**

```
mixer.LORangeMode(1)=mixSwept
```

**C++ Syntax**

```c++
HRESULT get_LORangeMode(long LO, enum *pVal)

HRESULT put_LORangeMode(long LO, enum newVal)
```

**Interface**
IMixer3
**LORangeMode Property**

**Description**
Sets or returns the LO sweep mode.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

Note: There is also a LORangeMode on the Mixer Interface

**VB Syntax**

```
obj.LORangeMode (n) = value
```

**Variable**

(Type) - Description

- **obj**: A Converter Object
- **n**: (Long) - LO stage number.
  - Choose from 1 or 2
- **value**: (Enum as NARangeMode) - LO sweep mode. Choose from:
  - 0 - naSwept
  - 1 - naFixed

**Return Type**
Enum

**Default**
0 - naSwept

**Examples**

```
mixer.LORangeMode(1)=naSwept
```

**C++ Syntax**

```
HRESULT get_LORangeMode(long LO, enum *pVal )

HRESULT put_LORangeMode(long LO, enum newVal)
```

**Interface**
IConverter
# Loss Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Returns the insertion loss for the calibration standard.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>calstd.loss = value</code></td>
</tr>
</tbody>
</table>

**Variable** (Type) - Description

- `value` (single) - Insertion loss in Gohms / sec. (Giga Ohms per second of electrical delay)

**Return Type** Single

**Default** Not Applicable

**Examples**

```vbnet
calstd.loss = 3.5 'Write 3.5 Gohms of loss
stdLoss = calstd.loss 'Read the value of Loss
```

**C++ Syntax**

- `HRESULT get_Loss(float *pVal)`
- `HRESULT put_Loss(float newVal)`

**Interface** ICalStandard
## Loss (Power Segment) Property

### Description
Sets or returns the loss value associated with a PowerLossSegment.

### VB Syntax
```
lossSeg.Loss = value
```

### Variable (Type) - Description
- **lossSeg**: A PowerLossSegment (object)
- **value**: (double) – Loss value in dB. Loss is entered as a POSITIVE number.

### Return Type
Double

### Default
0

### Examples
```
lossSeg.Loss = 0.5 'Write
lossVal = lossSeg.Loss 'Read
```

### C++ Syntax
```
HRESULT put_Loss(Double newVal);
HRESULT get_Loss(Double *pVal);
```

### Interface
IPowerLossSegment
## LOStage Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the number of LO stages present in the mixer.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>obj.LOStage = value</code></td>
</tr>
<tr>
<td>Variable (Type)</td>
<td>- Description</td>
</tr>
<tr>
<td><code>obj</code></td>
<td>A Mixer Interface pointer to the Measurement (object)</td>
</tr>
<tr>
<td>Or</td>
<td>A Converter Object</td>
</tr>
<tr>
<td><code>value</code> (Long)</td>
<td>Number of LO stages. Choose from 1 or 2</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Default</td>
<td>1</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mixer.LOStage = 1</code> 'sets the LO Stage value to 1'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_LOStage(long *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_LOStage(long newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMixer</td>
</tr>
<tr>
<td></td>
<td>IConverter</td>
</tr>
</tbody>
</table>
## LOStartFrequency Property

**Description**
Sets or returns the LO start frequency value. This command can only be used with SMC (not VMC) measurements.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**
```vbnet
obj.LOStartFrequency(n) = value
```

**Variable**
- **obj**: A Mixer Interface pointer to the Measurement (object)
- Or
- A Converter Object

**value**: (double) - LO Start Frequency in Hertz.

**n**: (Long) - LO stage number
Choose from 1 or 2

**Return Type**: Double

**Default**: Not Applicable

**Examples**
```vbnet
Print mixer.LOStartFrequency(1) 'prints the value of the first LO start frequency
```

**C++ Syntax**
```c++
HRESULT get_LOStartFrequency(long id, double *pVal)
HRESULT put_LOStartFrequency(long id, double newVal)
```

**Interface**
- IMixer3
- IConverter
**LOStartPower Property**

**Description**
Sets or returns the Start value of a LO Power sweep. Also set imdx.SweepType to naIMDLOPowerSweep.

**VB Syntax**

```vbnet
obj.LOStartPower(n) = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>obj</code></td>
<td>A Converter Object</td>
</tr>
<tr>
<td><code>n</code></td>
<td>(Long) - LO stage number. Choose 1</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) - LO start power in dBm.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
-20 dBm

**Examples**

```vbnet
convtr.LOStartPower(1) = -5 'Sets the LO Power sweep start value
start = convtr.LOStartPower(1) 'Reads the start value
```

**C++ Syntax**

```c
HRESULT get_LOStartPower(long LOStage, double *pVal)
HRESULT put_LOStartPower(long LOStage, double newVal)
```

**Interface**
IConverter
**LOStopFrequency Property**

**Description**
Sets or returns the LO stop frequency value. This command can only be used with SMC (not VMC) measurements.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**

```vbnet
obj.LOStopFrequency (n) = value
```

**Variable** *(Type)* - Description

- **obj**
  - A Mixer Interface pointer to the Measurement (object)
  - Or
  - A Converter Object

- **value** *(double)* - LO Stop Frequency in Hertz.

- **n** *(Long)* - LO stage number
  - Choose from 1 or 2

**Return Type**
Double

**Default**
Not Applicable

**Examples**

```vbnet
Print mixer.LOStopFrequency(1) 'prints the value of the first LO stop frequency
```

**C++ Syntax**

```cpp
HRESULT get_LOStopFrequency(long id, double *pVal)
HRESULT put_LOStopFrequency(long id, double newVal)
```

**Interface**
IMixer3

IConverter
## LOStopPower Property

**Description**
Sets or returns the Stop value of a LO Power sweep. Also set `imdx.SweepType` to `naIMDLOPowerSweep`.

**VB Syntax**

```
obj.LOStopPower (n) = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>- Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>obj</code></td>
<td>A Converter Object</td>
</tr>
<tr>
<td><code>n</code></td>
<td>(Long) - LO stage number. Choose 1</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) - LO Stop power in dBm.</td>
</tr>
</tbody>
</table>

**Return Type**

Double

**Default**

-10 dBm

**Examples**

```vbnet
cnvtr.LOStopPower(1) = -5 'Sets the LO Power sweep Stop value
Stop = cnvtr.LOStopPower(1) 'Reads the Stop value
```

**C++ Syntax**

```c
HRESULT get_LOStopPower(long LOStage, double *pVal)

HRESULT put_LOStopPower(long LOStage, double newVal)
```

**Interface**

IConverter
### LowAmplitude Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the Low amplitude (voltage) of the pulse generator.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>extPulseGen.LowAmplitude = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>extPulseGen</code></td>
<td>An <code>ExternalPulseGenerator</code> <strong>(object)</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(Double)</strong> Pulse Generator low amplitude voltage.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>extPulseGen.LowAmplitude = 4</code> <strong>(Write)</strong></td>
</tr>
<tr>
<td></td>
<td><code>lo = extPulseGen.LowAmplitude</code> <strong>(Read)</strong></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_LowAmplitude (double *pValue)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_LowAmplitude (double newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IExternalPulseGenerator</td>
</tr>
</tbody>
</table>
### Description
Enables/disables the low frequency extension (LFE). Learn more.

### VB Syntax
`chan.LowFrequencyExtension = value`

### Variable (Type) - Description

- **value** *(boolean)* - Choose either:
  - **False** - LFE is disabled.
  - **True** - LFE is enabled.

### chan Channel (object)

### Return Type
Boolean

### Default
False

### Examples
- `chan.LowFrequencyExtension = True` *(Write)*
- `lfestate = chan.LowFrequencyExtension` *(Read)*

### C++ Syntax
```cpp
HRESULT get_LowFrequencyExtension(VARIANT_BOOL* lfestate);
HRESULT put_LowFrequencyExtension(VARIANT_BOOL lfestate);
```

### Interface
IChannel26
LXI Device ID State Property

Description

Opens and closes the LAN Status dialog with the LAN Status Indicator showing IDENTIFY.

The VNA supports this capability to satisfy a requirement of the LAN eXtensions for Instrumentation (LXI) standard. Changing the value of this property is the same operation that occurs when clicking the Toggle LXI Identification button on the Welcome web page presented by the VNA web server.

VB Syntax

```vbnet
app.LXIDeviceIDState = bool
```

Variable (Type) - Description

- `app` An Application (object)
- `bool` (boolean) Choose from:

  - **True** - Displays the LAN Status dialog with the Status Indicator showing IDENTIFY.
  - **False** -
    
      - If the dialog had been opened by this property, then the LAN Status dialog is closed.
      - If the dialog was opened manually, then it will stay open.

Return Type

Boolean

Default

False

Examples

```vbnet
app.LXIDeviceIDState = True 'Write
value = app.LXIDeviceIDState 'Read
```

C++ Syntax

```csharp
HRESULT get_LXIDeviceIDState( VARIANT_BOOL* pState);

HRESULT put_LXIDeviceIDState( VARIANT_BOOL  state);
```

Interface

IApplication14
**MagnitudeOffset Property**

**Description**
Offsets the data trace magnitude by the specified value.

To offset the data trace magnitude to a slope value that changes with frequency, use `MagnitudeSlopeOffset` Property.

To implement a Receiver Cal offset, use `LogMagnitudeOffset` property.

**VB Syntax**

```
meas.MagnitudeOffset = value
```

**Variable (Type) - Description**

- `meas` (object) - A Measurement object
- `value` (double) - Offset value in dB.

**Return Type**
Double

**Default**
0

**Examples**

```
meas.MagnitudeOffset = 4 'Write
offs = meas.MagnitudeOffset 'Read
```

**C++ Syntax**

```
HRESULT put_MagnitudeOffset(double newVal);
HRESULT get_MagnitudeOffset(double *pVal);
```

**Interface**
IMeasurement4
### MagnitudeSlopeOffset Property

**Description**
Offsets the data trace magnitude to a value that changes linearly with frequency. The offset slope begins at 0 Hz.

To offset the entire data trace magnitude by a specified value, use `MagnitudeOffset` Property.

**VB Syntax**
```vbnet
meas.MagnitudeSlopeOffset = value
```

**Variable**
- `meas` (object) - A Measurement object
- `value` (double) - Offset slope value in dB / 1GHz.

**Return Type**
Double

**Default**
0

**Examples**
```vbnet
meas.MagnitudeSlopeOffset = 4 'Writes a slope offset of 4dB/1GHz.
offs = meas.MagnitudeSlopeOffset 'Read
```

**C++ Syntax**
```cpp
HRESULT put_MagnitudeSlopeOffset(double newVal);
HRESULT get_MagnitudeSlopeOffset(double *pVal);
```

**Interface**
IMeasurement4
MainToneIFBandwidth Property

Description
Sets and returns the IF Bandwidth for measurement of the Main tones.

VB Syntax
```vb
imd.MainToneIFBandwidth = value
```

Variable (Type) - Description

- `imd` (A SweptIMD Object)
- `value` (Double) IF Bandwidth in Hz. Choose from:
  - 1 | 2 | 3 | 5 | 7 | 10 | 15 | 20 | 30 | 50 | 70 |
  - 100 | 150 | 200 | 300 | 500 | 700 | 1k | 1.5k | 2k | 3k | 5k | 7k | 10k | 15k | 20k | 30k | 50k |
  - 70k | 100k | 150k | 200k | 280k | 360k | 600k

Learn more about setting IFBW for IMD.

If an invalid number is specified, the analyzer will round up to the closest valid number.

- **Return Type**: Double
- **Default**: 1 kHz

Examples
```vb
imd.MainToneIFBandwidth = 2e3 'Write
value = imd.MainToneIFBandwidth 'Read
```

C++ Syntax
```cpp
HRESULT get_MainToneIFBandwidth(double *pVal)
HRESULT put_MainToneIFBandwidth(double newVal)
```

Interface
ISweptIMD
## MarkCoupControlsMkrState Property

**Write/Read**  

**Description**  Set and return whether the Coupled Markers setting controls the ON|OFF state of markers that are coupled. Learn more about Coupled Markers. Refer also to CoupledMarkers Property.

**VB Syntax**  

```
pref.MarkCoupControlsMkrState = value
```

**Variable**  

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pref</code></td>
<td>A Preferences (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean) - Choose from:</td>
</tr>
</tbody>
</table>

- **True** - With Coupled Markers ON, when a marker is turned on, the same-numbered marker on all coupled traces will also be turned on. Likewise, turning off a marker will turn it off on all coupled traces.

- **False** - Turning a marker on or off will have no effect on the markers on other traces.

**Return Type**  Boolean

**Default**  False

**Examples**  

```
pref.MarkCoupControlsMkrState = True 'Write

value = pref.MarkCoupControlsMkrState 'Read
```

**C++ Syntax**  

```
HRESULT get_MarkCoupControlsMkrState (VARIANT_BOOL* preference);

HRESULT put_MarkCoupControlsMkrState (VARIANT_BOOL val)
```

**Interface**  IPreferences15
MarkCoupMethPresetIsChan Property

Set and return whether Coupled Markers is set to Channel or All after Preset. Learn more about Coupled Markers. Refer also to CoupledMarkers Property and MarkCoupPresetIsOn.

**VB Syntax**

`pref.MarkCoupMethPresetIsChan = value`

**Variable**

- **pref** (Type) - Description
  
  A Preferences (object)

- **value** (Boolean) - Choose from:
  
  **True** - Marker Coupling Method is set to Channel after Preset.

  **False** - Marker Coupling Method is set to ALL after Preset.

**Return Type**

Boolean

**Default**

False

**Examples**

```vbnet
pref.MarkCoupMethPresetIsChan = True 'Write
value = pref.MarkCoupMethPresetIsChan 'Read
```

**C++ Syntax**

```cpp
HRESULT get_MarkCoupMethPresetIsChan (VARIANT_BOOL* preference);
HRESULT put_MarkCoupMethPresetIsChan (VARIANT_BOOL val)
```

**Interface**

IPreferences15
# MarkCoupPresetIsOn Property

**Write/Read**

**Description**
Set and return whether Coupled Markers is set to ON or OFF after Preset. Learn more about Coupled Markers.

**VB Syntax**

```
pref.\[MarkCoupPresetIsOn\] = value
```

**Variable**

- **Type**: Description
- **pref** A Preferences *(object)*
- **value** *(Boolean)* - Choose from:
  - **True** - Coupled Markers is ON after Preset.
  - **False** - Coupled Markers is OFF after Preset.

**Return Type**

Boolean

**Default**

False

**Examples**

```
pref.MarkCoupPresetIsOn = True 'Write
value = pref.MarkCoupPresetIsOn 'Read
```

**C++ Syntax**

```
HRESULT get_MarkCoupPresetIsOn (VARIANT_BOOL* preference);
HRESULT put_MarkCoupPresetIsOn (VARIANT_BOOL val)
```

**Interface**

IPreferences15
**MarkerAnnotation Property**

**Description**  
Returns the Y-axis marker value from the specified tuning sweep. Use `IsMarkerOn` to confirm if a marker was used for the tuning sweep.

**VB Syntax**  
```vb
value = embedLODiag.MarkerAnnotation(n)
```

**Variable**  
*(Type)* - Description

- **value** *(String)* Variable to store the returned data.
- **embedLODiag** An `EmbeddedLODiagnostic` *(object)*
- **n** *(Long)* Tuning sweep number. Use `NumberOfSweeps` to find the number of sweeps taken.

**Default**  
Not Applicable

**Examples**  
```vb
data= embedLO.MarkerAnnotation 3 'read
```

**C++ Syntax**  
```cpp
HRESULT MarkerAnnotation(long sweep, BSTR* annotation);
```

**Interface**  
`IEmbeddedLODiagnostic`
## MarkerFormat Property

**Description**
Sets the format of all the markers in the measurement. To override this setting for an individual marker, use `mark.Format`.

**VB Syntax**
`meas.MarkerFormat = value`

**Variable**
- `meas` (Type) - Description: A Measurement (object)
- `value` (enum NAMarkerFormat) - Choose from:
  - 0 - naMarkerFormat_LinMag
  - 1 - naMarkerFormat_LogMag
  - 2 - naMarkerFormat_Phase
  - 3 - naMarkerFormat_Delay
  - 4 - naMarkerFormat_Real
  - 5 - naMarkerFormat_Imaginary
  - 6 - naMarkerFormat_SWR
  - 7 - naMarkerFormat_LinMagPhase
  - 8 - naMarkerFormat_LogMagPhase
  - 9 - naMarkerFormat_RealImaginary
  - 10 - naMarkerFormat_ComplexImpedance
  - 11 - naMarkerFormat_ComplexAdmittance
  - 12 - naMarkerFormat_Kelvin
  - 13 - naMarkerFormat_Fahrenheit
  - 14 - naMarkerFormat_Centigrade

**Return Type**
Not Applicable

**Default**
1 - naMarkerFormat_LogMag

**Examples**
```vbnet
meas.MarkerFormat = naMarkerFormat_SWR 'Write
```
**C++ Syntax**
HRESULT put_MarkerFormat(tagNAMarkerFormat NewFormat)

**Interface**
IMeasurement
InterpolateMarkers Method

Description
Turns all Marker Interpolation ON and OFF for the measurement. Marker interpolation enables X-axis resolution between the discrete data values. The analyzer will calculate the X and Y-axis data values between discrete data points. To override this property for individual markers, use the Interpolated property.

VB Syntax
meas.Interpolate = value

Variable (Type) - Description
meas A Measurement (object)
value (boolean)
False - Turns interpolation OFF for all markers in the measurement
True - Turns interpolation ON for all markers in the measurement

Return Type
Boolean
Default
True (ON)

Examples
meas.Interpolate = 1

C++ Syntax
HRESULT InterpolateMarkers(VARIANT_BOOL bNewVal)

Interface
IMeasurement
### Number Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of the marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>marknum = mark.Number</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>marknum</code></td>
<td>(long) - Variable to store marker number</td>
</tr>
<tr>
<td><code>mark</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>marknum = mark.Number 'Read</code></td>
</tr>
</tbody>
</table>

#### C++ Syntax

`HRESULT get_Number(long *pVal)`

#### Interface

`IMarker`
# MarkerPosition Property

**Description**  
Returns the X-axis marker position from the specified tuning sweep.

**VB Syntax**  

```vbnet
value = embedLODiag.MarkerPosition(n)
```

**Variable (Type) - Description**

- `value` (**Double**) Variable to store the returned data.

- `embedLODiag` **An EmbeddedLODiagnostic (object)**

- `n` (**Long**) Tuning sweep number. Use `NumberOfSweeps` to find the number of sweeps taken.

**Default**  
Not Applicable

**Examples**  

```vbnet
data= embedLO.MarkerPosition 3 'read
```

**C++ Syntax**  

```c++
HRESULT MarkerPosition(long sweep, double *position);
```

**Interface**  
IEmbededLODiagnostic
## MarkerReadout Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables or disables the readout of markers in the window. To show the marker on the screen use ShowMarkerReadout Method.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>win.MarkerReadout = state</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td>win</td>
<td>A NAWindow (object)</td>
</tr>
<tr>
<td>state</td>
<td>(boolean)</td>
</tr>
<tr>
<td>True</td>
<td>enables marker readout</td>
</tr>
<tr>
<td>False</td>
<td>disables marker readout</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>True</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>win.MarkerReadout = True 'Write</code></td>
</tr>
<tr>
<td> </td>
<td><code>State = app.ActiveNAWindow.MarkerReadout 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_MarkerReadout(VARIANT_BOOL *pVal)</code></td>
</tr>
<tr>
<td> </td>
<td><code>HRESULT put_MarkerReadout(VARIANT_BOOL newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>INAWindow</td>
</tr>
</tbody>
</table>
## MarkerReadoutResponsePlaces Property

**Description**
For the Y-axis (response), sets the number digits to display after the decimal point in marker readouts.

**VB Syntax**
```
win.MarkerReadoutResponsePlaces = value
```

**Variable (Type) - Description**
- `win` - A NAWindow (object)
- `value` - (Long Integer) Number of digits to display. Choose a value between 1 and 4.

**Return Type**
Long Integer

**Default**
2

**Examples**
```
win.MarkerReadoutResponsePlaces = 3 'Write
value = app.ActiveNAWindow.MarkerReadoutResponsePlaces 'Read
```

**C++ Syntax**
```
HRESULT get_MarkerReadoutResponsePlaces(long *pVal)

HRESULT put_MarkerReadoutResponsePlaces(long newVal)
```

**Interface**
INAWindow3
MarkerReadoutSize Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the size of font used when displaying Marker Readout in the selected window.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>win.MarkerReadoutSize = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>win</code></td>
<td>A NAWindow (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum NAFontSize)</td>
</tr>
<tr>
<td>0</td>
<td><code>naDefault</code> - marker readout appears in default font size</td>
</tr>
<tr>
<td>1</td>
<td><code>naLarge</code> - marker readout appears in large font size</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long Integer</td>
</tr>
<tr>
<td>Default</td>
<td><code>naDefault</code></td>
</tr>
<tr>
<td>Examples</td>
<td><code>win.MarkerReadoutSize = naDefault 'write default size for marker readout</code></td>
</tr>
<tr>
<td></td>
<td><code>Dim Size As NAFontSize</code></td>
</tr>
<tr>
<td></td>
<td><code>Size = app.ActiveNAWindow.MarkerReadoutSize 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_MarkerReadoutSize(tagNAFontSize *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_MarkerReadoutSize(tagNAFontSize newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>INAWindow</td>
</tr>
</tbody>
</table>
## MarkerReadoutsPerTrace Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the number of marker readouts to display per trace. Display up to 20 marker readouts per window.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>win.MarkerReadoutsPerTrace = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>win</code></td>
<td>A NAWindow (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long Integer) Number of marker readouts to display. Choose a value between 1 and 10.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>5</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>win.MarkerReadoutsPerTrace = 3 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = app.ActiveNAWindow.MarkerReadoutsPerTrace 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_MarkerReadoutsPerTrace(long *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_MarkerReadoutsPerTrace(long newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>INAWindow3</td>
</tr>
</tbody>
</table>
**MarkerReadoutStimulusPlaces Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>For the X-axis (stimulus), sets the number digits to display after the decimal point in marker readouts.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>win.MarkerReadoutStimulusPlaces = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>win</code></td>
<td>A NAWindow (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long Integer)</em> Number of digits to display. Choose a value between 2 and 6.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>3</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>win.MarkerReadoutStimulusPlaces = 2 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = app.ActiveNAWindow.MarkerReadoutStimulusPlaces 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_MarkerReadoutStimulusPlaces(long *pVal)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_MarkerReadoutStimulusPlaces(long newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>INAWindow3</td>
</tr>
</tbody>
</table>
### MarkerReadoutXPosition Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the X-axis position of marker readouts. Readouts are right-justified at the specified position.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>win.MarkerReadoutXPosition = value</code></td>
</tr>
</tbody>
</table>
| **Variable** | *(Type)* - Description  
  *win* A NAWindow (object)  
  *value* (Double) X-axis position. Choose a value between 1 (far left) and 10 (far right). |
| **Return Type** | Double |
| **Default** | 10 |
| **Examples** |  
  `win.MarkerReadoutXPosition = 3 'Write`  
  `value = app.ActiveNAWindow.MarkerReadoutXPosition 'Read` |
| **C++ Syntax** | HRESULT get_MarkerReadoutXPosition(double *pVal)  
  HRESULT put_MarkerReadoutXPosition(double newVal) |
| **Interface** | INAWindow3 |
### MarkerReadoutYPosition Property

**Description**
Sets the Y-axis position of marker readouts. Readouts are right-justified at the specified position.

**VB Syntax**
```
win.MarkerReadoutYPosition = value
```

**Variable**
- **(Type)** - Description
  - **win**: A NAWindow (object)
  - **value**: (Double) Y-axis position. Choose a value between 1 (bottom) and 10 (top).

**Return Type**
Double

**Default**
10

**Examples**
```
win.MarkerReadoutYPosition = 3 'Write
value = app.ActiveNAWindow.MarkerReadoutYPosition 'Read
```

**C++ Syntax**
```
HRESULT get_MarkerReadoutYPosition(double *pVal)
HRESULT put_MarkerReadoutYPosition(double newVal)
```

**Interface**
INAWindow3
## Markers Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return the color of data trace markers for nth trace in a window.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>trace(n).Markers = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>trace(n)</strong> - Description</td>
</tr>
<tr>
<td></td>
<td>One of the 8 ComTraceColors objects</td>
</tr>
<tr>
<td></td>
<td><strong>value</strong> <em>(Long Integer)</em> - RGB color of the Markers pen.</td>
</tr>
</tbody>
</table>

Convert the three RGB colors to an integer as follows:

\[
RGB = R + (G \times 2^8) + (B \times 2^{16})
\]

To find the three RGB values from the Display Colors dialog, click **Change Color**, then **Define Custom Color**.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Long</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Varies for each trace.</td>
</tr>
</tbody>
</table>

| Examples    | R = 10  
|            | G = 10  
|            | B = 10  
|            | RGB = 10 + (10 \times 2^8) + (10 \times 2^{16})  
|            | `tracel.Markers = RGB 'Write`  
|            | `color = tracel.Markers 'Read`  |

| C++ Syntax  | HRESULT get_Markers(long* pVal);  
|            | HRESULT put_Markers(long newVal);  |

| Interface   | ITraceColors |
**MarkerState Property**

**Description**
Sets or returns the ON / OFF state of the specified marker.

**VB Syntax**
```vbnet
meas.MarkerState (n) = state
```

**Variable**
(Type) - Description
- **meas**
  A Measurement (object)
- **n**
  (Long Integer) Marker number to turn on or off.
- **state**
  (boolean) -
  True - turns the specified marker ON
  False - turns the specified marker OFF

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
meas.MarkerState(1) = True
reference = meas.MarkerState(2)
```

**C++ Syntax**
```c++
HRESULT get_MarkerState(long markerNum, VARIANT_BOOL bState)
HRESULT put_MarkerState(long markerNum, VARIANT_BOOL* bState)
```

**Interface**
IMeasurement3
<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the symbol to display for marker position.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>win.MarkerSymbol = value</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>win</code></td>
<td>A NAWindow (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Enum as naMarkerSymbol) Choose from:</td>
</tr>
<tr>
<td>0</td>
<td><code>naMarkerSymbolTriangle</code></td>
</tr>
<tr>
<td>1</td>
<td><code>naMarkerSymbolFlag</code></td>
</tr>
<tr>
<td>2</td>
<td><code>naMarkerSymbolLine</code></td>
</tr>
<tr>
<td>See pictures of symbols</td>
<td></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0 - <code>naMarkerSymbolTriangle</code></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>win.MarkerSymbol = naMarkerSymbolLine 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = app.ActiveNAWindow.MarkerSymbol 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_MarkerSymbol(double *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_MarkerSymbol(double newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>INAWindow3</td>
</tr>
</tbody>
</table>
### MarkerSymbolsAboveTrace Property

**Description**
Specifies whether or not to force marker symbols to be displayed above the trace. When ON, all marker symbols will be displayed above the trace and the active marker will be filled solid.

**VB Syntax**
```
win.MarkerSymbolsAboveTrace = value
```

**Variable (Type) - Description**

- `win` (NAWindow (object))
- `value` (Boolean)

**False** - ONLY the active marker is displayed above the trace. Inactive markers are displayed below the trace.

**True** - ALL marker symbols are displayed above the trace. The active marker is always filled solid.

**Return Type**
Boolean

**Default**
OFF (ON in IM Spectrum and SA measurement classes)

**Examples**
```
win.MarkerSymbolsAboveTrace = True 'Write
```
```
value = app.ActiveNAWindow.MarkerSymbolsAboveTrace 'Read
```

**C++ Syntax**
```
HRESULT get_MarkerSymbolsAboveTrace(VARIANT_BOOL* pbState)
```
```
HRESULT put_MarkerSymbolsAboveTrace(VARIANT_BOOL bState)
```

**Interface**
INAWindow4
### Type (Marker) Property

**Description**  
Sets and reads the marker type.

**VB Syntax**  
`mark.Type = value`

**Variable**  
- **chan**  
  A Marker (object)

- **value**  
  (enum NAMarkerType) - Marker Type. Choose from:
  - **0** - `naMarkerType_Normal` - the X-axis value for a normal marker will always be determined by the measurement data of the marker.
  - **1** - `naMarkerType_Fixed` - retains and keeps its x-axis value at the time the marker type is set.

**Return Type**  
Long Integer

**Default**  
`naMarkerType_Normal`

**Examples**  
- `mark.Type = naMarkerType_Normal` 'Write
- `MrkType = mark.Type` 'Read

**C++ Syntax**  
- `HRESULT get_Type(tagNAMarkerType *pVal)`
- `HRESULT put_Type(tagNAMarkerType newVal)`

**Interface**  
IMarker
Stimulus Property

Description
Sets and reads the X-Axis value of the marker.

If the marker is a delta marker, the value will be relative to the reference marker.

See Reference Marker example below.

See Also: Read Y-axis value.

VB Syntax
mark.Stimulus = value

Variable
(mark) - Description
(mark) A Marker (object)
(value) (double) - X-Axis value. Choose any number within the full span of the channel or User Range (if set).

Return Type
Double

Default
First activated Marker turns ON in the middle of the X-axis range. Subsequent markers turn ON at the position of the most recently active marker.

Examples
mark.Stimulus = 3e9 'Write
XVal = mark.Stimulus 'Read

C++ Syntax
HRESULT get_Stimulus(double *pVal)
HRESULT put_Stimulus(double newVal)

Interface
IMarker
Read-only

Value Property

Description
Reads the Y-axis value of the marker. You cannot set the Y-axis value of a marker. The marker remains at the position at the time you set `marker.Type`.

If the marker is a delta marker, the value will be relative to the reference marker.

See Reference marker example below.

See Also: Set and Read X-axis value.

**Note:** To accurately read the marker Y-axis value with trace smoothing applied, the requested format must match the displayed format. Otherwise, the returned value is un-smoothed data. For example, to read the smoothed marker value when measuring group delay, both the display format and the marker format must be set to (Group) Delay.

---

**VB Syntax**

`YValue = mark.Value (format)`

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>YValue</code></td>
<td></td>
<td>A variable to store the Y-axis value</td>
</tr>
<tr>
<td><code>mark</code></td>
<td></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><code>format</code></td>
<td>(enum NAMarkerFormat)</td>
<td>The format in which to return the marker's Y-axis value. The number in parenthesis following the format is the number of values that are returned in a variant array. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naMarkerFormat_LinMag (1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 - naMarkerFormat_LogMag (1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 - naMarkerFormat_Phase (1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 - naMarkerFormat_Delay (1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 - naMarkerFormat_Real (1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 - naMarkerFormat_Imaginary (1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6 - naMarkerFormat_SWR (1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7 - naMarkerFormat_LinMagPhase (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8 - naMarkerFormat_LogMagPhase (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>naMarkerFormat_RealImaginary (2)</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>naMarkerFormat_ComplexImpedance (3)</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>naMarkerFormat_ComplexAdmittance (3)</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>naMarkerFormat_Kelvin (1)</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>naMarkerFormat_Fahrenheit (1)</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>naMarkerFormat_Celsius (1)</td>
<td></td>
</tr>
</tbody>
</table>

Return Type: Variant - The (parens) in the previous list of formats indicates the number of values that are returned in a variant array

Default: Not applicable

Examples:

```
YVal = mark.Value(0) 'Read

' or

YVal = mark.Value(naMarkerFormat\_LinMag)
```

C++ Syntax:
```
HRESULT get_Value(tagNAMarkerFormat format, VARIANT *pVal)
```

Interface: IMarker
### MatchCorrectPower Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns match-correction ON or OFF. Use this command AFTER performing a Guided Power Cal. <a href="#">Learn more</a>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>corrMethods.MatchCorrectPower = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>corrMethods</code></td>
<td>CorrectionMethods (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean)</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>Turns match-correction ON</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>Turns match-correction OFF</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>True</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>corrMethods.MatchCorrectPower = True</code></td>
</tr>
</tbody>
</table>
| **C++ Syntax** | `HRESULT get_MatchCorrectPower(VARIANT_BOOL* val);`  
`HRESULT put_MatchCorrectPower(VARIANT_BOOL newVal);` |
| **Interface** | ICorrectionMethods |
### MaximumFrequency Property

Sets and Returns the maximum frequency for the calibration standard.

**VB Syntax**

```vbnet
calstd.MaximumFrequency = value
```

**Variable**

- **calstd**: A CalStandard *(object)*. Use calKit.GetCalStandard to get a handle to the standard.
- **value**: *(double)* - Maximum frequency in Hertz.

**Return Type**

Double

**Default**

Not Applicable

**Examples**

- `calstd.MaximumFrequency = 9e9` *(Write)*
- `maxFrequency = calstd.MaximumFrequency` *(Read)*

**C++ Syntax**

```cpp
HRESULT get_MaximumFrequency(double *pVal)
HRESULT put_MaximumFrequency(double newVal)
```

**Interface**

ICalStandard
## MaximumFrequency Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the maximum frequency of the remote VNA, including any over-sweep. Over-sweep frequencies can be set but are not specified.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.MaximumFrequency</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) - Variable to store the returned maximum frequency of the VNA.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.MaximumFrequency 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_MaximumFrequency(&amp;frequencyMax);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities</td>
</tr>
</tbody>
</table>
### MaximumFrequency Property

**Description**
Set and returns the maximum usable frequency specified for the power sensor.

**VB Syntax**
```vbnet
pwrSensor.MaximumFrequency = value
```

**Variable (Type) - Description**

- `pwrSensor` A `PowerSensor` (Object) or a `PowerSensorAsReceiver` (Object)
- `value` (double) - Frequency in Hertz.

**Return Type**
Double

**Default**
Device dependent

**Examples**
```vbnet
pwrSensor.MaximumFrequency = 6e9 'Write
MaxFreq = pwrSensor.MaximumFrequency 'Read
```

**C++ Syntax**
```cpp
HRESULT put_MaximumFrequency(double newVal);
HRESULT get_MaximumFrequency(double *pVal);
```

**Interface**
- `IPowerSensor`
- `IPowerSensorAsReceiver`
## MaximumIFFrequency Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Maximum allowed value for the IFFrequency Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = IfConfig.MaximumIFFrequency</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(double)</em> Variable to store the returned maximum allowed frequency that can be applied to the IFFrequency Property.</td>
</tr>
<tr>
<td><strong>IfConfig</strong></td>
<td>IFConfiguration (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>val = App.ActiveChannel.IFConfiguration.MaximumIFFrequency</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_MaximumIFFrequency( double * pMaxFreq);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IIFConfiguration3</td>
</tr>
</tbody>
</table>
**MaximumIterationsPerPoint Property**

**Description**
This command, along with IterationsTolerance Property deal with adjustments made to the source power.

Sets the maximum number of readings to take at each data point for iterating the source power. Power readings will continue to be made, and source power adjusted, until a reading is within the IterationsTolerance value or this max number of readings has been met. The last value to be read is the valid reading for that data point.

The following two commands allow for settling of power readings:

- **ReadingsPerPoint Property**
- **ReadingsTolerance Property**

**VB Syntax**

```vbnet
pwrCal.MaximumIterationsPerPoint = value
```

**Variable**

- **(Type)** - Description
- **pwrCal** - A `SourcePowerCalibrator` object
- **value** - Maximum number of readings. Choose any number between 1 and 100.

**Return Type**
Long Integer

**Default**
5

**Examples**

```vbnet
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.MaximumIterationsPerPoint = 5 'Write
MaxReads = powerCalibrator.MaximumIterationsPerPoint 'Read
```

**C++ Syntax**

```csharp
HRESULT get_MaximumIterationsPerPoint( long *pVal);

HRESULT put_MaximumIterationsPerPoint( long newVal);
```

**Interface**
ISourcePowerCalibrator3
**MaximumNumberOfChannels Property**

Description  Returns the maximum possible number of channels that can be used in the VNA.

**VB Syntax**

```vbnet
value = cap.MaximumNumberOfChannels
```

**Variable**  (Type) - Description

- `value` (Long) - Variable to store the returned maximum value for number of channels.
- `cap` A Capabilities (object)

**Return Type**  Long

**Default**  Not Applicable

**Examples**

```vbnet
value = cap.MaximumNumberOfChannels 'Read
```

**C++ Syntax**

```cpp
HRESULT get_MaximumNumberOfChannels(long *maximumNumberOfChans);
```

**Interface**  ICapabilities2
### MaximumNumberOfTracesPerWindow Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the maximum possible number of traces that can reside in any window.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = cap.MaximumNumberOfTracesPerWindow</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td>value</td>
<td><strong>(Long) - Variable to store the returned maximum value for number of traces.</strong></td>
</tr>
<tr>
<td>cap</td>
<td>A Capabilities <strong>(object)</strong></td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = cap.MaximumNumberOfTracesPerWindow</code> 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_MaximumNumberOfTracesPerWindow(long * maximumNumberOfTraces);</td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities2</td>
</tr>
</tbody>
</table>

1922
**MaximumNumberOfPoints Property**

**Description**
Returns the maximum possible number of data points.

**VB Syntax**
```vbnet
value = obj.MaximumNumberOfPoints
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td>(Long) - Variable to store the returned maximum value for number of points.</td>
</tr>
</tbody>
</table>

**cap**
A Capabilities (object)

or
A GainCompression (object)

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```vbnet
value = cap.MaximumNumberOfPoints 'Read
```

**C++ Syntax**
```csharp
HRESULT get_MaximumNumberOfPoints(double *maximumNumberOfPoints);
```

**Interface**
ICapabilities

IGainCompression
### MaximumNumberOfWindows Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the maximum possible number of windows that can be present on the VNA screen.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = cap.MaximumNumberOfWindows</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td>value</td>
<td><em>(Long)</em> - Variable to store the returned maximum value for number of windows.</td>
</tr>
<tr>
<td>cap</td>
<td>A Capabilities <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = cap.MaximumNumberOfWindows 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_MaximumNumberOfWindows(long * maximumNumberOfWindows );</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities2</td>
</tr>
</tbody>
</table>

---

1924
## MaximumReceiverStepAttenuator Property

**Description**
Returns the maximum amount of receiver attenuation.

**VB Syntax**
```
value = cap.MaximumReceiverStepAttenuator(n)
```

**Variable**
- **Type** - Description
- **value** (Double) - Variable to store the returned value of maximum receiver attenuation.
- **cap** A Capabilities (object)
- **n** (Long) - port number to query for step attenuators

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```
value = cap.MaximumReceiverStepAttenuator
```

**C++ Syntax**
```
HRESULT get_MaximumReceiverStepAttenuator(long portNumber, double * attenuation);
```

**Interface**
ICapabilities
# MaximumSourceALCPower Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a value indicating the maximum amount of source ALC power with 0 dB source attenuation. To calculate the maximum amount of source ALC power with a different level of attenuation, subtract the amount source attenuation from this value.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = cap.MaximumSourceALCPower(n)</code></td>
</tr>
<tr>
<td>Variable (Type) - Description</td>
<td><strong>value</strong> (Double) - Variable to store the returned value for the maximum amount of source ALC power.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td><code>n</code></td>
<td>(Long) - source number to query for maximum ALC power</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = cap.MaximumSourceALCPower</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_MaximumSourceALCPower(long sourceNum, double * power);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities</td>
</tr>
</tbody>
</table>
### MaximumSourceStepAttenuator Property

**Description**
Returns a value for the maximum amount of source attenuation.

**VB Syntax**
```vbnet
value = cap.MaximumSourceStepAttenuator(n)
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Double</td>
<td>Variable to store the returned value for the maximum amount of source attenuation.</td>
</tr>
<tr>
<td>cap</td>
<td>A Capabilities object</td>
<td></td>
</tr>
<tr>
<td>n</td>
<td>Long</td>
<td>port number to query for the maximum amount of source attenuation</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```vbnet
value = cap.MaximumSourceStepAttenuator 2 'Read
```

**C++ Syntax**
```cpp
HRESULT get_MaximumSourceStepAttenuator(long portNumber, double * attenuation );
```

**Interface**
ICapabilities
### MaximumUncertaintyPoints Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the maximum number of points for which uncertainties are to be computed for subsequent calibrations that are performed using Dynamic Uncertainty.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>uncertMan.MaximumUncertaintyPoints = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>uncertMan</code></td>
<td>An <code>UncertaintyManager</code> Object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean) Max number of points.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>500</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>uncertMan.MaximumUncertaintyPoints = 201</code></td>
</tr>
<tr>
<td></td>
<td>See example program</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT <code>get_MaximumUncertaintyPoints([out,retval] long* pNumPoints);</code></td>
</tr>
<tr>
<td></td>
<td>HRESULT <code>put_MaximumUncertaintyPoints([in] long numPoints);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IUncertaintyManager</td>
</tr>
</tbody>
</table>
# MaxOutput Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the &quot;Define Max As&quot; value for an external DC Source.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>extDC.MaxOutput = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>extDC</code></td>
<td>A <code>ExternalDCDevice</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Maximum value for the external DC Source.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>10 V</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>extDC.MaxOutput = 10</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>value = extDC.MaxOutput</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_MaxOutput( double* value);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_MaxOutput( double value);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IExternalDCDevice3</td>
</tr>
</tbody>
</table>
# MaxOutputState Property

Sets and returns the "Define Max As" ON/OFF state for an external DC Source.

## Description
Sets and returns the "Define Max As" ON/OFF state for an external DC Source.

## VB Syntax
`extDC.MaxOutputState = value`

## Variable (Type) - Description
- **extDC**: An `ExternalDCDevice (object)`
- **value**: (Boolean) "Define Max As" ON/OFF state. Choose from:
  - **True**: Turn "Define Max As" ON
  - **False**: Turn "Define Max As" OFF

## Return Type
Boolean

## Default
True

## Examples
```vbnet
extDC.MaxOutputState = True 'Write
value = extDC.MaxOutputState 'Read
```

## C++ Syntax
- `HRESULT get_MaxOutputState(VARIANT_BOOL *value)`
- `HRESULT put_MaxOutputState(VARIANT_BOOL value)`

## Interface
`IExternalDCDevice3`
# MaxPreciseTuningIterations Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the maximum number of tuning iterations to achieve the precise tolerance.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>obj.MaxPreciseTuningIterations = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td></td>
<td><code>obj</code> An EmbeddedLO (object) or</td>
</tr>
<tr>
<td></td>
<td>A ConverterEmbeddedLO (object)</td>
</tr>
<tr>
<td></td>
<td><code>value</code> (Long) Maximum number of tuning iterations.</td>
</tr>
<tr>
<td>Return Type</td>
<td><strong>(Long)</strong></td>
</tr>
<tr>
<td>Default</td>
<td>5</td>
</tr>
<tr>
<td>Examples</td>
<td><code>embedLO.MaxPreciseTuningIterations = 3 'write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = embedLO.MaxPreciseTuningIterations 'read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_MaxPreciseTuningIterations long* iter);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_MaxPreciseTuningIterations long iterator);</td>
</tr>
<tr>
<td>Interface</td>
<td>IEmbededLO</td>
</tr>
</tbody>
</table>
# MaxProduct Property

Sets and returns the maximum intermod product frequencies to be calibrated.

**Description**

**VB Syntax**

```
imd.MaxProduct = value
```

**Variable** *(Type) - Description*

- `imd` A **SweptIMDCal** *(object)*
- `value` Maximum IM products to calibrate. Choose from:
  - 2 - second order products
  - 3 - third order products
  - 5 - fifth order products
  - 7 - seventh order products
  - 9 - ninth order products

**Return Type** Long Integer

**Default** 3

**Examples**

```
imd.MaxProduct = 7 'Write
mprod = imd.MaxProduct 'Read
```

**C++ Syntax**

```
HRESULT get_MaxProduct(long * Val)
HRESULT put_MaxProduct(long newVal)
```

**Interface** ISweptIMD
### Mean Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the mean value of the measurement. To retrieve all 3 statistics value at the same time, use <code>meas.GetTraceStatistics</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>average = meas.Mean</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>average</code> (single)</td>
<td>Variable to store mean value</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Single</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not applicable</td>
</tr>
</tbody>
</table>
| **Examples** | Dim average as Single  
`average = meas.Mean 'Read` |
| **C++ Syntax** | HRESULT get_Mean(float* mean) |
| **Interface** | IMeasurement |
**Measurement Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the measurement handle of the trace object. Learn the difference between a Trace and a Measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>myMeas = trace.Measurement</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>myMeas</code></td>
<td>(Object) Variable to store the returned Measurement object.</td>
</tr>
<tr>
<td><code>trace</code></td>
<td>A Trace (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Object</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>myMeas = myTrace.Measurement</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_Measurement(IMeasurement** ppMeas);</td>
</tr>
<tr>
<td>Interface</td>
<td>ITrace2</td>
</tr>
</tbody>
</table>
### MeasurementClassProperties Property

**Description**  Provides access to the MeasurementClassProperties Object.

**VB Syntax**  

```
handle = cap.MeasurementClassProperties (measClass)
```

**Variable**  
- **Type** - Description
- **handle** (Variant) - Variable to store the returned object.
- **cap** A Capabilities (object)
- **measClass** (String) - Name of the measurement class to be accessed.

Use **AvailableMeasurementClasses Property** to return a list of measurement classes installed on the VNA.

**Return Type**  Object

**Default**  Not Applicable

**Examples**  

'Access the MeasurementClassProperties Object

```vbnet
Set app = CreateObject("AgilentPNA835x.Application")
Set cap = app.Capabilities
Set measProps = cap.MeasurementClassProperties("SweptIMD")
```

**C++ Syntax**  

```
HRESULT put_MeasurementClassProperties(IMeasurementClassProperties *handle);
```

**Interface**  ICapabilities8
## MeasurementClass Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the measurement class name from the channel. Use <code>CreateCustomMeasurementEx</code> to create a measurement from a class other than standard S-Parameters.</th>
</tr>
</thead>
</table>

**VB Syntax**  
`class = chan.MeasurementClass`

**Variable**  
- **Type** - Description
  - `class` (string) - Variable to store the returned measurement class name.
  - `chan` Channel (object)

**Return Type**  
String

**Default**  
Not Applicable

**Examples**  
`class = chan.MeasurementClass 'Read For a standard S-Parameter channel, returns... "Standard"`

**C++ Syntax**  
`HRESULT get_MeasurementClass();`

**Interface**  
IChannel15
### Medium Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Returns the media type of the calibration standard.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>calsdt.Medium = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>calsdt</code></td>
<td>A CalStandard <strong>(object)</strong>. Use <code>calKit.GetCalStandard</code> to get a handle to the standard.</td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(enum NACalStandardMedium)</strong> - Medium of the transmission line of the standard. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <strong>naCoax</strong> - Coaxial Cable</td>
</tr>
<tr>
<td></td>
<td>1 - <strong>naWaveGuide</strong></td>
</tr>
<tr>
<td>Return Type</td>
<td>Long Integer</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>calsdt.Medium = naCoax</code> <strong>Write</strong></td>
</tr>
<tr>
<td></td>
<td><code>stdMedium = calsdt.Medium</code> <strong>Read</strong></td>
</tr>
</tbody>
</table>

**Examples**

```vbnet
    calsdt.Medium = naCoax 'Write
    stdMedium = calsdt.Medium 'Read
```

**C++ Syntax**

```c++
    HRESULT get_Medium(tagNACalStandardMedium *pVal)
    HRESULT put_Medium(tagNACalStandardMedium newVal)
```

**Interface**

ICalStandard
## Memory Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return the memory trace color for nth trace in a window.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>trace(n).Memory = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>trace(n)</code></td>
<td>One of the 8 <code>ComTraceColors</code> objects</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long Integer) - RGB color of the Memory pen.</td>
</tr>
</tbody>
</table>

Convert the three RGB colors to an integer as follows:

\[
\text{RGB} = R + (G \times 2^8) + (B \times 2^{16})
\]

To find the three RGB values from the Display Colors dialog, click Change Color, then Define Custom Color.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Long</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Varies for each trace.</td>
</tr>
</tbody>
</table>

| Examples | R = 10  
G = 10  
B = 10  
RGB = 10+(10\times2^8)+(10\times2^{16})  
`trace1.Memory = RGB` 'Write  
`color = trace1.Memory` 'Read |
|----------|---------------------------------|

| C++ Syntax | HRESULT get_Memory(long* pVal);  
HRESULT put_Memory(long newVal); |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>ITraceColors</td>
</tr>
</tbody>
</table>
MemoryMarkers Property

Description
Set and return the color of memory trace markers for nth trace in a window.

VB Syntax
trace(n).MemoryMarkers = value

Variable
(Type) - Description
trace(n) One of the 8 ComTraceColors objects
value (Long Integer) - RGB color of the MemoryMarkers pen.

Convert the three RGB colors to an integer as follows:

RGB = R + (G*2^8) + (B*2^16)

To find the three RGB values from the Display Colors dialog, click Change Color, then Define Custom Color.

Return Type
Long

Default
Varies for each trace.

Examples
R = 10
G = 10
B = 10
RGB = 10+(10*2^8)+(10*2^16)
trace1.MemoryMarkers = RGB  'Write
color = trace1.MemoryMarkers  'Read

C++ Syntax
HRESULT get_MemoryMarkers(long* pVal);

HRESULT put_MemoryMarkers(long newVal);

Interface
ITraceColors
## MemShareEnabled Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables/disables exporting data to shared memory, which is the fastest way to transfer data between applications.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>sa.dft.MemShareEnabled = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>sa.dft</code></td>
<td>A SpectrumAnalyzerDFT <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Boolean)</em> Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>0 - OFF</strong> - Disable memory sharing.</td>
</tr>
<tr>
<td></td>
<td><strong>1 - ON</strong> - Enable memory sharing.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>OFF</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>sa.dft.MemShareEnabled = ON</code> <em>(Write)</em></td>
</tr>
<tr>
<td><code>value = sa.dft.MemShareEnabled</code> <em>(Read)</em></td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_MemShareEnabled(VARIANT_BOOL* enable)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_MemShareEnabled(VARIANT_BOOL enable)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISpectrumAnalyzerDFT</td>
</tr>
</tbody>
</table>
**MemShareName Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Assigns a specified name to the Microsoft Windows shared data mechanism when <em>MemShareEnabled</em> is enabled.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><em>sa.dft.MemShareName</em> = <em>MemFileName</em></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><em>sa.dft</em></td>
<td>A <em>SpectrumAnalyzerDFT (object)</em></td>
</tr>
<tr>
<td><em>MemFileName</em></td>
<td><em>(String)</em> Memory file name.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><em>sa.dft.MemShareName = &quot;Mem_Share&quot;</em> 'Write</td>
</tr>
<tr>
<td></td>
<td><em>name = sa.dft.MemShareName</em> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_MemShareName(BSTR* MemFileName);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_MemShareName(BSTR MemFileName);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISpectrumAnalyzerDFT</td>
</tr>
</tbody>
</table>
### MessageText Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns text for the specified eventID</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>app.MessageText, eventID, message</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
<tr>
<td>eventID</td>
<td>(enum naEventID) Choose from the list in Working with the Analyzer's Events</td>
</tr>
<tr>
<td>message</td>
<td>(string) - variable to store the returned message</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>RFNA.MessageText naEventID_ARRANGE_WINDOW_EXCEED_CAPACITY, message</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_MessageText( tagNAEventID msgID, BSTR* message)</td>
</tr>
<tr>
<td>Interface</td>
<td>IApplication</td>
</tr>
</tbody>
</table>
## MinimumFrequency Property

**Description**
Sets and Returns the minimum frequency for the calibration standard.

**VB Syntax**
```
calstd.MinimumFrequency = value
```

**Variable**
(Type) - Description
- `value` (double) - Minimum frequency in Hertz.

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```
calstd.MinimumFrequency = 300e3  'Write
minFrequency = calstd.MinimumFrequency  'Read
```

**C++ Syntax**
```
HRESULT get_MinimumFrequency(double *pVal)
HRESULT put_MinimumFrequency(double newVal)
```

**Interface**
ICalStandard
### MinimumFrequency Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the minimum frequency of the remote VNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>( value = cap.\text{MinimumFrequency} )</td>
</tr>
<tr>
<td>Variable</td>
<td>( \text{(Type)} - \text{Description} )</td>
</tr>
<tr>
<td>( value )</td>
<td>( \text{(Double)} - \text{Variable to store the returned minimum frequency of the VNA.} )</td>
</tr>
<tr>
<td>( cap )</td>
<td>A \text{Capabilities (object)}</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>( value = cap.\text{MinimumFrequency} 'Read )</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT \text{get_MinimumFrequency(double *pVal)}</td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities</td>
</tr>
</tbody>
</table>
## MinimumFrequency Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the minimum usable frequency specified for the power sensor.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pwrSensor.MinimumFrequency = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>pwrSensor</code></td>
<td>A <code>PowerSensor</code> (Object) or</td>
</tr>
<tr>
<td><code>value</code></td>
<td>A <code>PowerSensorAsReceiver</code> (Object)</td>
</tr>
<tr>
<td><strong>(Type)</strong></td>
<td>(double) - Frequency in Hertz.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Device dependent</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pwrSensor.MinimumFrequency = 300e3 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>MinFreq = pwrSensor).MinimumFrequency 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT put_MinimumFrequency(double newVal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_MinimumFrequency(double *pVal);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPowerSensor</td>
</tr>
<tr>
<td></td>
<td>IPowerSensorAsReceiver</td>
</tr>
</tbody>
</table>
## MinimumIFFrequency Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the minimum allowed value for the IFFrequency Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>value = IfConfig.MinimumIFFrequency</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(double) Variable to store the returned minimum allowed frequency that can be applied to the IFFrequency Property.</td>
</tr>
<tr>
<td>IfConfig</td>
<td>IFConfiguration (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>val = App.ActiveChannel.IFConfiguration.MinimumIFFrequency</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_MinimumIFFrequency( double * pMinFreq);</td>
</tr>
<tr>
<td>Interface</td>
<td>IIFConfiguration3</td>
</tr>
</tbody>
</table>
### MinimumNumberOfPoints Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the minimum possible number of data points for a data trace.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = cap.MinimumNumberOfPoints</code></td>
</tr>
<tr>
<td>Variable</td>
<td><code>value</code> (Long) - Variable to store the returned minimum value for number of points.</td>
</tr>
<tr>
<td>cap</td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = cap.MinimumNumberOfPoints</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_MinimumNumberOfPoints(double * minimumNumberOfPoints);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities</td>
</tr>
</tbody>
</table>
# MinimumReceiverStepAttenuator Property

**Description**
Returns a value indicating the minimum amount of receiver attenuation.

**VB Syntax**
```vbnet
value = cap.MinimumReceiverStepAttenuator(n)
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td>(Double)</td>
<td>Variable to store the returned minimum value of receiver attenuation.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities (object)</td>
<td></td>
</tr>
<tr>
<td><code>n</code></td>
<td>(Long)</td>
<td>port number to query for minimum value of receiver attenuation</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```vbnet
value = cap.MinimumReceiverStepAttenuator'Read
```

**C++ Syntax**
```csharp
HRESULT get_MinimumReceiverStepAttenuator(long portNumber, double *attenuation);
```

**Interface**
ICapabilities
MinimumSourceALCPower Property

**Description**
Returns a value indicating the minimum amount of source ALC power with 0 dB source attenuation.

To calculate the minimum amount of source ALC power with a different level of attenuation, subtract the amount source attenuation from this value.

**VB Syntax**
```vbnet
value = cap.MinimumSourceALCPower(n)
```

**Variable**
- `value` *(Double)* - Variable to store the returned minimum value of source ALC power.
- `cap` *A Capabilities (object)*
- `n` *(Long)* - source number to query for the minimum value of source ALC power.

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```vbnet
value = cap.MinimumSourceALCPower
```

**C++ Syntax**
```c++
HRESULT get_MinimumSourceALCPower(long sourceNum, double * power);
```

**Interface**
ICapabilities
# MinOutput Property

Sets and returns the "Define Min As" value for an external DC Source.

**VB Syntax**

\[ extDC.MinOutput = value \]

**Variable**

- **Type** - Description
  - \( extDC \) A `ExternalDCDevice` (object)
  - \( value \) (Double) Minimum value for the external DC Source.

**Return Type** Double

**Default** -10 V

**Examples**

- `extDC.MinOutput = -10` 'Write
- `value = extDC.MinOutput` 'Read

**C++ Syntax**

```c++
HRESULT get_MinOutput( double* value);

HRESULT put_MinOutput( double value);
```

**Interface** IExternalDCDevice3
### MinOutputState Property

Sets and returns the "Define Min As" ON/OFF state for an external DC Source.

**VB Syntax**

```vbnet
extDC.MinOutputState = value
```

**Variable**

- **extDC**: An `ExternalDCDevice (object)`
- **value**: (Boolean) "Define Min As" ON/OFF state. Choose from:
  - **True**: Turn "Define Min As" ON
  - **False**: Turn "Define Min As" OFF

**Return Type**: Boolean

**Default**: True

**Examples**

```vbnet
extDC.MinOutputState = True 'Write
value = extDC.MinOutputState 'Read
```

**C++ Syntax**

```cpp
HRESULT get_MinOutputState(VARIANT_BOOL *value)
HRESULT put_MinOutputState(VARIANT_BOOL value)
```

**Interface**: IExternalDCDevice3
# MixerCharacterizationFile Property

**Description**  
Set the filename of the S2P file used to characterize the calibration mixer

**VB Syntax**  
```vbnet
smc.MixerCharacterizationFile = value
```

**Variable (Type) - Description**

- `smc`  
  An SMCType (object)

- `value`  
  (String) Full path, file name, and extension of the mixer characterization file.

**Return Type**  
String

**Default**  
Not applicable

**Example**  
```vbnet
SMC.MixerCharacterizationFile = "C:\Program Files(x86)\Keysight\Network Analyzer\Documents\default.S2P"
```

**C++ Syntax**  
```cpp
HRESULT put_MixerCharacterizationFile(BSTR Value);
```

**Interface**  
SMCType5
## Mode Property

**Description**
Sets the type of transform.

### VB Syntax
\[
\text{trans}.\text{Mode} = \text{value}
\]

### Variable
- **trans**: A Transform (object)
- **value**: (enum NATransformMode) - Choose from:
  - 0 - naTransformBandpassImpulse
  - 1 - naTransformLowpassImpulse
  - 2 - naTransformLowpassStep

### Return Type
NATransformMode

### Default
0 - naTransformBandpassImpulse

### Examples
- \[
\text{trans}.\text{Mode} = \text{naTransformLowpassStep} \quad \text{'Write}
\]
- \[
\text{transmode} = \text{trans}.\text{Mode} \quad \text{'Read}
\]

### C++ Syntax
- HRESULT get_Mode(tagNATransformMode *pVal)
- HRESULT put_Mode(tagNATransformMode newVal)

### Interface
ITransform
## Mode Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the stimulus mode for balanced measurements.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>balStim.Mode = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>balStim</code></td>
<td>A <code>BalancedStimulus</code> <em>(object)</em></td>
</tr>
</tbody>
</table>
| `value`     | *(Enum NABALSTIMulus) - Stimulus Mode. True modes are applicable only with Opt S93460A/B - iTMSA. When a True-Mode is selected, the Balanced port powers are automatically uncoupled. Choose from:*
|             | 0 - `naSEStim`: Single-Ended stimulus                       |
|             | 1 - `naTMStim`: True-Mode stimulus                          |
|             | 2 - `naFTMStim`: Forward only True-Mode stimulus            |
|             | 3 - `naRTMStim`: Reversed only True-Mode stimulus           |
| Return Type | `Enum`                                                      |
| Default     | 0 - `naSEStim`: Single-Ended stimulus                       |
| Examples    | `balStim.Mode = naTMStim 'Write`                           |
|             | `variable = balStim.Mode 'Read`                             |
| C++ Syntax  | HRESULT get_Mode(tagNABALSTIMulus *eVal );                  |
|             | HRESULT put_Mode(tagNABALSTIMulus eVal );                   |
| Interface   | IBalancedStimulus                                          |
**ModelNumber Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the model number of the VNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>(\text{value} = \text{cap.ModelNumber})</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(String) - Variable to store the returned value.</td>
</tr>
<tr>
<td>cap</td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>(\text{value} = \text{cap.ModelNumber} )</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_ModelNumber(BSTR * model);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities11</td>
</tr>
</tbody>
</table>
## ModulatorDelay Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the time lag between the pulse drive signal and the actual RF output.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>pulse.ModulatorDelay = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>Type</strong> - Description</td>
</tr>
<tr>
<td><code>pulse</code></td>
<td>A PulseGenerator (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) - Modulator delay value.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>40 ns</td>
</tr>
<tr>
<td>Examples</td>
<td><code>pulse.ModulatorDelay = 40ns 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = pulse.ModulatorDelay 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_ModulatorDelay(double* val);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_ModulatorDelay(double val);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IPulseGenerator5</td>
</tr>
</tbody>
</table>
Multiplier Property

Description
Sets and returns the multiplier value to be used when coupling this range to the primary range. Learn more about multiplier value.

This setting is valid only if this range is coupled to the primary range.

VB Syntax

```
FOMRange.Multiplier = value
```

Variable (Type) - Description

| object | An FOMRange (object) |
| value | (Double) - Multiplier value.- (Unitless) |

Return Type
Double

Default
1

Examples

```
fomRange.Multiplier = 2 'Write
Mult = fomRange.Multiplier 'Read
```

C++ Syntax

```
HRESULT get_Multiplier(double *pVal)
HRESULT put_Multiplier(double pVal)
```

Interface
IFOMRange
### MultiToneDataDisplay Property

**Description**  
Sets and returns the data display mode.

**VB Syntax**  
```
sa.coherence.MultiToneDataDisplay = value
```

**Variable**  
*sa.coherence A SpectrumAnalyzerCoherence (object)*

*value (enum NASACoherentDataDisplay)* - Choose from:

- **0** naSAShowAll: Legacy SA mode - all frequency points are displayed.
- **1** naSAZeroNonTones: All the frequencies that are not on the multi-tone coherence grid are set to 200 dBm before being displayed.

**Return Type**  
Enum as NASACoherentDataDisplay

**Default**  
naSAShowAll

**Examples**  
```
sa.coherence.MultiToneDataDisplay = naSAShowAll 'Write
DisplayMode = sa.coherence.MultiToneDataDisplay 'Read
```

**C++ Syntax**  
```
HRESULT get_MultiToneDataDisplay(tagNASACoherentDataDisplay* value)
HRESULT put_MultiToneDataDisplay(tagNASACoherentDataDisplay value)
```

**Interface**  
ICoherenceSA
MultiToneEnable Property

Description
Enables/disables multitone image rejection. When enabled, the window type is set to No Window and the list of RBW possible values is recomputed according to the multitone spacing. When disabled, the window type is set back to what it was before enabling and the RBW list is also set to the previous setting.

VB Syntax
```
sa.coherence.MultiToneEnable = value
```

Variable (Type) - Description
A SpectrumAnalyzerCoherence (object)
```
value (Boolean) Choose from:
```
- 0 - OFF - Multitone image rejection disabled.
- 1 - ON - Multitone image rejection enabled.

Return Type
Boolean

Default
OFF

Examples
```
sa.coherence.MultiToneEnable = ON 'Write
value = sa.coherence.MultiToneEnable 'Read
```

C++ Syntax
```
HRESULT get_MultiToneEnable(VARIANT_BOOL* enable)
HRESULT put_MultiToneEnable(VARIANT_BOOL enable)
```

Interface
ICoherenceSA
## MultiToneHarmonicRejection Property

**Description**  
Sets and returns the number of test signal harmonics you want to be protected against. This adds constraints to the list of LOs used to cover the span.

**VB Syntax**  
`sa.coherence.MultiToneHarmonicRejection = value`

**Variable**  
**Type** - Description

- `sa.coherence`  
  A `SpectrumAnalyzerCoherence` (object)

- `value`  
  (`Long`) Number of test signal harmonics to be protected. The more this number is increased, the more constraints are added on the span LOs setting.

**Return Type**  
`Long`

**Default**  
0

**Examples**  
```
   sa.coherence.MultiToneHarmonicRejection = 0     'Write
   value = sa.coherence.MultiToneHarmonicRejection  'Read
```

**C++ Syntax**  
```cpp
HRESULT put_MultiToneHarmonicRejection(long value);

HRESULT get_MultiToneHarmonicRejection(long* value);
```

**Interface**  
ICoherenceSA
### MultiToneNyquistProtection Property

**Description**
Sets and returns the Nyquist protection level. Avoids Nyquist images of the IF higher order signal to fall back on multitone frequencies. This setting can only be set > 1 if the tone spacing of the multitone is not an integer divider of 100 MHz.

**VB Syntax**
```
sa.coherence.MultiToneNyquistProtection = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sa.coherence</code></td>
<td>A SpectrumAnalyzerCoherence (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) Nyquist protection level.</td>
</tr>
</tbody>
</table>

**Return Type**
Long

**Default**
0

**Examples**
```
sa.coherence.MultiToneNyquistProtection = 2  'Write
value = sa.coherence.MultiToneNyquistProtection  'Read
```

**C++ Syntax**
```
HRESULT put_MultiToneNyquistProtection(long value);
HRESULT get_MultiToneNyquistProtection(long* value);
```

**Interface**
ICoherenceSA2
## MultiTonePeriod Property

**Description**
Sets and returns the test signal repetition rate (in seconds). This value is 1/MultiToneImageRejectSpacing.

**VB Syntax**
```vb
sa.coherence.MultiTonePeriod = value
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sa.coherence</td>
<td>A SpectrumAnalyzerCoherence (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Double) Test signal repetition rate (in seconds)</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
1E6

**Examples**
```vb
sa.coherence.MultiTonePeriod = 1E6 'Write
value = sa.coherence.MultiTonePeriod 'Read
```

**C++ Syntax**
```cpp
HRESULT put_MultiTonePeriod(double freqval);
HRESULT get_MultiTonePeriod(double* freqval);
```

**Interface**
ICoherenceSA
## MultiToneReference Property

**Description**  
Sets and returns the multitone image rejection offset frequency. If the multitone grid does not start from 0 Hz, this command is used to set its offset. To make this more convenient, this command accepts as well the frequency of any tone of the multitone grid (Hz).

**VB Syntax**  
```
sa.coherence.MultiToneReference = value
```

**Variable**  
(sa.coherence) - A `SpectrumAnalyzerCoherence` (object)

value - (Double) Offset frequency (in Hz)

**Return Type**  
Double

**Default**  
0

**Examples**  
```vbnet
sa.coherence.MultiToneReference = 0 'Write
value = sa.coherence.MultiToneReference 'Read
```

**C++ Syntax**  
```c++
HRESULT put_MultiToneReference(double freqval);
HRESULT get_MultiToneReference(double* freqval);
```

**Interface**  
ICoherenceSA
## MultitoneSettingsValid Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the current multitone settings and determine if they are valid or not.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = sa.coherence.MultitoneSettingsValid</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>sa.coherence</code></td>
<td>A <code>SpectrumAnalyzerCoherence</code> <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long)</em> Variable to store the returned value. A &quot;1&quot; is valid and a &quot;0&quot; is invalid.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>value = sa.coherence.MultitoneSettingsValid  'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_MultitoneSettingsValid(VARIANT_BOOL* value);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICoherenceSA2</td>
</tr>
</tbody>
</table>
## MultiToneSpacing Property

**Description**
Sets and returns the tone spacing of the multitone signal (in Hz). This value is \(1/\text{MultiToneImageRejectPeriod}\).

**VB Syntax**

```vbnet
sa.coherence.MultiToneSpacing = value
```

**Variable**

- **sa.coherence**
  - A `SpectrumAnalyzerCoherence` object
- **value**
  - `Double` Frequency spacing of multitone signal (in Hz)

**Return Type**
Double

**Default**
1000000

**Examples**

```vbnet
sa.coherence.MultiToneSpacing = 1E6 'Write
value = sa.coherence.MultiToneSpacing 'Read
```

**C++ Syntax**

```cpp
HRESULT put_MultiToneSpacing(double freqval);

HRESULT get_MultiToneSpacing(double* freqval);
```

**Interface**
ICoherenceSA
## Name Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the Name of the Cal Set.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>CalSet.Name = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>CalSet</code></td>
<td><em>(object)</em> - A Cal Set object</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(string)</em> - Name of the Cal Set.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

### Examples

```vbnet
dim pna
set
pna=createobject("AgilentPNA835x.Application")

dim calsets
set calsets=pna.getcalmanager.calsets

dim c
for each c in calsets
wscript.echo c.name
'Changes the name of CalSet_1
if c.name="CalSet_1" then c.name="New"
next```

### C++ Syntax

```cpp
HRESULT get_Name(BSTR *name)

HRESULT put_Name(BSTR name);
```

### Interface

ICalSet4
# Name (CalKit) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Returns a name for the selected calibration kit.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>calKit.Name = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong> (Type) - Description</td>
<td></td>
</tr>
<tr>
<td><code>calKit</code></td>
<td>A CalKit (object).</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(string) - Calibration Kit name. Any string name, can include numerics, period, and spaces; any length (although the dialog box display is limited to about 30 characters).</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td></td>
</tr>
<tr>
<td><code>calKit.Name = &quot;MyCalKit&quot;</code></td>
<td>'Write</td>
</tr>
<tr>
<td>KitName = calKit.Name</td>
<td>'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Name(BSTR *pVal)</td>
</tr>
<tr>
<td>HRESULT put_Name(BSTR newVal)</td>
<td></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalKit</td>
</tr>
</tbody>
</table>
Name (PathConfig) Property

Description
Returns the name of the current configuration only if NO individual element settings had been changed since selecting or storing a configuration. When element settings change, the path configuration name is cleared.

VB Syntax

\[ name = pathConfig.Name \]

Variable

(name) Variable to store the returned configuration name.

pathConfig
A PathConfiguration (object)

Return Type
String

Default
'default' - name of the default factory configuration

Examples

\[ name=pathConf.Name \]

C++ Syntax

HRESULT get_Name(BSTR* ppName)

Interface
IPathConfiguration
### Name Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the name of the current element object</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>$name = pathElement.Name$</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>name</td>
<td>(String) Variable to store the returned element name.</td>
</tr>
<tr>
<td>pathElement</td>
<td>A PathElement (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>name = pathElement.Name</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Name(BSTR* ppName)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPathElement</td>
</tr>
</tbody>
</table>
### Name (ExternalDevice) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the name of the External Device.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>extDev.Name = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong> (Type) - Description</td>
<td><code>extDev</code> A ExternalDevice Object (object). <code>value</code> (string) - External Device name. Any string name limited to alpha-numeric characters.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Device&lt;n&gt;</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>extDev.Name = &quot;MySource&quot;</code>  <code>extDevName = extDev.Name</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_Name(BSTR *pVal)</code>  <code>HRESULT put_Name(BSTR newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IExternalDevice</td>
</tr>
</tbody>
</table>

See example program to configure PMAR device  
See example program to configure External Source
## Name Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the name of this FOM range object.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = FOMRange.Name</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(string) - Variable to store the returned range name.</em></td>
</tr>
<tr>
<td><strong>FOMRange</strong></td>
<td>An FOMRange <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>Rname = fomRange.Name 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Name(BSTR *pRName)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFOMRange</td>
</tr>
</tbody>
</table>
# Name (Measurement) Property

**Description**: Sets (or returns) the Name of the measurement. Measurement names must be unique among the set of measurements. Measurement names cannot be an empty string.

**Note**: This is the same name as trace.Name; when one changes, the other changes.

**VB Syntax**
```
meas.Name = value
```

**Variable (Type) - Description**
- `meas`: A Measurement (object)
- `value`: (string) - A user defined name of the measurement

**Return Type**: String

**Default**: "CH1_S11_1" - name of the default measurement

**Examples**
```
meas.Name = "Filter BPass"  'Write
MName = meas.Name  'Read
```

**C++ Syntax**
```
HRESULT get_Name(BSTR *pVal)
HRESULT put_Name(BSTR newVal)
```

**Interface**: IMeasurement
# Name Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the name of the Power Sensor (object) to be used as part of a Guided Power Calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>pSensor.Name = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>pSensor</code></td>
<td>A GuidedCalibrationPowerSensor (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(string) - Name of the power sensor. The power sensor must be already configured as a PMAR device using this name. Learn how to remotely configure a PMAR device.</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
<th><code>Sensor.Name = &quot;26GHzSensor&quot;</code> 'write'</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>value = pSensor.Name</code> Read</td>
</tr>
</tbody>
</table>

- See example program

<table>
<thead>
<tr>
<th>C++ Syntax</th>
<th>HRESULT get_Name(BSTR *psensorName)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HRESULT put_Name(BSTR pSensorName)</td>
</tr>
</tbody>
</table>

| Interface    | IGuidedCalibrationPowerSensor                                                                         |
### Name (trace) Property

**Description**
Sets or returns the name of the Trace. Use the trace name to identify the trace and refer to the trace in the collection.

**Note:** This is the same name as meas.Name; when one changes, the other changes.

**VB Syntax**

```vbnet
trac.Name = value
```

**Variable**

- **(Type)** - Description
- **trac** - A Trace **(object)**
- **value** - *(String)* Trace name

**Return Type**

String

**Default**

"CH1_S11_1" - name of the default measurement

**Examples**

```
trace.Name = "myTrace" 'Write
traceName = Name.Trace 'Read
```

**C++ Syntax**

```cpp
HRESULT put_Name(BSTR name)
HRESULT get_Name(BSTR *name)
```

**Interface**

ITrace
<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the name of the Cable.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td></td>
</tr>
<tr>
<td><code>value = oCable.Name</code></td>
<td></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(String) Cable name.</td>
</tr>
<tr>
<td><code>oCable</code></td>
<td>A <strong>Cable</strong> Object</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = oCable.Name</code></td>
</tr>
<tr>
<td></td>
<td>See example program</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Name([out,retval] BSTR* pCableName);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IUncertaintyCable</td>
</tr>
</tbody>
</table>
NetworkFilename Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>object.NetworkFilename(n) = filename</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>object</code></td>
<td>SMCTYPE (object) or VMCType (object)</td>
</tr>
<tr>
<td><code>n</code></td>
<td>(Integer) Apply network to input or output of mixer. Choose from:</td>
</tr>
<tr>
<td></td>
<td>1 - Input of mixer</td>
</tr>
<tr>
<td></td>
<td>2 - Output of mixer</td>
</tr>
<tr>
<td><code>filename</code></td>
<td>(String) Filename of the S2P used for embedding or de-embedding. Use the full path name, file name, and .S2P suffix, enclosed in quotes.</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>VMC.Filename(2) = &quot;D:\WaveguideAdapt.S2P&quot;</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT put_NetworkFilename(short networkNum, BSTR filename);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_NetworkFilename(short networkNum, BSTR *filename);</td>
</tr>
<tr>
<td>Interface</td>
<td>SMCTYPE2</td>
</tr>
<tr>
<td></td>
<td>VMCType2</td>
</tr>
</tbody>
</table>

1976
**NetworkMode Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Allows you to embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>object.NetworkMode(n) = value</td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td>object</td>
<td>SMCTYPE (object) or VMCTYPE (object)</td>
</tr>
<tr>
<td>n</td>
<td>(Integer) Apply network to input or output of mixer. Choose from:</td>
</tr>
<tr>
<td>1</td>
<td>1 - Input of mixer</td>
</tr>
<tr>
<td>2</td>
<td>2 - Output of mixer</td>
</tr>
<tr>
<td>value</td>
<td>(ENum as ENetworkMode) Choose from:</td>
</tr>
<tr>
<td>NO_NETWORK</td>
<td>Do nothing with effects of S2P file</td>
</tr>
<tr>
<td>EMBED_NETWORK</td>
<td>Add effects of S2P file from the measurement results.</td>
</tr>
<tr>
<td>DEEMBED_NETWORK</td>
<td>Remove effects of S2P file from the measurement results.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>NO_NETWORK</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>VMC.NetworkMode = EMBED_NETWORK</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT put_NetworkMode(short networkNum, enum ENetworkMode networkMode);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_NetworkMode(short networkNum, enum ENetworkMode *networkMode);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>SMCTYPE2</td>
</tr>
<tr>
<td></td>
<td>VMCTYPE2</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Read the port mapping of “in A” port for a 4-port SNP file to be embedded. Use NetworkPortMap Method to set the port map for all four ports.</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = fixture.NetworkPortMapA(network)</code></td>
</tr>
</tbody>
</table>
| **Variable**    | **(Type)** - **Description**  
|                 | value (Long) | Variable to store the returned value. |
|                 | fixture (object) | A Fixturing (object) |
|                 | network (Integer) | Network position. Choose from 1 or 2. |
| **Default**     | 1 |
| **Examples**    | `value = fixture.NetworkPortMapA(1)` |
| **C++ Syntax**  | HRESULT get_NetworkPortMapA(short* Net); |
| **Interface**   | IFixturing6 |
### NetworkPortMapB Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the port mapping of “in B” port for a 4-port SNP file to be embedded.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use</td>
<td><strong>NetworkPortMap</strong> Method to set the port map for all four ports.</td>
</tr>
</tbody>
</table>

#### VB Syntax

```vbnet
type = fixture.NetworkPortMapB(network)
```

#### Variable

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(Long) Variable to store the returned value.</td>
</tr>
<tr>
<td>fixture</td>
<td>A Fixturing (object)</td>
</tr>
<tr>
<td>network</td>
<td>(Integer) Network position. Choose from 1 or 2.</td>
</tr>
</tbody>
</table>

#### Default

2

#### Examples

```vbnet
type = fixture.NetworkPortMapB(1)
```

#### C++ Syntax

```
HRESULT get_NetworkPortMapB(short* Net);
```

#### Interface

IFixturing6
### NetworkPortMapC Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the port mapping of “out A” port for a 4-port SNP file to be embedded. Use <code>NetworkPortMap</code> Method to set the port map for all four ports.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) Variable to store the returned value.</td>
</tr>
<tr>
<td><code>fixture</code></td>
<td>A <code>Fixturing</code> (object)</td>
</tr>
<tr>
<td><code>network</code></td>
<td>(Integer) Network position. Choose from 1 or 2.</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>3</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = fixture.NetworkPortMapC(1)</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_NetworkPortMapC(short* Net);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IFixturing6</code></td>
</tr>
</tbody>
</table>
NetworkPortMapD Property

Description
Read the port mapping of “out B” port for a 4-port SNP file to be embedded.

Use NetworkPortMap Method to set the port map for all four ports.

VB Syntax
value = fixture.NetworkPortMapD(network)

Variable (Type) - Description
value (Long) Variable to store the returned value.
fixture A Fixturing (object)
network (Integer) Network position. Choose from 1 or 2.
Default 4

Examples
value = fixture.NetworkPortMapD(1)

C++ Syntax
HRESULT get_NetworkPortMapD(short* Net);

Interface IFixturing6
# NoiseAverageFactor Property

**Description**
Sets and reads the averaging of the noise receiver.

**VB Syntax**

```vbnet
noise.NoiseAverageFactor = value
```

**Variable**

- **Type** - Description
- **noise** - A `NoiseFigure` *(object)*
- **value** - *(long integer)* - Averaging value. Choose a number between 1 and 99.

**Return Type**
Long Integer

**Default**
1

**Examples**

```vbnet
noise.NoiseAverageFactor = 10 'Write
AvgNoise = noise.NoiseAverageFactor 'Read
```

**C++ Syntax**

```csharp
HRESULT get_NoiseAverageFactor(long* pVal)

HRESULT put_NoiseAverageFactor(long newVal)
```

**Interface**
INOiseFigure
# NoiseAverageState Property

## Description
Turns Noise Averaging ON and OFF.

## VB Syntax

```vbnet
noise.NoiseAverageState = value
```

## Variable

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>noise</code></td>
<td>A NoiseFigure (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(boolean) - Averaging state.</td>
</tr>
</tbody>
</table>

- **False** - Turns Noise Averaging OFF
- **True** - Turns Noise Averaging ON

## Return Type
Boolean

## Default
False

## Examples

```vbnet
noise.NoiseAverageState = OFF 'Write
NoiseAvgState = noise.NoiseAverageState 'Read
```

## C++ Syntax

```cpp
HRESULT get_NoiseAverageState(VARIANT_BOOL* on);
HRESULT put_NoiseAverageState(VARIANT_BOOL on);
```

## Interface
INoiseFigure
## NoiseBandwidth Property

**Description**  
Set the bandwidth of the noise receiver.

**VB Syntax**  
`noise.NoiseBandwidth = value`

**Variable**  
- **Type**
  - `noise`  
    - A `NoiseFigure` (object)
  - `value`  
    - (double) Bandwidth value.

For `NoiseReceiver = naNoiseReceiver` (Opt 029) choose from: 800 KHz, 2 MHz, 4 MHz, 8 MHz, or 24 MHz or the numerical equivalent, such as 8e6 and so forth. **NOTE:** The Receiver Characterization Method = "Power Meter" is NOT allowed when the Noise Bandwidth is 8 MHz or 24 MHz.

For `NoiseReceiver = naStandardReceiver` (Opt 028) choose from: 720 kHz or 1.2 MHz

If the value does not match one of these, it is rounded up to the next valid bandwidth value.

**Return Type**  
Double

**Default**  
4 MHz for `naNoiseReceiver`

1.2 MHz for `naStandardReceiver`

**Examples**  
```
noise.NoiseBandwidth = 2E6 'Write
NoiseBW = noise.NoiseBandwidth 'Read
```

**C++ Syntax**  
```c++
HRESULT get_NoiseBandwidth(double *pVal);
HRESULT put_NoiseBandwidth(double newVal);
```

**Interface**  
INoiseFigure
### NoiseFigureBW Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Sets or returns the noise figure bandwidth.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>seg.NoiseFigureBW = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>seg - A Segment (object)</td>
</tr>
<tr>
<td><strong>value</strong></td>
<td>(Double) Noise figure bandwidth.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>seg.NoiseFigureBW = 1000 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = seg.NoiseFigureBW 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_NoiseFigureBW(double* pVal);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_NoiseFigureBW(double pVal);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISegment4</td>
</tr>
</tbody>
</table>

1985
**NoiseFigureBWOption Property**

**Description**
Enables or disables noise figure bandwidth setting.

**VB Syntax**
```vbnet
segs.NoiseFigureBWOption = value
```

**Variable (Type) - Description**

- `segs` A `Segments` collection *(object)*
- `value` *(boolean)*
  - `True` - Enable noise figure bandwidth setting.
  - `False` - Disable noise figure bandwidth setting.

**Return Type**
Boolean

**Default**
False

**Examples**
- `segs.NoiseFigureBWOption = True 'Write`
- `IFBWstate = segs.NoiseFigureBWOption 'Read`

**C++ Syntax**
```csharp
HRESULT get_NoiseFigureBWOption(VARIANT_BOOL* pVal)
HRESULT put_NoiseFigureBWOption(VARIANT_BOOL pVal)
```

**Interface**
ISegments7
# NoiseReceiver Property

**Description**
Sets and returns the receiver to use for noise measurements.

**VB Syntax**
```vbnet
noise.NoiseReceiver = value
```

**Variable**
- **Type** - Description
  - `noise` A `NoiseFigure` (object)
  - `value` (Enum as `NANoiseReceiverMode`) Noise receiver. Choose from:
    - **0 - naStandardReceiver** The standard VNA receiver. (Opt 028 or 029)
    - **1 - naNoiseReceiver** The noise receiver. (Opt 029 ONLY)

**Return Type**
Enum

**Default**
1 - `naNoiseReceiver`

**Examples**
- ```vbnet
  noise.NoiseReceiver = naNoiseReceiver 'Write
  NoiseRec = noise.NoiseReceiver 'Read
  ```

**C++ Syntax**
```c++
HRESULT get_NoiseReceiver(tagNoiseReceiverMode *pVal);
HRESULT put_NoiseReceiver(tagNoiseReceiverMode newVal);
```

**Interface**
`INoiseFigure5`
# NoiseReceiverNoiseBWList Property

**Description**
Returns the list of supported Noise Bandwidths values when using a noise receiver (option 029). Learn more about Opt. 029.

**VB Syntax**

```vbnet
value = cap.NoiseReceiverNoiseBWList
```

**Variable** (Type) - Description

- `value` Variant containing one-dimensional array of long integers.
- `cap` A Capabilities (object)

**Default**
Not Applicable

**Examples**

```vbnet
value = cap.NoiseReceiverNoiseBWList  'Read
```

**C++ Syntax**

```c++
HRESULT get_NoiseReceiverNoiseBWList (Variant *list);
```

**Interface**
ICapabilities13
**NoiseReceiverSweepTime Property**

**Description**

Returns the APPROXIMATE time the channel will take to make one noise receiver sweep given the current setup. This, along with the sweep time for a standard receiver measurement and the following calculations, can tell you how long a “single” sweep would take so that you can set an appropriate "timeout" in your program.

Use **Sweep Time Property** to perform the standard sweep time query, shown as **SSwpTime** below.

**To calculate the total sweep time:**

Noise Figure on amplifiers (Vector Correction ON):

- \[ 2 \times SSwpTime + X \times \text{NoiseReceiverSweepTime} \]

- Where \( X \) = the number of noise receiver impedance state sweeps. (Default is 4).

Noise Figure on converters (NFX) correction on - increased number of sweeps due to extra mixer sweeps and source pulling:

- \[ 4 \times SSwpTime + X \times \text{NoiseReceiverSweepTime} \] (without source pulling)

- \[ 8 \times SSwpTime + X \times \text{NoiseReceiverSweepTime} \] (with source pulling)

- Where \( X \) = the number of noise receiver impedance state sweeps. (Default is 4).

**Note:** The number of sweeps to perform a noise measurement is annotated at the bottom of the Noise Figure screen.

**VB Syntax**

\[ swpTime = NF.NoiseReceiverSweepTime() \]

**Variable**

- **swpTime** Variable to store the returned sweep time value (in seconds).
- **NF** A **NoiseFigure** (object)

**Return Type**

Double

**Default**

Not Applicable

**Examples**

\[ swpTime = NF.NoiseReceiverSweepTime() \]

**C++ Syntax**

```cpp
HRESULT get_NoiseReceiverSweepTime (Double* SwpTime);
```
<table>
<thead>
<tr>
<th>Interface</th>
<th>INoiseFigure3</th>
</tr>
</thead>
</table>

1990
# NoiseGain Property

## Description
Sets and reads the gain state of the noise receiver. This setting is NOT used when \texttt{NoiseReceiver = naStandardReceiver (Opt 028)}.

## VB Syntax
\texttt{noise.NoiseGain = value}

## Variable
- \texttt{noise} (Type) - Description
  - A \texttt{NoiseFigure (object)}
- \texttt{value} (long integer) - Gain value. Choose from:
  - \texttt{0} Low Gain: select if the gain of your DUT is relatively high (>35 dB).
  - \texttt{15} Medium Gain: select if the gain of your DUT is about average (20 dB to 45 dB).
  - \texttt{30} High Gain: select if the gain of your DUT is relatively low (<30 dB).

If the value does not match one of these, it is rounded up to the next legal value.

Learn more about Noise Receiver Gain setting.

## Return Type
Long Integer

## Default
30

## Examples
\texttt{noise.NoiseGain = 30 'Write}

\texttt{GainNoise = noise.NoiseGain 'Read}

## C++ Syntax
\texttt{HRESULT get_NoiseGain(long* pVal)}

\texttt{HRESULT put_NoiseGain(long newVal)}

## Interface
INoiseFigure
### NoiseGainCTCheck

**Description**
Turns noise threshold checking ON and OFF.

**VB Syntax**

```vbnet
noise.NoiseGainCTCheck = value
```

**Variable**

- **noise**
  - A `NoiseFigure2` (object)
- **value**
  - (boolean) - Averaging state.

  - 0- Turns noise threshold checking OFF
  - 1- Turns noise threshold checking ON

**Return Type**
Boolean

**Default**
0 - OFF

**Examples**

```vbnet
noise.NoiseGainCTCheck = OFF 'Write
NoiseCT = noise.NoiseGainCTCheck 'Read
```

**C++ Syntax**

```cpp
HRESULT get_NoiseGainCTCheck(VARIANT_BOOL* on);
HRESULT put_NoiseGainCTCheck(VARIANT_BOOL on);
```

**Interface**
INoiseFigure3
## NoiseSourceCalKitType Property

**Description**
Set and read the Cal Kit that will be used for the Noise Source adapter.

An adapter is always necessary to connect a 346C Noise Source to the VNA port 2. Select a Cal Kit that is the same type and gender as the noise source connector.

If the Noise Source mates directly to VNA port 2, then set this type to "None".

**VB Syntax**

```
noise.NoiseSourceCalKitType = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>noise</code></td>
<td>A NoiseCal (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(string) Cal Kit type. To read possible cal kit strings for the adapter:</td>
</tr>
</tbody>
</table>

1. Change the port connector type to that of the noise source using: `ConnectorType`
2. Then read the possible cal kit strings for that port using: `CompatibleCalKits`

**Return Type**

String

**Default**

Not applicable

**Examples**

```vbnet
noise.NoiseSourceCalKitType = "N4691-60004 ECAL"
```

```vbnet
calkit = noise.NoiseSourceCalKitType
```

**C++ Syntax**

```
HRESULT get_NoiseSourceCalKitType(BSTR* pValue)
```

```
HRESULT put_NoiseSourceCalKitType(BSTR pNewValue)
```

**Interface**

INoiseCal
# NoiseSourceCold Property

**Description**
Sets and returns the temperature of the noise source connector.

**VB Syntax**

```vbnet
noise.NoiseSourceCold = value
```

**Variable (Type) - Description**

- **noise**
  A NoiseCal (**object**)

- **value**
  (**double**) Noise source temperature in Kelvin.

**Return Type**
Double

**Default**
Not Applicable

**Examples**

```vbnet
noise.NoiseSourceCold = 295 'Write
temp = noise.NoiseSourceCold 'Read
```

**C++ Syntax**

```c++
HRESULT get_NoiseSourceCold(double* pTemp)
HRESULT put_NoiseSourceCold(double pNewTemp)
```

**Interface**
INoiseCal
# NoiseSourceConnectorType Property

**Description**  
Set and read the Noise Source connector type and gender.

The Keysight 346C has an "APC 3.5 male" connector.

**VB Syntax**  
```vbnet
noise.NoiseSourceConnectorType = value
```

**Variable**  
(Type) - Description  
- `noise`: A `NoiseCal` (object)  
- `value`: (string) Connector type. Use `ValidConnectorType` to return a list of valid connector types.

**Return Type**  
String

**Default**  
Not applicable

**Examples**  
```vbnet
noise.NoiseSourceConnectorType = "APC 3.5 male" 'Write
connector = noise.NoiseSourceConnectorType 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_NoiseSourceConnectorType(BSTR* pConnectorType)
HRESULT put_NoiseSourceConnectorType(BSTR pConnectorType)
```

**Interface**  
`INoiseCal`
# NoiseSourceState Property

**Description**
Sets and reads the noise source (28V) ON and OFF.

**VB Syntax**

```vbnet
app.NoiseSourceState = state
```

**Variable**

- **app** - An Application (object)
- **state** - (boolean)

**Default**
For VNA models with a Noise Figure option (028/029/H29), the 28V line is ON at application start and after a preset. The ON/OFF state is also available from a VNA softkey menu.

For VNA models WITHOUT a Noise Figure option (028/029/H29), the 28V line is OFF at application start and its state is not affected by a preset. The ON/OFF state is NOT available from a VNA softkey menu.

**Examples**

```vbnet
app.NoiseSourceState = True 'Write
coupl = app.NoiseSourceState 'Read
```

**C++ Syntax**

```c++
HRESULT put_NoiseSourceState(VARIANT_BOOL bState)
HRESULT get_NoiseSourceState(VARIANT_BOOL *bState)
```

**Interface**
IAplication13
### NoiseTuner Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the noise tuner identifier, which is an ECal model and serial number string.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>noise.NoiseTuner = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td></td>
<td><code>noise</code> A NoiseFigure (object)</td>
</tr>
<tr>
<td></td>
<td><code>value</code> (string) Noise Tuner. Return the connected ECal identifiers by sending GetCalKitTypeString and passing the module number.</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>noise.NoiseTuner = &quot;N4691-60004 ECal 02822&quot; 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>noiseT = noise.NoiseTuner 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_NoiseTuner(BSTR* pValue)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_NoiseTuner(BSTR pNewValue)</td>
</tr>
<tr>
<td>Interface</td>
<td>INoiseFigure</td>
</tr>
</tbody>
</table>
## NoiseTunerIn Property

**Description**
Sets and returns the port identifier of the ECal noise tuner that is connected to the VNA Source.

**VB Syntax**
`noise.NoiseTunerIn = value`

**Variable**
- `noise` (Type) - Description
  - A `NoiseFigure` (object)
- `value` (string) Noise Tuner port identifier that is connected to the VNA source, as it is labeled on the ECal module. For example, for 2-port ECal modules, choose either "A" or "B".

**ReturnType**
String

**Default**
Not Applicable

**Examples**
- `noise.NoiseTunerIn = "A"` 'Write
- `EcalPort = noise.NoiseTunerIn` 'Read

**C++ Syntax**
- `HRESULT get_NoiseTunerIn(BSTR* pValue)`
- `HRESULT put_NoiseTunerIn(BSTR pNewValue)`

**Interface**
INoiseFigure
### NoiseTunerOut Property

**Description**  
Sets and returns the port identifier of the ECal noise tuner that is connected to the DUT.

**VB Syntax**  
```
noise.NoiseTunerOut = value
```

**Variable (Type) - Description**
- `noise` (A NoiseFigure object)
- `value` (string) Noise Tuner port identifier that is connected to the DUT, as it is labeled on the ECal module. For example, for 2-port ECal modules, choose either "A" or "B".

**Return Type**  
String

**Default**  
Not Applicable

**Examples**
- `noise.NoiseTunerOut = "A"`  
- `EcalPort = noise.NoiseTunerOut`

**C++ Syntax**
- `HRESULT get_NoiseTunerOut(BSTR* pValue)`
- `HRESULT put_NoiseTunerOut(BSTR pNewValue)`

**Interface**  
INoiseFigure
### NominalIncidentPowerState Property

**Description**
Toggles the Nominal Incident Power setting ON and OFF. This setting is ONLY to be used with SMC measurements, not VMC. Learn more about Nominal Incident Power.

**VB Syntax**
```vbnet
mixer.NominalIncidentPowerState = bool
```

**Variable**
- **mixer** (Type) - Description
  - A Mixer (object)
- **bool** (boolean) - Nominal Incident Power State. Choose from:
  - 1 -(True) Turn nominal incident power ON
  - 0 -(False) Turn nominal incident power OFF

**Return Type**
Boolean

**Default**
0 -(False)

**Examples**
```vbnet
mixer.NominalIncidentPowerState = True 'sets NominalIncidentPowerState to ON
```

**C++ Syntax**
```cpp
HRESULT get_NominalIncidentPowerState(VARIANT_BOOL *pVal)
HRESULT put_NominalIncidentPowerState(VARIANT_BOOL val);
```

**Interface**
IMixer4
NormalizePoint Property

Description
Sets and returns the sweep data point around which to perform broadband and precise tuning.

VB Syntax
obj.NormalizePoint = value

Variable (Type) - Description
obj  An EmbeddedLO (object) or a ConverterEmbeddedLO (object)

value (Long) Mixer Sweep data point. Choose a data point number between 1 and the max number of data points in the sweep that has the least amount of expected noise.

Return Type (Long)

Default  Center point in the sweep span

Examples
embedLO.NormalizePoint = 101 'write
value = embedLO.NormalizePoint 'read

C++ Syntax
HRESULT get_NormalizePoint(long *point);
HRESULT put_NormalizePoint(long point);

Interface IEmbededLO
## NormalizePoint Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the data point used for normalizing the phase measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>obj.NormalizePoint = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>obj</code></td>
<td>A Mixer Interface pointer to the Measurement (object)</td>
</tr>
<tr>
<td></td>
<td>A Converter (Object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(Long)</strong> - Normalization data point. Choose a data point number between 1 and the max number of data points in the sweep that has the least amount of expected noise.</td>
</tr>
</tbody>
</table>

**Return Type**: Long Integer  
**Default**: Center point in the sweep  
**Examples**: `mixer.NormalizePoint = 101`  
**C++ Syntax**:  
```
HRESULT get_NormalizePoint(Long * val);
HRESULT put_NormalizePoint(Long val);
```
**Interface**: IMixer13
### Number (Measurement) Property

**Description**
Returns the Number of the measurement. Measurement numbers are assigned internally.

**Note**: Measurement numbers are NOT the same as their number in the Measurements collection. Measurement number is used to identify the measurement associated with an event.

This property is used to identify measurements when events occur through the OnMeasurementEvent callback. For example:

```vbnet
OnMeasurementEvent (naEventId_MSG_LIMIT_FAILED, 3)
```

<table>
<thead>
<tr>
<th>VB Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>measNum = meas.Number</code></td>
<td>variable to store the measurement number</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable (Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>measNum</code> (long)</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><code>meas</code></td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**: Long Integer

**Default**: "1" - number of the default measurement

**Examples**

```vbnet
measNum = meas.Number
```

**C++ Syntax**: `HRESULT get_Number(long *MeasurementNumber)`

**Interface**: IMeasurement
### Number Property

**Description**
Returns the number of the Auxiliary Trigger connector pair currently being used with the instance of the `AuxTrigger` object. Set the trigger pair with the `AuxTrig` object.

**VB Syntax**

```vbnet
value = auxTrig.Number
```

**Variable**

- `auxTrig` (Type) - Description
  - An `AuxTrigger` (object)

- `value` (Long Integer) - Connector pair. PNA-X returns 1 or 2. All other models that do not have the Aux trigger connector returns 1.

**Return Type**

Single

**Default**

Not Applicable

**Examples**

```vbnet
value = auxTrig.Number 'Read the value
```

**C++ Syntax**

```cpp
HRESULT get_Number(long *auxID);
```

**Interface**

IAuxTrigger
### NumberOfFrequencyPoints Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the number of frequency points for a Gain Compression channel. Applies to all acquisition modes.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>gca.NumberOfFrequencyPoints = value</code></td>
</tr>
</tbody>
</table>
| **Variable (Type)** - Description | **gca** - A `GainCompression` **(object)**  
|  | **value** - **(integer)** - Frequency points. Do not exceed the max number of points. [Learn more](#) |
| **Return Type** | Integer                                                                                                          |
| **Default** | 201                                                                                                              |
| **Examples** | `gca.NumberOfFrequencyPoints = 101` 'Write'  
|  | `freqPts = gca.NumberOfFrequencyPoints` 'Read'                                                                     |
| **C++ Syntax** | HRESULT get_NumberOfFrequencyPoints(int* pVal)  
|  | HRESULT put_NumberOfFrequencyPoints(int newVal)                                                                     |
| **Interface** | IGainCompression                                                                                                   |
## NumberOfPoints (PowerCalRange) Property

**Description**
Sets and gets the number of points for range <m> for source port<n>.

**VB Syntax**

```
PwrrCalRange .NumberOfPoints = points
```

**Variable**
(Type) - Description

- `points` (Long) - Variable to store the returned number of points.

**PwrrCalRange**
A PowerCalRange **(object)**

**Return Type**
Long

**Default**
Not Applicable

**Example**

```
CalibrateAllChannels.IndependentPowerCalibration(3).PowerCalRange(1).NumberOfPoints = 7 'Write

points = CalibrateAllChannels.IndependentPowerCalibration(3).PowerCalRange(1).NumberOfPoints 'Read
```

**C++ Syntax**

```c
HRESULT get_NumberOfPoints(long* points);

HRESULT put_NumberOfPoints(long points);
```

**Interface**
IPowerCalRange
**NumberofPoints Property**

**Description**
Sets or returns the Number of Points of the channel.

Sets or returns the Number of Points of the segment.

**See Also**

- Measurement2 Interface to learn how this method differs from meas.NumberofPoints
- Gain Compression Number of Points.
- Swept IMD limitations

**VB Syntax**

```
object.NumberOfPoints = value
```

**Variable**

*object* (Type) - Description

- **object** Channel (object)

  or

  CalSet (object) - Read-only property

  *value* (long) - Number of Points.

  For channel, choose any number from 1 to the VNA max number of points.

  For segment, the total number of points in all segments cannot exceed the VNA maximum. A segment can have as few as 1 point.

**Return Type**

Long Integer

**Default**

201 for channel

21 for segment

**Examples**

```
chan.NumberOfPoints = 201 'sets the number of points for all measurements in the channel. -Write
numofpts = chan.NumberOfPoints 'Read
```

**C++ Syntax**

```
HRESULT get_NumberOfPoints(long *pVal)
HRESULT put_NumberOfPoints(long newVal)
```
Interface

IChannel
ISegment

|CalSet3
**NumberOfPoints Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of data points of the measurement. To understand how this property is useful, see IMeasurement2 Interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = meas.NumberOfPoints</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) - variable to store the returned value</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>Print meas.NumberOfPoints 'prints the number of data points</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_NumberOfPoints(long *pVal);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
### NumberOfPorts Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of hardware test ports on the VNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = app.NumberOfPorts</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(long integer) - variable to contain the returned value.</td>
</tr>
<tr>
<td>Return Type</td>
<td>(long integer)</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>iNumPorts = app.NumberOfPorts</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT NumberOfPorts(long* NumPorts)</td>
</tr>
<tr>
<td>Interface</td>
<td>IApplication</td>
</tr>
</tbody>
</table>
**NumberOfPorts Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of ports on the specified testset. Returns 0 if no test set is connected.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = tset.NumberOfPorts</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td>value</td>
<td>(Long) variable to store the returned information.</td>
</tr>
<tr>
<td>tset</td>
<td>A <code>TestsetControl</code> object. OR An <code>E5091Testset</code> object.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = testset1.NumberOfPorts</code></td>
</tr>
</tbody>
</table>

See E5091A Example Program
See External Testset Program

**C++ Syntax**

```cpp
HRESULT get_NumberOfPorts(long *numberOfPorts);
```

**Interface**

ITestsetControl

IE5091Testset
### NumberOfPowerPoints Property

**Description**
Set and read the number of data points in each power sweep. Applies ONLY to 2D acquisition modes.

**VB Syntax**
```vbnet
gca.NumberOfPowerPoints = value
```

**Variable (Type) - Description**
- `gca` - A `GainCompression` object
- `value` - (integer) - Power points. Do not exceed the max number of points.

For 2D sweeps, total = frequency x power. Max = 20,001

For Smart sweep, total = frequency. Max = 10,000.

**Return Type**
Integer

**Default**
26

**Examples**
```vbnet
gca.NumberOfPowerPoints = 31 'Write
pwrPts = gca.NumberOfPowerPoints 'Read
```

**C++ Syntax**
```
HRESULT get_NumberOfPowerPoints(int* pVal)

HRESULT put_NumberOfPowerPoints(int newVal)
```

**Interface**
IGainCompression
### NumberOfSweeps Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of tuning sweeps used for the latest embedded LO measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = embedLODiag.NumberOfSweeps</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) Variable to store the returned data.</td>
</tr>
<tr>
<td><code>embedLODiag</code></td>
<td>An EmbeddedLODiagnostic (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>(Long)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>data= embedLODiag.NumberOfSweeps</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_NumberOfSweeps(long * numSweeps);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IEmbeddedLODiagnostic</td>
</tr>
</tbody>
</table>

**About Embedded LO**

Read-only
## OccupiedBandCenter Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the occupied bandwidth center frequency.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = mkr.OccupiedBandCenter</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>value</code></td>
<td>Variable to store the center frequency value.</td>
</tr>
<tr>
<td><code>mkr</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = mkr.OccupiedBandCenter 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_OccupiedBandCenter(double* pVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker7</td>
</tr>
</tbody>
</table>
### OccupiedBandPercent Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return the percentage of the band power to search for.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><em>mkr.OccupiedBandPercent = value</em></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td>mkr</td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>99.0</td>
</tr>
<tr>
<td>Examples</td>
<td>*mkr.OccupiedBandPercent = 99  *Write</td>
</tr>
<tr>
<td></td>
<td>*value = mkr.OccupiedBandPercent  *Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT put_OccupiedBandPercent(double pVal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_OccupiedBandPercent(double* pVal);</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker7</td>
</tr>
</tbody>
</table>
# OccupiedBandPowerdBm Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the occupied bandwidth power.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = mkr.OccupiedBandPowerdBm</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td></td>
<td><code>value</code> - Variable to store the power value.</td>
</tr>
<tr>
<td></td>
<td><code>mkr</code> - A Marker (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = mkr.OccupiedBandPowerdBm 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_OccupiedBandPowerdBm(double* pVal)</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker7</td>
</tr>
</tbody>
</table>
## OccupiedBandSpan Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the occupied bandwidth span.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( value = mkr.OccupiedBandSpan )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>( value ) - Description</td>
</tr>
<tr>
<td>( mkr )</td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>( value = mkr.OccupiedBandSpan ) 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_OccupiedBandSpan(double* pVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker7</td>
</tr>
</tbody>
</table>
### OccupiedBandState Property

**Description**
Sets and returns the occupied bandwidth on/off state.

**VB Syntax**

```vbnet
mkr.OccupiedBandState = value
```

**Variable (Type) - Description**

- **mkr**
  - A Marker (object)
- **value**
  - (Enum as NaStates) - Choose from:
    - **naOFF** (0) - Turns occupied bandwidth OFF.
    - **naON** (1) - Turns occupied bandwidth ON.

**Return Type**
Enum

**Default**
naOFF (0)

**Examples**

```vbnet
mkr.OccupiedBandState = naON 'Write
obwstate = mkr.OccupiedBandState 'Read
```

**C++ Syntax**

```cpp
HRESULT OccupiedBandState(tag NAStates* pState);
HRESULT OccupiedBandState(tag NAStates pState)
```

**Interface**
IMarker7

**Note:** If occupied band state is turned ON, then Band Power or Band Noise is turned OFF.
## Offset Property

Sets and returns the offset value to be used when coupling this range to the primary range.

This setting is valid only if the specified range is coupled to the primary range.

### VB Syntax

\[ \text{FOMRange.Offset} = \text{value} \]

### Variable

- **Object**: An FOMRange (object)
- **Value**: (Double) - Offset value. -(Unitless)

### Return Type

Double

### Default

0

### Examples

- `fomRange.Offset = 1e9` 'Write
- `Offs = fomRange.Offset` 'Read

### C++ Syntax

```
HRESULT get_Offset(double *pVal)
HRESULT put_Offset(double pVal)
```

### Interface

IFOMRange
### About Receiver Attenuation

**OffsetReceiverAttenuator Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return whether to offset the reference receiver by the amount of receiver attenuation. <a href="#">Learn more.</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>This setting remains until changed or until the hard drive is changed or reformatted.</td>
<td></td>
</tr>
</tbody>
</table>

**VB Syntax**

```
pref.OffsetReceiverAttenuator = value
```

**Variable**

- **Type** - **Description**
  - `pref` A Preferences *(object)*
  - `value` *(Boolean)* - Choose from:
    - **False** Do NOT offset the test port receivers.
    - **True** Offset the test port receivers.

**Return Type**

*Boolean*

**Default**

- **True** PNA-X models
- **False** E836xB and PNA-L models

**Examples**

```
pref.OffsetReceiverAttenuator = 1 'Write
Rcvroffset = pref.OffsetReceiverAttenuator 'Read
```

**C++ Syntax**

```
HRESULT get_OffsetReceiverAttenuator(VARIANT_BOOL * val);
HRESULT put_OffsetReceiverAttenuator(VARIANT_BOOL val);
```

**Interface**

*IPreferences6*
**OffsetSourceAttenuator Property**

**Description**
Set and return whether to mathematically offset the reference receivers by the amount of source attenuation. Learn more.

This setting remains until changed or until the hard drive is changed or reformatted.

**VB Syntax**
```
pref.OffsetSourceAttenuator = value
```

**Variable**
- **Type** - Description
- `pref` A Preferences (object)
- `value` (Boolean) - Choose from:
  - **False** Do NOT offset the reference receivers.
  - **True** Offset the reference receivers.

**Return Type**
Boolean

**Default**
- **True** PNA-X models
- **False** E836xB and PNA-L models

**Examples**
```
pref.OffsetSourceAttenuator = 1 'Write
Rcvroffset = pref.OffsetSourceAttenuator 'Read
```

**C++ Syntax**
```
HRESULT get_OffsetSourceAttenuator(VARIANT_BOOL * val);
HRESULT put_OffsetSourceAttenuator(VARIANT_BOOL val);
```

**Interface**
IPreferences6
**OmitIsolation Property - Superseded**

**Description**
This command is replaced with SetIsolationPaths and GetIsolationPaths. Sets and returns whether Isolation portion of the calibration will be performed or not.

**VB Syntax**
```
obj.OmitIsolation = bool
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>obj</code></td>
<td>SMCType (object)</td>
</tr>
<tr>
<td>or</td>
<td>VMCType (object)</td>
</tr>
<tr>
<td><code>bool</code></td>
<td>(Boolean)</td>
</tr>
</tbody>
</table>

- **True** - Isolation is NOT performed
- **False** - Isolation is performed

**Return Type**
Boolean

**Default**
True

**Examples**
```
value = SMC.OmitIsolation
```

**C++ Syntax**
```
HRESULT put_OmitIsolation (VARIANT_BOOL bState)
HRESULT get_OmitIsolation (VARIANT_BOOL *bState)
```

**Interface**
SMCType

VMCType
**OneMarkerReadoutPerTrace Property - Superseded**

**Description**
Either show marker readout of only the active trace or all of the traces simultaneously.

**Note:** This method is replaced by MarkerReadoutsPerTrace Property

**VB Syntax**
```vbnet
win.OneReadoutPerTrace = state
```

**Variable**
- **(Type)** - Description
  - `win` - A NAWindow (object)
  - `value` - (boolean)

- **True** - Shows the readout of only the active marker for each trace.
- **False** - Shows up to 5 marker readouts per trace, up to 20 total readouts.

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
win.OneMarkerReadoutPerTrace = True 'Write

State = app.ActiveNAWindow.OneMarkerReadoutPerTrace 'Read
```

**C++ Syntax**
- HRESULT get_OneMarkerReadoutPerTrace(VARIANT_BOOL *pVal)
- HRESULT put_OneMarkerReadoutPerTrace(VARIANT_BOOL newVal)

**Interface**
INAWindow
## Options Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a string identifying the analyzer option configuration. Refer to the option number differences between the common option numbers and those returned using this command.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = app.Options</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td>app</td>
<td>An Application <em>(object)</em></td>
</tr>
<tr>
<td>value</td>
<td><em>(string)</em> - variable to contain the returned string</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>availOptions = app.Options</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Options(BSTR* OptionString)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>
**OrientECALModule Property**

**Description**
Specifies if the VNA should perform orientation of the ECal module during calibration. Orientation is a technique by which the VNA automatically determines which ports of the module are connected to which ports of the VNA. Orientation begins to fail at very low power levels or if there is much attenuation in the path between the VNA and the ECal module.

**Note:** For guided calibrations where Orient is OFF and the same ECal module is used in more than one Connection Step, you are not allowed to specify how the ECal module is connected. Instead, the VNA determines the orientation. The VNA does not verify that you made the connection properly.

This setting remains until the VNA is restarted or this command is sent again.

This command, and `ECALPortMapEx`, can be used to perform ECal using the Guided Calibration interface.

**VB Syntax**
```
cal.OrientECALModule = value
```

**Variable (Type) - Description**
- `cal` A Calibrator (object)
- `value` (boolean)

- **False** – `DoECAL1PortEX` and `DoECAL2PortEx` methods will use the value of the `ECALPortMapEx` property to determine the port connections.

- **True** - `DoECAL1PortEX` and `DoECAL2PortEx` methods will use auto Orientation technique to determine port connections.

**Return Type**
Boolean

**Default**
True

**Examples**
```
Dim cal As Calibrator
Dim bOrient As Boolean

Set cal = PNAapp.ActiveChannel.Calibrator

cal.OrientECALModule = False 'Write
bOrient = cal.OrientECALModule 'Read
```
C++ Syntax
HRESULT put_OrientECALModule(VARIANT_BOOL bOrient);
HRESULT get_OrientECALModule(VARIANT_BOOL *bOrient);

Interface
ICalibrator3
# OutputChannel Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the output channel of the pulse generator.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>extPulseGen.OutputChannel = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>extPulseGen</code></td>
<td>An <code>ExternalPulseGenerator (object)</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long)</em> Pulse Generator output port. Choose from 1 or 2.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>extPulseGen.OutputChannel = 2 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>port = extPulseGen.OutputChannel 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_OutputChannel (long *pValue)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_OutputChannel (long newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IExternalPulseGenerator</td>
</tr>
</tbody>
</table>
# OutputFixedFrequency Property

**Description**
Sets or returns the mixer output fixed frequency value.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**

```vbnet
obj.OutputFixedFrequency = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>obj</code></td>
<td>A Mixer Interface pointer to the Measurement (object)</td>
</tr>
<tr>
<td>Or</td>
<td>A Converter Object</td>
</tr>
</tbody>
</table>

| `value` | (double) - Output Fixed Frequency in Hertz. |

**Return Type**
Double

**Default**
Not Applicable

**Examples**

- `Print mixer.OutputFixedFrequency` 'prints the output fixed frequency value of the mixer.'

**C++ Syntax**

```cpp
HRESULT get_OutputFixedFrequency(double *pVal)
HRESULT put_OutputFixedFrequency(double newVal)
```

**Interface**
IMixer3
IConverter
Write/Read

About Testset Control

OutputPort Property

Description

Switches an input to one of the valid outputs on the specified E5091A. The following are valid input/output combinations. If a combination other than these are sent, an “invalid argument” error will occur.

<table>
<thead>
<tr>
<th>Input</th>
<th>Valid Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td>T1 - If Port 2 already is connected to T1, then Port 2 will be switched to T2.</td>
</tr>
<tr>
<td>2</td>
<td>T1 - If Port 1 already is connected to T1, then Port 1 will be switched to A.</td>
</tr>
<tr>
<td></td>
<td>T2</td>
</tr>
<tr>
<td>3</td>
<td>R1+</td>
</tr>
<tr>
<td></td>
<td>R2+</td>
</tr>
<tr>
<td></td>
<td>R3+ If option 007 (7port), R2 is selected.</td>
</tr>
<tr>
<td>4</td>
<td>R1-</td>
</tr>
<tr>
<td></td>
<td>R2-</td>
</tr>
<tr>
<td></td>
<td>R3- If option 007 (7port), R2 is selected.</td>
</tr>
</tbody>
</table>

Note: Do not confuse the similar Testset.OutputPorts Property, which sets or gets the port mapping for ALL ports.

VB Syntax

```
testsets(1).OutputPort (chNum,input) = output
```

Variable

(Type) - Description

*testsets(1)* An item from *Testsets (collection)*

Learn how to identify a testset in the collection.

*chNum* (Long) Channel number of the measurement.

*input* (Long) Testset Input port. Choose from 1|2|3|4.

*output* (Enum as NAE5091OutputPort) Output port to switch to specified Input. Choose from:

0 or naE5091PortA - Port A

1 or naE5091PortT1 - Port T1

2 or naE5091PortT2 - Port T2

2029
3 or **naE5091PortR1** - Port R1

4 or **naE5091PortR2** - Port R2

5 or **naE5091PortR3** - Port R3

**Return Type**  
Enum

**Default**  
Not Applicable

**Examples**  
See Example Program

**C++ Syntax**  
HRESULT get_OutputPort(long channelNum, long inputPort, E5091OutputPort *outPort);
HRESULT put_OutputPort(long channelNum, long inputPort, E5091OutputPort outPort);

**Interface**  
IE5091Testsets
**OutputPorts (Cal Set) Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the port mapping for the Cal Set.</th>
</tr>
</thead>
</table>

**VB Syntax**  
`portMap = calset.OutputPorts`

**Variable**  
*(-Type) - Description*

`portMap`  
(String) Variable to store the returned string. The returned values are the physical ports. The POSITION of the returned values corresponds to the logical ports.

For example, with an N44xx test set, if the returned string is "PNA 1,TS 2,PNA 2, TS 4" this means:

- PNA 1 is assigned to logical port 1
- TS 2 is assigned to logical port 2
- PNA 2 is assigned to logical port 3
- TS 4 is assigned to logical port 4

**calset**  
A Cal Set object.

**Return Type**  
String

**Default**  
Depends on the test set.

**Example**  
`portMap = calset.OutputPorts`

**C++ Syntax**  
`HRESULT get_OutputPorts(BSTR *mapping);`

**Interface**  
ICalset5
OutputPorts Property

Description
Sets or gets the port mappings for ALL ports. An “invalid argument” error will occur if you attempt to set an illegal port combination.

Refer to your testset documentation for valid port combinations.

Note: Do not confuse the similar E5091.OutputPort Property, which sets or gets the port mapping for a single port.

VB Syntax
\[ tset.OutputPorts(chNum) = portList \]

Variable (Type) - Description

- **tset** A TestsetControl object.
- **chNum** (Long) Channel number of the measurement.
- **portList** (String) A comma-separated list of port mappings. Spaces are ignored at the beginning and end of this text, and before or after commas. Space characters in other locations are not ignored.

Return Type String

Default Not Applicable

Examples See External Testset Program

C++ Syntax

```cpp
HRESULT get_OutputPorts(long channelNum, BSTR *outPorts);
HRESULT put_OutputPorts(long channelNum, BSTR outPorts);
```

Interface ITestsetControl
OutputRangeMode Property

**Description**
Sets or returns the Output sweep mode.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**Note:** There is also a OutputRangeMode Property on the Converter Object.

**VB Syntax**

```vbnet
obj.OutputRangeMode = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>A Mixer Interface pointer to the Measurement (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Enum as MixerRangeMode) - Output sweep mode. Choose from:</td>
</tr>
</tbody>
</table>

0 - mixSwept

1 - mixFixed

**Return Type**
Enum

**Default**
0 - mixSwept

**Examples**

```vbnet
mixer.OutputRangeMode = mixSwept
```

**C++ Syntax**

```c++
HRESULT get_OutputRangeMode(long *pVal)

HRESULT put_OutputRangeMode(long newVal)
```

**Interface**
IMixer6
## OutputRangeMode Property

**Description**
Sets or returns the Output sweep mode.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the *Calculate* and *Apply* commands as you would do from the user interface.

**Note:** There is also a `OutputRangeMode` Property on the Mixer Interface.

### VB Syntax

```vbnet
obj.OutputRangeMode = value
```

### Variable

**Type** - Description

| obj | A Converter Object |

<table>
<thead>
<tr>
<th>value</th>
<th>(Enum as NARangeMode) - Output sweep mode. Choose from:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td><code>naSwept</code></td>
</tr>
<tr>
<td>1</td>
<td><code>naFixed</code></td>
</tr>
</tbody>
</table>

### Return Type

Enum

### Default

0 - `naSwept`

### Examples

```vbnet
conv.OutputRangeMode = naSwept
```

### C++ Syntax

```cpp
HRESULT get_OutputRangeMode(long *pVal)

HRESULT put_OutputRangeMode(long newVal)
```

### Interface

IConverter
### OutputSideband Property

**Description**  Specify whether to select the sum (High) or difference (Low) products.

- When one LO is used: Input (+ or -) LO1 = Output frequency.
- When two LO's are used: IF (+ or -) LO2 = Output frequency. See Also: **IFSideband Property**

If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

**VB Syntax**  

```
obj.OutputSideband = value
```

**Variable (Type) - Description**

- **obj**  A Converter Object
- **value**  *(enum as ConverterSideBand)*  - Choose from:
  - 0 - or **naLowSide**  Minus (-) on the Mixer setup dialog
  - 1 - or **naHighSide**  Plus (+) on the Mixer setup dialog

**Return Type**  Enum as ConverterSideBand

**Default**  naLowSide

**Examples**  

```
Print converter.OutputSideband 'prints the value of the OutputSideband
```

**C++ Syntax**

```
HRESULT get_OutputSideband(ConverterSideBand *pVal)

HRESULT put_OutputSideband(ConverterSideBand newVal)
```

**Interface**  IConverter
### OutputSideband Property

**Description**
Specify whether to select the sum (High) or difference (Low) products.

- When one LO is used: Input + or - LO1 = Output frequency
- When two LOs are used: IF1 + or - LO2 = Output frequency

Use **IFSideband_Property** when two LOs are used to determine the IF1 frequency.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

**Note:** There is also an **OutputSideband_Property** on the Converter Object.

**VB Syntax**

```
mixer.OutputSideband = value
```

**Variable** *(Type) - Description*

- **mixer** A **Mixer** *(object)*
- **value** *(FCASideBand)* -

Choose from:

- **0 - LOW**  Minus (-) on the Mixer setup dialog
- **1 - HIGH**  Plus (+) on the Mixer setup dialog

**Return Type**
Enum as FCASideBand

**Default**
LOW

**Examples**

```
Print mixer.OutputSideband  'prints the value of the OutputSideband
```

**C++ Syntax**

```
HRESULT get_OutputSideband(FCASideBand *pVal)
HRESULT put_OutputSideband(FCASideBand newVal)
```

**Interface**
IMixer
## OutputStartFrequency Property

### Description
Sets or returns the mixer output start frequency.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

### VB Syntax
```vbnet
obj.OutputStartFrequency = value
```

### Variable (Type) - Description
- **obj**: A Mixer Interface pointer to the Measurement (object)
  
  Or

  A Converter Object

- **value**: (double) - Output Start Frequency in Hertz.

### Return Type
Double

### Default
Not Applicable

### Examples
```vbnet
Print mixer.OutputStartFrequency 'prints the value of the OutputStartFrequency
```

### C++ Syntax
```cpp
HRESULT get_OutputStartFrequency(double *pVal)

HRESULT put_OutputStartFrequency(double newVal)
```

### Interface
- IMixer
- IConverter
OutputStopFrequency Property

Description
Sets or returns the mixer Output Stop frequency.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

VB Syntax
obj.OutputStopFrequency = value

Variable
(obj) - Description

obj
A Mixer Interface pointer to the Measurement (object)

Or

A Converter Object

value (double) - Output stop frequency in Hertz.

Return Type
Double

Default
Not Applicable

Examples
Print mixer.OutputStopFrequency 'prints the value of the OutputStopFrequency

C++ Syntax
HRESULT get_OutputStopFrequency(double *pVal)

HRESULT put_OutputStopFrequency(double newVal)

Interface
IMixer
IConverter
## Parameter Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the measurement Parameter. To change the parameter, use <code>meas.ChangeParameter</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>measPar = meas.Parameter</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>measPar</code></td>
<td><strong>(string) - Variable to store Parameter string</strong></td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (<strong>object</strong>)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>measPar = meas.Parameter</code> 'Read'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT <code>get_Parameter(BSTR *pVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
### Parameter (Embedded LO) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the name of the parameter of the specified tuning sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = embedLODiag.Parameter(n)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(String)</em> Variable to store the returned data.</td>
</tr>
<tr>
<td><code>embedLODiag</code></td>
<td>An EmbeddedLODiagnostic <em>(object)</em></td>
</tr>
<tr>
<td><code>n</code></td>
<td><em>(Long)</em> Tuning sweep number. Use NumberOfSweeps to find the number of sweeps taken.</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>data = embedLO.Parameter 3 'read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT Parameter(long sweep, BSTR * param);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IEmbededLODiagnostic</td>
</tr>
</tbody>
</table>
**ParameterList Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a list of all existing parameters.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>params = DIQ.ParameterList</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>params</code></td>
<td>(Variant Array) Comma-separated list of parameters, each parameter in the form “name:expression”.</td>
</tr>
<tr>
<td><code>DIQ</code></td>
<td>A Differential I/Q (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>(Variant Array)</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Example</td>
<td><code>params = diq.ParameterList</code></td>
</tr>
<tr>
<td></td>
<td>'Read'</td>
</tr>
<tr>
<td></td>
<td>'Returns'</td>
</tr>
<tr>
<td></td>
<td>&quot;IPwrF1:a1_F1&quot;,&quot;OPwrF1:b2_F1&quot;,&quot;GainF1:b2_F1/a1_F1&quot;</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_ParameterList(VARIANT* params);</td>
</tr>
<tr>
<td>Interface</td>
<td>IDIQ</td>
</tr>
</tbody>
</table>
**Parent Property**

**Description**
Returns a handle to the parent object of the collection object being referred to in the statement. The parent property allows the user to traverse from an object back up the object hierarchy.

**VB Syntax**
```
collection.Parent
```

**Variable**
(Variable) - Description
- `collection`

**Return Type**
Object

**Default**
Not Applicable

**Examples**
```
parentobj = chans.Parent 'returns a handle to the parent object (Application) of the chans collection. -Read
```

**C++ Syntax**
```
HRESULT get_Parent(IApplication* *pApplication); //IChannels, IChannel, IMeasurements, INAWindows, and IExternalDevices
```
<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All listed above</td>
<td></td>
</tr>
</tbody>
</table>
PassFailLogic Property

**Description**
Sets and reads the logic of the PassFail line on the HANDLER IO connector (pin 33). Learn more about Global Pass/Fail.

*Note:* This line is connected to both the Handler IO and Aux IO in the VNA. Therefore, this command will affect both of these connectors in the same way.

**VB Syntax**

```
object.PassFailLogic = value
```

**Variable** *(Type) - Description*

- **object** *(object)* - An Aux I/O or Handler I/O object
- **value** *(enum as NARearPanelIOLogic)* Choose from:

  0 - **naPositiveLogic** - Causes the PassFail line to have positive logic (high = pass, low = fail).

  1 - **naNegativeLogic** - Causes the PassFail line to have negative logic (high = fail, low = pass).

**Return Type**
Long Integer

**Default**
aPositiveLogic

**Examples**

```vbscript
aux.PassFailLogic = naNegativeLogic 'Write
Text1.Text = aux.PassFailLogic 'Read
```

**C++ Syntax**

```
HRESULT put_PassFailLogic ( tagNARearPanelIOLogic Mode );
HRESULT get_PassFailLogic ( tagNARearPanelIOLogic* Mode );
```

**Interface**

IHWAuxIO

IHWMaterialHandlerIO
**PassFailMode Property**

**Description**
Sets and reads the mode of the PassFail line on the HANDLER IO connector (pin 33). Learn more about Global Pass/Fail.

**VB Syntax**
```vbnet
object.PassFailMode = value
```

**Variable**
- **object** - An Aux I/O or Handler I/O object
- **value** - (enum as NAPassFailMode). Choose from:
  - **0 - naDefaultPassNoWaitMode** - the line stays in PASS state. When a device fails, then the line goes to fail IMMEDIATELY.
  - **1 - naDefaultPassWaitMode** - the line stays in PASS state. When a device fails, then the line goes to fail after the Sweep End line is asserted.
  - **2 - naDefaultFailWaitMode** - the line stays in FAIL state. When a device passes, then the line goes to PASS state after the Sweep End line is asserted.

**Return Type**
Long Integer

**Default**
0 - naDefaultPassNoWaitMode

**Examples**
```vbnet
HWAuxIO.PassFailMode = naDefaultPassNoWaitMode 'Write
mode = HWAuxIO.PassFailMode 'Read
```

**C++ Syntax**
```cpp
HRESULT put_PassFailMode ( tagNAPassFailMode Mode );
HRESULT get_PassFailMode ( tagNAPassFailMode* Mode );
```

**Interface**
IHWAuxIO
IHWMaterialHandlerIO
**PassFailPolicy Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the policy used to determine how global pass/fail is computed. Learn more about Global Pass/Fail.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>object.PassFailPolicy = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td></td>
<td><strong>object</strong> - An Aux I/O or Handler I/O object</td>
</tr>
<tr>
<td></td>
<td><strong>value</strong> - (enum as NAPassFailPolicy) Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <strong>naPolicyAllTests</strong> - Pass/Fail Status returns PASS if all tests on all measurements pass.</td>
</tr>
<tr>
<td></td>
<td>1 - <strong>naPolicyAllMeas</strong> - Pass/Fail Status returns PASS if all measurements have associated tests, and all tests pass. FAIL is returned if even one measurement has no associated limit test. Only those measurements which are not in HOLD mode contribute to the pass/fail result.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long Integer</td>
</tr>
<tr>
<td>Default</td>
<td>naPolicyAllTests</td>
</tr>
<tr>
<td>Examples</td>
<td><code>matHndler.PassFailPolicy = naPolicyAllTests</code></td>
</tr>
<tr>
<td></td>
<td><code>policy = aux.PassFailPolicy</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT put_PassFailPolicy ( tagNAPassFailPolicy Policy);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_PassFailPolicy ( tagNARearPanelIOLogic* Policy);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IHWAuxIO4</td>
</tr>
<tr>
<td></td>
<td>IHWMaterialHandlerIO2</td>
</tr>
</tbody>
</table>
PassFailScope Property

Description
Sets and reads the Scope of the PassFail line on the HANDLER IO connector (pin 33). Learn more about Global Pass/Fail.

VB Syntax
object.PassFailScope = value

Variable
(object) - An Aux I/O or Handler IO object
(value) - Choose from:

0 - naChannelScope - The PassFail line returns to its default state before sweeps on the next channel start. (A channel measurement may require several sweeps.)

1 - naGlobalScope - The PassFail line returns to its default state before the sweeps for the next triggerable channel start.

The default state of the PassFail line before a measurement occurs and after a failure occurs is set by the PassFailMode property.

Return Type
enum NAPassFailScope

Default
1 - naGlobalScope

Examples
HWAuxIO.PassFailScope = naGlobalScope 'Write
scope = HWAuxIO.PassFailScope 'Read

C++ Syntax
HRESULT put_PassFailScope ( tagNAPassFailScope Scope );
HRESULT get_PassFailScope ( tagNAPassFailScope* Scope );

Interface
IHWAuxIO
IHWMaterialHandlerIO
Read-Only

PassFailStatus Property

**Description**
Returns the most recent pass/fail status value. Use this command as follows:

1. Set the VNA trigger scope to GLOBAL
2. Set the VNA trigger source to MANUAL or EXTERNAL.
3. Configure and enable Limit Testing
4. Trigger the VNA.
5. Use the *OPC? (with SCPIStringParser object) to determine when the sweep is complete.
6. Use the **PassFailStatus** property to obtain the global pass/fail result.

Learn more about Global Pass/Fail.

**VB Syntax**
```
var = object.PassFailStatus
```

**Variable**
- **Type** - Description
  - **var** (enum as NAPassFailStatus) Variable to store returned status. One of the following will be returned:
    - **0 - naStatusFail** - all measurements not in HOLD mode have been swept, and one or more limit tests failed according to the specified Pass/Fail policy.
    - **1 - naStatusPass** - all measurements not in HOLD mode have been swept, and all associated limit tests have passed.
    - **2 - naStatusNone** - status cannot be determined because measurements are in progress.

- **object** (object) - An Aux I/O or Handler I/O object

**Return Type**
- Long Integer

**Default**
- Not Applicable

**Examples**
```
status = aux.PassFailStatus 'Read
```

**C++ Syntax**
```
HRESULT get_PassFailPolicy ( tagNAPassFailStatus* status);
```

**Interface**
- IHWAuxIO4
- IHWMaterialHandlerIO2
### Path Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies an interface to use for the power meter / sensor during a source power calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>pwrMtrInterface.Path = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description <code>pwrMtrInterface</code> (object) <code>value</code> (enum as NACommunicationPath)</td>
</tr>
<tr>
<td></td>
<td>Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naGPIB - GPIB interface</td>
</tr>
<tr>
<td></td>
<td>1 - naUSB - USB interface</td>
</tr>
<tr>
<td></td>
<td>2 - naLAN - LAN interface</td>
</tr>
<tr>
<td></td>
<td>3 - naAny - Any VISA resource string or a visa alias</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum</td>
</tr>
<tr>
<td>Default</td>
<td>naGPIB</td>
</tr>
<tr>
<td>Examples</td>
<td>See example</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT put_Path(tagNACommunicationPath pNewVal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_Path(tagNACommunicationPath *pVal);</td>
</tr>
<tr>
<td>Interface</td>
<td>IPowerMeterInterface</td>
</tr>
</tbody>
</table>
PathCalMethod Property

**Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
<th>Description</th>
</tr>
</thead>
</table>
| guidedCal | GuidedCalibration (object) | **Note:** This command replaces ThruCalMethod.  
(Read-Write) Specifies the calibration method for each port pair. |
| port1 | First port of the pair to be calibrated. | **Note:** Sending this command will overwrite the VNAs SmartCal determinations for the most accurate cal method for your connector settings and Cal Kits. Send this command ONLY if you have a deliberate reason for overwriting the SmartCal logic. You can send the query form of this command to learn the cal method determined by SmartCal. |
| port2 | Second port of the pair to be calibrated. | |
| "caltype1[,caltype2]" | (String) Cal types for 1st and 2nd ports of the pair, enclosed in a single pair of quotes. NOT case-sensitive. | See Thru Pairs Sequence to learn how to send this and other Thru commands. |

After sending this command, send the query form to be sure that the command was accepted. If not, then the chosen Cal method is not compatible with the specified Thru method. For example, if the specified Thru method is Unknown Thru, an attempt to set Enhanced Response Cal should be rejected.

See an example of a 4-port guided calibration using COM.

**VB Syntax**

```vbnet
guidedCal.PathCalMethod (port1, port2) = "caltype1[,caltype2]"
```

**caltype1** Cal type for the pair if caltype2 is not specified. Otherwise, Cal type for port 1. Choose from:

- “TRL”
- “SOLT”
- “QSOLTN”
- “EnhRespN”

For the last two arguments, replace N with the port to be used as the source port, which MUST be one of the port pair.
Optional argument. Use only when performing an adapter removal cal on the pair. This argument specifies the Cal Type on the second port; caltype1 then specifies the Cal Type of the first port.

Choose from the same arguments as caltype1.

**Return Type**  
**String** - Returns comma-separated cal types.

**Default**  
The most accurate cal method for the current cal.

**Example**

```
guidedCal.PathCalMethod(2,3) = "TRL" 'Write trl for port pair

guidedCal.PathCalMethod(1,4) = "TRL,SOLT" 'Write adapter removal cal, consisting of trl on port 1 and solt on port 4

calmethod = guided.PathCalMethod(1,4) 'Read previous example, returns: "TRL,SOLT"
```

**C++ Syntax**

```cpp
HRESULT get_PathCalMethod(long firstport, long secondport, BSTR *calMethod);

HRESULT put_PathCalMethod(long firstport, long secondport, BSTR calMethod);
```

**Interface**  
IGuidedCalibration3
PathConfigurationElement Property

Description  Allows you to set and return the Path Configuration settings for a Cal All calibration.

VB Syntax  \texttt{calAll.PathConfigurationElement(element).value} = \texttt{setting}

Variable  \begin{itemize}
  \item \texttt{calAll} - A \texttt{CalibrateAllChannels (object)}
  \item \texttt{element} - (String) Path configuration element to be set. \texttt{See a list of configurable RF Path elements.}
\end{itemize}

\textit{Is this also for IF Path configuration elements?}

\texttt{setting} - (String) Path configuration element setting.

Return Type  String

Default  Default settings vary with each element.

Examples  \texttt{calAll.PathConfigurationElement("Port1NoiseTuner").value = "Internal"}

\textquote{returns the PathConfigurationElement setting}

\texttt{value = calAll.PathConfigurationElement("Port1NoiseTuner").value}

C++ Syntax  \texttt{HRESULT put\_PathConfigurationElement(BSTR name, IPathElement** pElement);}

Interface  ICalibrateAllChannels
### PathElement Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Returns the name of the setting for the given path element name or sets the value of a path element.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>powerRange.PathElement(element).value = setting</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>powerRange</code></td>
<td>A PowerRange (object)</td>
</tr>
<tr>
<td><code>element</code></td>
<td>(String) Path configuration element to be set. See a list of configurable RF Path elements.</td>
</tr>
<tr>
<td><code>setting</code></td>
<td>(String) Path configuration element setting.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Default settings vary with each element.</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>`powerRange.PathElement(&quot;Src2Out1LowBand&quot;).value = &quot;HiPwr&quot;</td>
</tr>
<tr>
<td></td>
<td>'returns the PathElement setting</td>
</tr>
<tr>
<td></td>
<td><code>value = powerRange.PathElement(&quot;Src2Out1LowBand&quot;).value</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_PathElement(BSTR name, IPathElement** pElement);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPowreRange</td>
</tr>
</tbody>
</table>
**PathElements Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a string with the names of all valid RF path configuration elements.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( \text{elements} = \text{powerRange.PathElements} )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><strong>powerRange</strong></td>
<td>A PowerRange (object)</td>
</tr>
<tr>
<td><strong>elements</strong></td>
<td>(String) Variable to store RF path configuration element names.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>( \text{elements} = \text{powerRange.PathElements} )</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_PathElements(BSTR *pathElements);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPowreRange</td>
</tr>
</tbody>
</table>
**PathThruMethod Property**

**Description**

**Note:** This command replaces ThruCalMethod.

(Read-Write) Specifies the calibration THRU method for each port pair.

**Note:** Sending this command will overwrite the VNAs SmartCal determination for the thru method. Send this command ONLY if you have a deliberate reason for overwriting the SmartCal logic. You can send the query form of this command to learn the THRU method determined by SmartCal.

See Thru Pairs Sequence to learn how to send this and other Thru commands.

See an example of a 4-port guided calibration using COM.

**VB Syntax**

```
guidedCal.PathThruMethod (port1, port2) = "ThruType1[,ThruType2]"
```

**Variable (Type) - Description**

- `guidedCal` GuidedCalibration (object)
- `port1` First port of the pair to be calibrated.
- `port2` Second port of the pair to be calibrated.
- "ThruType1[,ThruType2]" (String) Thru methods for 1st and 2nd ports of the pair, enclosed in a single pair of quotes. NOT case-sensitive.

**thruType1** Calibration thru method for the pair if thruType2 is not specified. Otherwise, thru method for port 1.

Choose from:

- **“Defined Thru”** Measures a Thru for which there is a stored definition in the Cal Kit.
- **“Zero Thru”** Measures a Zero length Thru, also known as Flush-Thru.
- **“Undefined Thru”** A thru type for which there is NOT a stored definition in the Cal Kit. Also known as Unknown Thru. Valid ONLY for SOLT cal type.
- **“Undefined Thru using a Defined Thru”** (ECal modules ONLY) Measures the internal Thru as an Unknown Thru.
**ThruType2** (String) Optional argument. Use ONLY when Adapter Removal Cal is specified for the pair using PathCalMethod. When specifying ThruType2, this is the only valid argument: "**Defined Thru, Defined Thru**"

**Return Type**  **String** - Returns comma-separated ThruTypes.

Always returns two parts:

If the second part of the string is empty, adapter removal is NOT being performed.

If the string is "Defined Thru, Defined Thru", adapter removal IS being performed.

**Default** The most accurate THRU method for the current cal.

**Example**

```
guidedCal.PathThruMethod(2,3) = "Zero Thru"  'Write for port pair

guidedCal.PathThruMethod(1,4) = "Defined Thru, Defined Thru"  'Write for adapter removal cal.

calmethod = guided.PathThruMethod(1,4)  'Read previous example, return: "Defined Thru, Defined Thru"
```

**C++ Syntax**

```
HRESULT get_PathThruMethod(long firstport, long secondport, BSTR *thruMethod);

HRESULT put_PathThruMethod(long firstport, long secondport, BSTR thruMethod);
```

**Interface** IGuidedCalibration3
PeakExcursion Property

Description: Sets and reads the peak excursion value for the specified marker. The Excursion value determines what is considered a "peak".

VB Syntax: `mark.PeaekExcursion = value`

Variable (Type) - Description
- `mark`: A Marker (object)
- `value`: (single) - Peak Excursion. Choose any number between -500 and 500

Return Type: Single

Default: 3

Examples:
- `mark.PeakExcursion = 1 'Write`
- `PkExcur = mark.PeakExcursion 'Read`

C++ Syntax:
- `HRESULT get_PeakExcursion(float *pVal)`
- `HRESULT put_PeakExcursion(float newVal)`

Interface: IMarker
## PeakThreshold Property

**Description**
Sets peak threshold for the specified marker. If a peak (using the criteria set with PeakExcursion) is below this reference value, it will not be considered when searching for peaks.

**VB Syntax**
```vbnet
mark.PeerThreshold = value
```

**Variable**
- **Type** - Description
  - `mark` - A Marker (object)
  - `value` - Peak Threshold. Choose any number between -500 and 500

**Return Type**
Single

**Default**
-100db

**Examples**
```vbnet
mark.PeakThreshold = 1 'Write
PkThresh = mark.PeakThreshold 'Read
```

**C++ Syntax**
```cpp
HRESULT get_PeakThreshold(float *pVal)
HRESULT put_PeakThreshold(float newVal)
```

**Interface**
IMarker
### PeakToPeak Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Peak to Peak value of the measurement. To retrieve all 3 statistics value at the same time, use <code>meas.GetTraceStatistics</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pp = meas.PeenToPeak</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>pp</code></td>
<td><strong>(single) - Variable to store peak-to-peak value</strong></td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement <strong>(object)</strong></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Single</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pp = meas.PeenToPeak</code> <strong>(Read)</strong></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_PeakToPeak(float* pp)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
Percent Property

Description Sets group delay aperture using a percent of the channel frequency span.

VB Syntax `gdAperture.Percent = value`

Variable (Type) - Description

`gdAperture` A `GroupDelayAperture` (object)

`value` (Double) Percent of frequency span to use for the aperture setting. Choose between the equivalent of 2 points and 100 percent of the channel frequency span.

Return Type Double

Default Percent range that equates to 11 points.

This can be changed to two points with a preference setting.

Examples `gdAperture.Percent = 25` 'Write

`aperture = gdAperture.Percent` 'Read

C++ Syntax

`HRESULT get_Percent(double Percent *pVal)`

`HRESULT put_Percent(double Percent newVal)`

Interface `IGroupDelayAperture`
**PerformPowerCalibration Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables Guided Power Cal and sets the source port to be calibrated.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>guidedCal.PerformPowerCalibration (port) = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td></td>
<td><code>guidedCal</code> GuidedCalibration (object)</td>
</tr>
<tr>
<td></td>
<td><code>port</code> (Long integer) Source port to be calibrated. ONLY one port may be calibrated with a Guided Power Cal.</td>
</tr>
<tr>
<td></td>
<td><code>value</code> (Boolean)</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> Perform a Guided Power Calibration.</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> Do NOT perform a Guided Power Calibration</td>
</tr>
<tr>
<td>Return Type</td>
<td><strong>Boolean</strong></td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
<tr>
<td>Example</td>
<td><code>guided.PerformPowerCalibration(1) = True</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_PerformPowerCalibration(long port,VARIANT_BOOL* val);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_PerformPowerCalibration(long port,VARIANT_BOOL newVal);</td>
</tr>
<tr>
<td>Interface</td>
<td>IGuidedCalibration7</td>
</tr>
</tbody>
</table>
## Period Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the pulse-period (1/PRF) for ALL PNA-X internal pulse generators. The resolution of the period is 16.667nS.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>pulse.Period = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td>pulse</td>
<td>A PulseGenerator <em>(object)</em></td>
</tr>
<tr>
<td>value</td>
<td><em>(Double)</em> Pulse period in seconds. Choose a value from about 33ns to about 70 seconds.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>1e-3 sec</td>
</tr>
<tr>
<td>Examples</td>
<td><code>pulse.Period = 1ms 'Write</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_Period(double* period);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_Period(double period);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IPulseGenerator</td>
</tr>
</tbody>
</table>
### PhaseOffset Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the Phase Offset for the active channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>meas.PhaseOffset = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement <strong>(object)</strong></td>
</tr>
<tr>
<td>value</td>
<td><strong>(double)</strong> - PhaseOffset in degrees. Choose any number between: -360 and +360</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
<tr>
<td>Examples</td>
<td><code>meas.PhaseOffset = 25</code> <strong>(Write)</strong></td>
</tr>
<tr>
<td></td>
<td><code>poffset = meas.PhaseOffset</code> <strong>(Read)</strong></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_PhaseOffset(double *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_PhaseOffset(double newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
## PhaseAsFixture Property

**Description**
Sets and reads the state of phase offset as a fixture with True Mode balanced measurements. Learn more about iTMSA phase and power offset.

**VB Syntax**
```vb
balStim.PhaseAsFixture = value
```

**Variable**
- **balStim**: A BalancedStimulus (object)
- **value**: (Boolean) State of phase offset as a fixture.

**Return Type**
Boolean

**Default**
False

**Examples**
- `balStim.PhaseAsFixture = False` 'Write
- `var = balStim.PhaseAsFixture` 'Read

**C++ Syntax**
```cpp
HRESULT get_PhaseAsFixture (VARIANT_BOOL *bVal)

HRESULT put_PhaseAsFixture (VARIANT_BOOL bVal)
```

**Interface**
IBalancedStimulus
## PhaseControlMode Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Write and read the Phase Control mode for the specified port.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>phase.PhaseControlMode(srcPort) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><strong>phase</strong></td>
<td>A PhaseControl Object</td>
</tr>
<tr>
<td><strong>srcPort</strong></td>
<td>(Long Integer) Source port for which to make phase control settings.</td>
</tr>
</tbody>
</table>

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

<table>
<thead>
<tr>
<th><strong>value</strong></th>
<th>(Enum as NAPhaseControlMode) Choose from:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - <strong>naPhaseControlOff</strong></td>
<td>Do NOT control the phase of <code>&lt;port&gt;</code>.</td>
</tr>
<tr>
<td>1 - <strong>naPhaseControlParameter</strong></td>
<td>Control the phase of <code>&lt;port&gt;</code>.</td>
</tr>
<tr>
<td>2 - <strong>naPhaseControlReference</strong></td>
<td>(READ-only) - reference port for a 'controlled' (parameter) Port.</td>
</tr>
<tr>
<td>3 - <strong>naPhaseControlOpenLoop</strong></td>
<td>Sets a raw phase value for either swept phase or fixed phase, but no receivers are used to control the phase.</td>
</tr>
</tbody>
</table>

| **Return Type** | Enum |
| **Default** | 0 - **naPhaseControlOff** |

**Examples**

```vbnet
phase.PhaseControlMode 1 = naPhaseControlParameter ' Write
value = phase.PhaseControlMode 2' Read
```

**C++ Syntax**

```c++
HRESULT get_PhaseControlMode(long port, enum NAPhaseControlMode* pVal);

HRESULT put_PhaseControlMode(long port, enum NAPhaseControlMode newVal);
```

**Interface**

`IPhaseControl`
# PhaseCorrectionData Property

**Description**
Write and read an array of phase offsets. Each phase offset is summed with each phase point to get new target value. The VNA attempts to set phase for each target value. The number of phase offset values must be the same as the number of data points.

**VB Syntax**
```
phase.PhaseCorrectionData(srcPort) = value
```

**Variable**
- `(Type) - Description`
- `phase` A `PhaseControl` Object
- `srcPort` (Long Integer) Source port for which to make phase control settings.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

**Value**
(Long values) Phase offset data array.

**Return Type**
Long data array

**Default**
Not Applicable

**Examples**
- `phase.PhaseCorrectionData 1 = 10,11,12`  
  Write 3 phase offset values
- `value = phase.PhaseCorrectionData 2`  
  Read

**C++ Syntax**
```
HRESULT get_PhaseCorrectionData(long port, long* pVals);

HRESULT put_PhaseCorrectionData(long port, long newVals);
```

**Interface**
IPhaseControl
# PhaseCorrectionEnabled Property

**Description**  
Write and read whether to use the phase correction offset array. Use PhaseCorrectionData to write or read the offset data.

**VB Syntax**  
```
phase.PhaseCorrectionEnabled(srcPort) = value
```

**Variable**  
**Type** - Description
- `phase` A `PhaseControl` Object
- `srcPort` (Long Integer) Source port for which to make phase control settings.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port

- `value` (Boolean) Phase correction array state.
  - **True** – Apply phase correction offset array.
  - **False** – Do NOT apply phase correction offset array.

**Return Type**  
Boolean

**Default**  
False

**Examples**  
```
phase.PhaseCorrectionEnabled 1 = True ' Write
dis = phase.PhaseCorrectionEnabled 2' Read
```

**C++ Syntax**  
```
HRESULT get_PhaseCorrectionEnabled(long port, VARIANT_BOOL* pVal);

HRESULT put_PhaseCorrectionEnabled(long port, VARIANT_BOOL newVal);
```

**Interface**  
IPhaseControl
### PhaseDisplayMinLevel Property

**Description**
Set and read the phase display minimum level.

**VB Syntax**
```
sa.coherence.PhaseDisplayMinLevel = value
```

**Variable**

- `sa.coherence` (Type) - Description
  - A `SpectrumAnalyzerCoherence` object
- `value` (Double) - Phase display minimum level (in dBm).

**Return Type**
Double

**Default**
-50 dBm

**Examples**
```
sa.coherence.PhaseDisplayMinLevel = -50 dBm  'Write
value = sa.coherence.PhaseDisplayMinLevel  'Read
```

**C++ Syntax**
```
HRESULT put_PhaseDisplayMinLevel(double level);
HRESULT get_PhaseDisplayMinLevel(double* levell);
```

**Interface**
ICoherenceSA2
### PhaselIterationNumber Property

**Description**
Write and read the maximum number of background phase sweeps to perform.

**VB Syntax**

```vbnet
phase.PhaseIterationNumber(srcPort) = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phase</td>
<td>A PhaseControl Object</td>
</tr>
<tr>
<td>srcPort</td>
<td>(Long Integer) Source port for which to make phase control settings.</td>
</tr>
</tbody>
</table>

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(Double) Number of background sweep iterations. Choose a value between 1 and 25.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
5

**Examples**

```vbnet
phase.PhaseIterationNumber 1 = 15 ' Write
value = phase.PhaseIterationNumber 2' Read
```

**C++ Syntax**

```csharp
HRESULT get_PhaseIterationNumber(long port, double* pVal);

HRESULT put_PhaseIterationNumber(long port, double newVal);
```

**Interface**
IPhaseControl
## PhaseParameter Property

**Description**
Write and read the ratioed receivers (parameter) and paired port to use for phase control.

**VB Syntax**

```vbnet
phase.PhaseParameter(srcPort) = value
```

**Variable**

- **phase** (Type) - Description
  A PhaseControl Object

- **srcPort** (Long Integer) Source port for which to make phase control settings.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port

- **value** (String) Ratioed parameter. Choose any two VNA physical receivers. Use either standard receiver notation ("R/R3") or logical receiver notation ("a1/a3"). Separate the two receiver names by a forward slash '/'. For example: "a1/a3".

**Return Type**
String

**Default**
"a1/b1"

**Examples**

```vbnet
phase.PhaseParameter 1 = "a1/a3" ' Write
value = phase.PhaseParameter 2' Read
```

**C++ Syntax**

```cpp
HRESULT get_PhaseParameter(long port, BSTR* pVal);

HRESULT put_PhaseParameter(long port, BSTR newVal);
```

**Interface**
IPhaseControl
## PhaseParameterModes Property

**Description**
Returns the available phase control modes for the specified port. Use `PhaseControlMode` to set the Phase Control mode.

**VB Syntax**

```vbnet
phase.PhaseParameterModes(srcPort) = value
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>phase</code></td>
<td>A <code>PhaseControl</code> Object</td>
<td></td>
</tr>
<tr>
<td><code>srcPort</code></td>
<td>(Long Integer) Source port for which to make phase control settings.</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number.

To learn more see [Remotely Specifying a Source Port](#).

<table>
<thead>
<tr>
<th><code>value</code></th>
<th>(String) Choose from:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Off&quot;</td>
<td>- Turn phase control OFF</td>
<td></td>
</tr>
<tr>
<td>&quot;Openloop&quot;</td>
<td>- Sets a raw phase value for either swept phase or fixed phase, but no receivers are used to control the phase.</td>
<td></td>
</tr>
<tr>
<td>&quot;Parameter&quot;</td>
<td>- Sets and controls the phase of the signal at &lt;port&gt;.</td>
<td></td>
</tr>
<tr>
<td>&quot;Reference&quot;</td>
<td>- Reference port for a controlled (parameter) port. Use <code>PhaseReferencePort</code> to set the reference port.</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type** String

**Default** "Off"

**Example**

```vbnet
value = phase.PhaseParameterModes 2' Read
```

**C++ Syntax**

```c
HRESULT get_PhaseParameterModes(long port, BSTR* pVal);
```

**Interface** IPhaseControl
# PhaseProcessState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables/disables phase computing.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>sa.coherence.PhaseProcessState = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>sa.coherence</code></td>
<td>A <code>SpectrumAnalyzerCoherence</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(Boolean)</strong> Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>0</strong> - <strong>OFF</strong> - Phase computing disabled.</td>
</tr>
<tr>
<td></td>
<td><strong>1</strong> - <strong>ON</strong> - Phase computing enabled.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>OFF</td>
</tr>
<tr>
<td>Examples</td>
<td><code>sa.coherence.PhaseProcessState = ON</code> <code>Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = sa.coherence.PhaseProcessState</code> <code>Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_PhaseProcessState(VARIANT_BOOL* enable)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_PhaseProcessState(VARIANT_BOOL enable)</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ICoherenceSA2</td>
</tr>
</tbody>
</table>
### PhaseReference Property

**Description**  
Sets and returns the Phase Reference ID to be used for the Phase Reference calibration. Use `GetConnectedPhaseReferences` to read the phase references currently connected to the VNA USB.

<table>
<thead>
<tr>
<th>VB Syntax</th>
<th><code>phasRef.PhaseReference = value</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>phasRef</code></td>
<td>A <code>PhaseReferenceCalibration</code> Object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(String) Phase Reference ID name.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**Examples**  
```
phase.PhaseReference = "MYPRT0001"
```

**See example program**

**C++ Syntax**  
```
HRESULT get_PhaseReference(BSTR PhaseReference* pVals);
HRESULT put_PhaseReference(BSTR PhaseReference newVals);
```  

**Interface**  
`IPhaseReferenceCalibration`
### PhaseReferencePort Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the reference port for the Phase Control measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>phase.PhaseReferencePort(srcPort) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>phase</code></td>
<td>A PhaseControl Object</td>
</tr>
<tr>
<td><code>srcPort</code></td>
<td>(Long Integer) Source port for which to make phase control settings.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use <code>chan.getPortNumber</code> to translate the string into a port number. To learn more see <a href="#">Remotely Specifying a Source Port</a>.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long Integer) Reference port number. ONLY specific ports are available to be a reference for each source port. <a href="#">Learn more</a>.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Depends on the controlled port.</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>phase.PhaseReferencePort 1 = 3 ' Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = phase.PhaseReferencePort 2 ' Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_PhaseReferencePort(long port, long* pVal);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_PhaseReferencePort(long port, long newVal);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPhaseControl</td>
</tr>
</tbody>
</table>
## PhaseSwpAsFixture Property

**Description**
Sets and reads the state of phase offset as a fixture with True Mode balanced measurements.

**VB Syntax**
`balStim.PhaseAsFixture = value`

**Variable**
- **Type**: Description
- **balStim**: A BalancedStimulus (object)
- **value**: (Boolean) State of phase offset as a fixture.

**False**  Offset is applied but is NOT included as a fixture in the output calculations.

**True**  Offset is applied and included as a fixture in the output calculations.

**Return Type**
Boolean

**Default**
False

**Examples**
- `balStim.PhaseAsFixture = False`  'Write
- `var = balStim.PhaseAsFixture`  'Read

**C++ Syntax**
- HRESULT get_PhaseAsFixture (VARIANT_BOOL *bVal)
- HRESULT put_PhaseAsFixture (VARIANT_BOOL bVal)

**Interface**
IBalancedStimulus2
### PhaseSweepPoints Property

**Description**  
Set and read the number of phase points. For the tuning tone at the output, a phase sweep is done for each point.

**VB Syntax**  
```vbnet
HotS22.PhaseSweepPoints = value
```

**Variable**  
*Type* - Description  

| HotS22 | A `ActiveParametersApp` *(object)* |

| value | *(Long)* - Number of phase points. Do not exceed the max number of phase points (50). |

**Return Type**  
Long

**Default**  
8

**Examples**  
```vbnet
HotS22.PhaseSweepPoints = 201  'Write
value = HotS22.PhaseSweepPoints  'Read
```

**C++ Syntax**  
```c++
HRESULT get_PhaseSweepPoints(long* value)

HRESULT put_PhaseSweepPoints(long value)
```

**Interface**  
IActiveChannelSettings
### PhaseSwpState Property

**Description**
Sets and reads the state of phase sweep. **Sweep type** must be set to CWTime.

**VB Syntax**
```
balStim.PhaseSwpState = value
```

**Variable (Type) - Description**
- **balStim**
  A BalancedStimulus (object)
- **value**
  (Boolean) State of phase sweep.

  - **False**  Phase sweep disabled.
  - **True**  Phase sweep enabled.

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
balStim.PhaseSwpState = False 'Write
var = balStim.PhaseSwpState 'Read
```

**C++ Syntax**
```
HRESULT get_PhaseSwpState (VARIANT_BOOL *bVal)
```
```
HRESULT put_PhaseSwpState (VARIANT_BOOL bVal)
```

**Interface**
IBalancedStimulus2
# PhaseTolerance Property

**Description**
Write and read the tolerance value to be used for background phase sweeps.

**VB Syntax**
```vbnet
phase.PhaseTolerance(srcPort) = value
```

**Variable**
- *(Type)* - Description
- **phase**
  A *PhaseControl* Object
- **srcPort**
  (Long Integer) Source port for which to make phase control settings.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port

**value**
(Double) Tolerance for background sweeps in degrees. Choose a value between 1 and 5.

**Return Type**
Double

**Default**
5 degrees

**Examples**
```vbnet
phase.PhaseTolerance 1 = 3 ' Write
value = phase.PhaseTolerance 2' Read
```

**C++ Syntax**
```c++
HRESULT get_PhaseTolerance(long port, double* pVal);
HRESULT put_PhaseTolerance(long port, double newVal);
```

**Interface**
IPhaseControl
## Pin Property

| Description | Returns the Pin result of a PNOP or PSAT marker search.  
|             | PNOP In = Marker 4 X-axis value  
|             | PSAT In = Marker 2 X-axis value  
| VB Syntax   | \( pIn = pMark.Pin \)  
| Variable    | (Type) - Description  
| \( pIn \)   | (double) - Variable to store returned value  
| \( pMark \) | A PNOP (object) or PSaturation (object)  
| Return Type | Double  
| Default     | Not applicable  
| Examples    | \( pIn = pMark.Pin \)  

See example program  

### C++ Syntax

```
HRESULT get_Pin(double* pNewVal)
```

### Interface

IPNOP or IPSaturation
### PinOffset Property

**Description**

Sets and returns the PinOffset value used to calculate various PNOP parameters.

Also set **BackOff Property**.

A sweep must be executed (single or continuous) and **SearchPowerNormalOperatingPoint Method** must be sent before reading marker results.

To turn off the PNOP markers, either turn them off individually or **DeleteAllMarkers**.

To search a **User Range** with the PNOP search, first activate marker 1. The user range used with the PNOP search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

**VB Syntax**

```vbnet
pnop.PinOffset = value
```

**Variable**

*(Type) - Description*

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pnop</code></td>
<td>A PNOP (object)</td>
<td></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double)</td>
<td>PinOffset value in dB. Choose any number between:-500 and 500</td>
</tr>
</tbody>
</table>

**Return Type**

Double

**Default**

0 dB

**Examples**

```vbnet
pinOffs = pnop.PinOffset 'Read
```

See example program

**C++ Syntax**

```cpp
HRESULT put_PinOffset(double newVal);

HRESULT get_PinOffset(double* pNewVal)
```

**Interface**

IPNOP
**PMaxBackOff Property**

**Description**
Sets and returns the backoff value used to calculate various PSAT parameters.

A sweep must be executed (single or continuous) and `SearchPowerSaturation Method` must be sent before reading marker results.

To turn off the PSAT markers, either turn them off individually or `DeleteAllMarkers`.

To search a User Range with the PSAT search, first activate marker 1. The user range used with the PNOP search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

**VB Syntax**

```vbnet
pSat.PMaxBackOff = value
```

**Variable (Type) - Description**

- `pSat` A `PSaturation (object)`
- `value` (double) - Backoff value in dB. Choose any number between -500 and 500

**Return Type**
Double

**Default**
0 dB

**Examples**

```vbnet
backoff = pSat.PMaxBackOff 'Read
```

See example program

**C++ Syntax**

```c++
HRESULT put_PMaxBackOff(double newVal);

HRESULT get_PMaxBackOff(double* pNewVal)
```

**Interface**
`IPSaturation`
# PMaxIn Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the PMaxIn result of a PNOP and PSAT marker search.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMax In</td>
<td>Marker 3 X-axis value</td>
</tr>
</tbody>
</table>

## VB Syntax

```vb
pMaxIn = pMark.PMaxIn
```

## Variable

- **Type**: (double)
- **Description**: Variable to store returned value
- **pMaxIn**: A PNOP (object) or PSaturation (object)

## Return Type

Double

## Default

Not applicable

## Examples

```vb
pMaxIn = pMark.PMaxIn 'Read
```

See example program

## C++ Syntax

```cpp
HRESULT get_PMaxIn(double* pNewVal)
```

## Interface

IPNOP or IPSAT
### PMaxOut Property

**Description**

Returns the PMaxOut result of the PNOP and PSAT marker search.

\[ \text{PMaxOut} = \text{Marker 3 Y-axis value} \]

**VB Syntax**

\[ pMaxOut = pMark.PMaxOut \]

**Variable**

- **Type**: (double)
- Description: Variable to store returned value

- **pMark**: A PNOP (object) or PSaturation (object)

**Return Type**

Double

**Default**

Not applicable

**Examples**

```
pMaxOut = pMark.PMaxOut 'Read
```

See example program

**C++ Syntax**

```
HRESULT get_PMaxOut(double* pNewVal)
```

**Interface**

IPNOP or IPSaturation
## PointAveragingState Property

### Description
Turns point averaging ON or OFF for all measurements on the channel.

### VB Syntax
```vbnet
chan.PointAveragingState = state
```

### Variable
- **Type** - Description
  - **chan** A Channel (object)
  - **state** (Enun)

#### False
- Turns point averaging OFF

#### True
- Turns point averaging ON

### Return Type
Boolean

### Default
False

### Examples
```vbnet
chan.PointAveragingState = True 'Write
aver = chan.PointAveragingState 'Read
```

### C++ Syntax
```csharp
HRESULT get_PointAveragingState(BOOL *pVal)
HRESULT put_PointAveragingState(BOOL newVal)
```

### Interface
IChannel16
PointCmd Property

Description
Sets and returns the Point Read Commands for an external DC Meter or Point Set Commands for an external DC Source.

VB Syntax
extDC.PointCmd = value

Variable
extDC An ExternalDCDevice (object)
value (String)

For DC Source, sets the Point Set Commands. Use {\%f} to specify a double value and {\%d} to specify an integer.

For DC Meter, sets the Point Read Commands (for example, meas:volt?).

Return Type String

Default " " Empty String

Examples
extDC.PointCmd = "sour1:volt {%f}" 'Write
value = extDC.PointCmd 'Read

C++ Syntax
HRESULT get_PointCmd(BSTR* cmd)
HRESULT put_PointCmd(BSTR cmd)

Interface IExternalDCDevice3
### PointDwell Property

**Description**
Sets and returns the "Dwell Before/After Point" value for an external DC Device which can be configured as either a DC Meter or a DC Source.

**VB Syntax**
`extDC.PointDwell = value`

**Variable**
*extDC* - An *ExternalDCDevice* (object)

*value* - *(Double)*

For DC Meter, the dwell time (in seconds) before making a data point measurement.

For DC Source, the dwell time (in seconds) after making a data point setting.

**Return Type**
Double

**Default**
3 milliseconds

**Examples**
```
extDC.PointDwell = 10e-3 'Write
dwell = extDC.PointDwell 'Read
```

**C++ Syntax**

```
HRESULT get_PointDwell (double *pValue)
HRESULT put_PointDwell (double newVal)
```

**Interface**
IExternalDCDevice
### Points Property

**Description**
Sets group delay aperture using a fixed number of data points.

**VB Syntax**
```vbnet
gdAperture.Points = value
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>gdAperture</th>
<th>A GroupDelayAperture (object)</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(Double) Number of data points to use for the aperture setting. Choose between two points and the number of points in the channel.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
Points range that equates to 11 points.

This can be changed to two points with a preference setting.

**Examples**
```vbnet
gdAperture.Points = 25 'Write
aperture = gdAperture.Points 'Read
```

**C++ Syntax**
```cpp
HRESULT get_Points(double Points *pVal)  
HRESULT put_Points(double Points newVal)
```

**Interface**
IGroupDelayAperture
## PointSweepState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns point sweep ON or OFF for all measurements on the channel. Point sweep measures both the forward and reverse parameters at each frequency point before stepping to the next frequency. The display trace is updated after the forward and reverse parameters are measured at that frequency point.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan.PointSweepState = state</code></td>
</tr>
</tbody>
</table>
| **Variable (Type) - Description** | `chan` A Channel (object)  

`state` (boolean) 

- **False** - Turns point sweep OFF  
- **True** - Turns point sweep ON |
| **Return Type** | Boolean |
| **Default** | False |
| **Examples** | 

```vbnet
chan.PointSweepState = True 'Write
averg = chan.PointSweepState 'Read
```

**C++ Syntax** | `HRESULT get_PointSweepState(VARIANT_BOOL *pVal)`  

`HRESULT put_PointSweepState(VARIANT_BOOL newVal)` |
| **Interface** | IChannel16 |
### Port1 Property  **Superseded**

<table>
<thead>
<tr>
<th>Description</th>
<th>This command is replaced by <a href="#">PortDelay property</a>. Sets a Port Extension value for Port 1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>portExt.Port1 = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>portExt</code></td>
<td>A Port Extension <strong>(object)</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(double)</strong> - Port Extension value in seconds. Choose any number between -10 and 10</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
</tbody>
</table>
| **Examples** | `portExt.Port1 = 10e-6 'Write`  
`prt1 = portExt.Port1  'Read` |
| **C++ Syntax** |  
HRESULT get_Port1(double *pVal)  
HRESULT put_Port1(double newVal) |
| **Interface** | IPortExtension |
**Port2 Property  Superseded**

**Description**  This command is replaced by PortDelay property.

Sets a Port Extension value for Port 2

**VB Syntax**  
`portExt.Port2 = value`

**Variable**  
- `portExt` (Type) - Description
- `value` (double) - Port Extension value in seconds. Choose any number between -10 and 10

**Return Type**  Double

**Default**  0

**Examples**  
- `portExt.Port2 = 10e-6`  'Write
- `prt2 = portExt.Port2`  'Read

**C++ Syntax**  
- `HRESULT get_Port2(double *pVal)`
- `HRESULT put_Port2(double newVal)`

**Interface**  IPortExtension
Port3 Property  **Superseded**

Description  This command is replaced by *PortDelay property.*

Sets a Port Extension value for Port 3

VB Syntax  

```vbnet
portExt.Port3 = value
```

Variable  

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>portExt</td>
<td>A Port Extension <em>(object)</em></td>
</tr>
<tr>
<td>value</td>
<td><em>(double)</em> - Port Extension value in seconds. Choose any number between -10 and 10</td>
</tr>
</tbody>
</table>

Return Type  Double

Default  0

Examples  

```vbnet
portExt.Port3 = 10e-6  'Write
prt3 = portExt.Port3  'Read
```

C++ Syntax  

```c++
HRESULT get_Port3(double *pVal)
HRESULT put_Port3(double newVal)
```

Interface  IPortExtension
### Port1NoiseTunerSwitchPresetsToExternal Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the default setting for the Noise Tuner switch. <a href="#">Learn more.</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>pref.Port1NoiseTunerSwitchPresetsToExternal = bool</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td></td>
<td><code>pref</code> - A Preferences (object)</td>
</tr>
<tr>
<td></td>
<td><code>bool</code> - Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Sets the default (preset) to INTERNAL</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Sets the default (preset) to EXTERNAL</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>True</td>
</tr>
<tr>
<td>Examples</td>
<td><code>pref.Port1NoiseTunerSwitchPresetsToExternal = False</code> <strong>Write</strong></td>
</tr>
<tr>
<td></td>
<td><code>prefer = pref.Port1NoiseTunerSwitchPresetsToExternal</code> <strong>Read</strong></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT put_Port1NoiseTunerSwitchPresetsToExternal(VARIANT_BOOL bValue)</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_Port1NoiseTunerSwitchPresetsToExternal(VARIANT_BOOL *bValue)</td>
</tr>
<tr>
<td>Interface</td>
<td>IPreferences8</td>
</tr>
</tbody>
</table>
Port2PdeembedCktModel Property

Description Select whether or not to load a 2-port De-embedding circuit model for the specified port number. Circuit model is applied when both "USER" is selected and the filename is specified. To set the filename, use strPort2Pdeembed_S2PFile Property

Note: This command affects ALL measurements on the channel.

VB Syntax  
```
fixture.Port2PdeembedCktModel(port) = value
```

Variable (Type) - Description
- `fixture` (A Fixturing (object))
- `port` (Integer) Port number to receive circuit model.
- `value` (Enum as NAFixturing2PdeembedCkt)
  0 `naFix2PD_USER` load a 2-port De-embedding circuit model
  1 `naFix2PD_NONE` no model

Return Type Long Integer

Default `naFix2PD_NONE`

Examples
```
fixture.Port2PdeembedCktModel(2) = naFix2PD_USER  'Write
value = fixture.Port2PdeembedCktModel(1)       'Read
```

C++ Syntax
```
HRESULT get_Port2PdeembedCktModel(short port tagNAfixturing2PdeembedCkt *pVal)
HRESULT put_Port2PdeembedCktModel(short port tagNAfixturing2PdeembedCkt newVal)
```

Interface IFixturing
### Port2PdeembedState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns de-embedding ON or OFF for all ports on the channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>fixture.Port2PdeembedState = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>fixture</code></td>
<td>A Fixturing <strong>(object)</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(Boolean)</strong></td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>- Turns De-embedding OFF</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>- Turns De-embedding ON</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>fixture.Port2PdeembedState = False</code> <strong>'Write</strong></td>
</tr>
<tr>
<td></td>
<td><code>value = fixture.Port2PdeembedState</code> <strong>'Read</strong></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Port2PdeembedState(VARIANT_BOOL *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Port2PdeembedState(VARIANT_BOOL newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFixturing</td>
</tr>
</tbody>
</table>
PortArbzImag Property

Description
Sets and returns the Imaginary portion of the impedance value for the specified single-ended port. Use PortArbzReal to set the real value. Or use PortArbzZ0 to set both values together.

VB Syntax
```
fixture.PortArbzImag(portNum) = value
```

Variable
(Type) - Description
-fixture (A Fixturing object)
-portNum (Integer) Single-ended port number to receive impedance value.
-value (Double) Real Impedance value. Choose a value between -1E18 and 1E18

Return Type
Double

Default
0

Examples
```
fixture.PortArbzImag(2) = 75 'Write
value = fixture.PortArbzImag(1) 'Read
```

C++ Syntax
```
HRESULT get_PortArbzImag( short portNum, double *pVal)
HRESULT put_PortArbzImag( short portNum, double newVal)
```

Interface
IFixturing3
PortArbzReal Property

Description
Sets and returns the Real portion of the impedance value for the specified single-ended port. Use PortArbzImag to set the imaginary value. Or use PortArbzZ0 to set both values together.

VB Syntax
```vbnet
fixture.PortArbzReal(portNum) = value
```

Variable
- `fixture` (Type) - Description
  - A Fixturing (object)
- `portNum` (Integer) - Single-ended port number to receive impedance value.
- `value` (Double) - Real Impedance value. Choose a value between 0 to 1E7

Return Type
- Double

Default
- 50

Examples
```vbnet
fixture.PortArbzReal(2) = 75 'Write
value = fixture.PortArbzReal(1) 'Read
```

C++ Syntax
```cpp
HRESULT get_PortArbzReal( short portNum, double *pVal)
HRESULT put_PortArbzReal( short portNum, double newVal)
```

Interface
- IFixturing3
### PortArbzState Property

**Description**
Turns Port Impedance ON or OFF for all ports on the channel.

**VB Syntax**
```vbnet
fixture.PortArbzState = value
```

**Variable (Type) - Description**
- **fixture** A Fixturing (object)
- **value** (Boolean)

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>Turns Port Impedance OFF</td>
</tr>
<tr>
<td>True</td>
<td>Turns Port Impedance ON</td>
</tr>
</tbody>
</table>

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
fixture.PortArbzState = False 'Write
value = fixture.PortArbzState 'Read
```

**C++ Syntax**
- HRESULT get_PortArbzState(VARIANT_BOOL *pVal)
- HRESULT put_PortArbzState(VARIANT_BOOL newVal)

**Interface**
IFixturing
## PortArbzZ0 Property

**Description**
Sets and returns the Real portion of the impedance value for the specified single-ended port. The imaginary portion is automatically set to 0.0.

To set both values separately, use `PortArbzReal` and `PortArbzImag`.

**VB Syntax**

```vbnet
fixture.PortArbzZ0(portNum) = value
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fixture</td>
<td>A Fixturing</td>
<td>(object)</td>
</tr>
<tr>
<td>portNum</td>
<td>(Integer)</td>
<td>Single-ended port number to receive impedance value.</td>
</tr>
<tr>
<td>value</td>
<td>(Double)</td>
<td>Impedance value. Choose a value between 0 to 1E7.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
50

**Examples**

```vbnet
fixture.PortArbzZ0(2) = 75 'Write
value = fixture.PortArbzZ0(1) 'Read
```

**C++ Syntax**

```csharp
HRESULT get_PortArbzZ0( short portNum, double *pVal)
HRESULT put_PortArbzZ0( short portNum, double newVal)
```

**Interface**
IFixturing3
### PortCatalog Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a comma-separated list of the Output port selections that are available for a given logical input port. Read the number of input ports for the test set using <code>NumberOfPorts</code> Property.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = tset.PortCatalog(logPort)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(String) Variable to store the returned information.</td>
</tr>
<tr>
<td><code>tset</code></td>
<td>A <code>TestsetControl</code> object.</td>
</tr>
<tr>
<td><code>logPort</code></td>
<td>(Long) Logical Input port number for which to return valid output ports.</td>
</tr>
<tr>
<td>Return Type</td>
<td><strong>String</strong></td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = testset1.PortCatalog 2</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_PortCatalog(long inputPort, BSTR *outPort);</code></td>
</tr>
<tr>
<td>Interface</td>
<td><code>ITestsetControl</code></td>
</tr>
</tbody>
</table>
# PortCLogic Property

Sets and reads the logic mode of Port C on the Handler IO connector.

## VB Syntax

```vbnet
AuxIO.PortCLogic = value
```

## Variable

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuxIO (object) - A Hardware Aux I/O object</td>
<td>value (Enum as NaRearPanelIOLogic) - Choose from:</td>
</tr>
<tr>
<td>0 - <code>naPositiveLogic</code> - The associated data line goes <strong>HIGH</strong> when writing a 1 to a PortC bit.</td>
<td>1 - <code>naNegativeLogic</code> - The associated data line goes <strong>LOW</strong> when writing a 1 to a PortC bit.</td>
</tr>
</tbody>
</table>

When Port C is in Output/Write mode, a change in logic causes the output lines to change state immediately. For example, Low levels change to High levels.

When Port C is in Input/Read mode, a change in logic will not cause the lines to change, but data read from Port C will reflect the change in logic.

## Return Type

Enum

## Default

1 - `naNegativeLogic`

## Examples

```vbnet
auxIO.PortCLogic = value 'Write
value = auxIo.PortCLogic 'Read
```

## C++ Syntax

```c++
HRESULT put_PortCLogic ( tagNARearPanelIOLogic Mode );
HRESULT get_PortCLogic ( tagNARearPanelIOLogic* Mode );
```

## Interface

IHWAuxIO
## PortCMode Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads whether Port C is setup for writing or reading data on the Handler IO connector.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( \text{AuxIO.PortCMode} = \text{value} )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>( \text{AuxIO} )</td>
<td>(object) - A Hardware Aux I/O object</td>
</tr>
<tr>
<td>( \text{value} )</td>
<td>(enum as NaPortMode) - Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - \text{naInput} - set the port for reading</td>
</tr>
<tr>
<td></td>
<td>1 - \text{naOutput} - set the port for writing</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum as NaPortMode</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1 - naInput</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>( \text{auxIo.get_PortCMode} = \text{naInput} ) 'Write</td>
</tr>
<tr>
<td></td>
<td>( \text{value} = \text{auxIo.get_PortCMode} ) 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT _PortCMode( tagNAPortMode* pMode );</td>
</tr>
<tr>
<td></td>
<td>HRESULT _PortCMode( tagNAPortMode pMode );</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IHWAuxIO</td>
</tr>
</tbody>
</table>
## PortCoupleToSystemMedia Property

**Description**  
Sets and returns the state of coupling with the system Media type.

**VB Syntax**  
`fixture.PortCoupleToSystemMedia (port) = value`

**Variable** *(Type)* - Description  
- `fixture` A Fixturing Object  
- `value` (Boolean) Coupling state. Choose from:
  - **True** - Media type is coupled with the system setting.  
  - **False** - Media type is NOT coupled with the system setting.

**Return Type** Boolean

**Default** True

**Examples**
```vbnet
fixture.PortCoupleToSystemMedia = True
bool = fixture.PortCoupleToSystemMedia
```

**C++ Syntax**
```c
HRESULT get_PortCoupleToSystemMedia(VARIANT_BOOL *pVal);
HRESULT put_PortCoupleToSystemMedia(VARIANT_BOOL newVal);
```

**Interface** IFixturing4
### PortCoupleToSystemVelocity Property

Sets and returns the state of coupling with the system Velocity Factor.

**VB Syntax**

```vbnet
fixture.PortCoupleToSystemVelocity(port)= value
```

**Variable**

- **Type** - Description
- **fixture** A Fixturing Object
- **port** (Integer) Port Number that will receive the coupling change.
- **value** (Boolean) Coupling state. Choose from:
  - **True** - Velocity Factor is coupled with the system setting.
  - **False** - Velocity Factor is NOT coupled with the system setting.

**Return Type**

Boolean

**Default**

True

**Examples**

```vbnet
fixture.PortCoupleToSystemVelocity(2)= True
bool = fixture.PortCoupleToSystemVelocity(1)
```

**C++ Syntax**

```cpp
HRESULT get_PortCoupleToSystemVelocity(short portNum, VARIANT_BOOL *pVal);

HRESULT put_PortCoupleToSystemVelocity(short portNum, VARIANT_BOOL newValue);
```

**Interface**

IFixturing4
PortDelay Property

Description
Sets and returns the Port Extensions Delay value for the specified port number.

*Note:* This command affects ALL measurements on the channel.

This command replaces Port 1 Port 2 Port_3 Properties.

VB Syntax

```
fixture.PortDelay(port) = value
```

Variable (Type) - Description
- `fixture` A Fixturing (object)
- `port` (Integer) Port number to receive delay value.
- `value` (Double) Delay value in seconds. Choose a value between -1E18 and 1E18.

Return Type
Double

Default
0

Examples

```
fixture.PortDelay(2) = .002 'Write
value = fixture.PortDelay(1) 'Read
```

C++ Syntax

```
HRESULT get_PortDelay(short port double *pVal)
HRESULT put_PortDelay(short port double newVal)
```

Interface
IFixturing
PortDescription Property

Description  For each port of the ECal module that is going to be characterized, sets and reads the description of the adapters, cable, or fixture to be included in the user characterization. This description is stored with the characterization in the ECal module.

Set this description before sending Initialize or the default (empty string) will be used.

VB Syntax  

```
userChar.PortDescription (port)= value
```

Variable  (Type) - Description

- **userChar**: An ECalUserCharacterizer Object
- **port**: (Enum) ECal port for which description is to be set. Choose from:
  - 1 or naECalPort_A
  - 2 or naECalPort_B
  - 3 or naECalPort_C
  - 4 or naECalPort_D

- **value**: (String) Descriptive text, limited to 24 characters maximum.

Return Type  String

Default  "" (Empty String)

Examples  

```vbnet
userChar.PortDescription (naECalPort_C)= "3.5 mm adapter, SN 00001"
```

C++ Syntax  HRESULT get_PortDescription(NAECalPort port, *BSTR description);

HRRESULT put_PortDescription(NAECalPort port, BSTR description);

Interface  IECalUserCharacterizer
## PortDistance Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the port extension delay in physical length (distance).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>fixture.PortDistance (port)= value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>fixture</code></td>
<td>A Fixturing Object</td>
</tr>
<tr>
<td><code>port</code></td>
<td>(Integer) Port Number that will receive the delay setting.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Physical length in distance. First specify units with</td>
</tr>
<tr>
<td></td>
<td><code>PortDistanceUnit</code></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>fixture.PortDistance(2)= .01</code></td>
</tr>
<tr>
<td></td>
<td><code>value = fixture.PortDistance(2)</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_PortDistance(short portNum, double *pVal);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_PortDistance(short portNum, double newVal);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFixturing4</td>
</tr>
</tbody>
</table>
# PortNADistanceUnit Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the units for specifying port extension delay in physical length (distance).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>fixture.PortNADistanceUnit = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>fixture</code></td>
<td>A Fixturing Object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum naDistanceUnit) Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 – naDistanceUnit_Meter</td>
</tr>
<tr>
<td></td>
<td>1 – naDistanceUnit_Feet</td>
</tr>
<tr>
<td></td>
<td>2 – naDistanceUnit_Inch</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>naDistanceUnit_Meter</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>fixture.PortNADistanceUnit(2) = naDistanceUnit_Meter </code> <em>Write</em></td>
</tr>
<tr>
<td></td>
<td><code>value = fixture.PortNADistanceUnit(2) </code> <em>Read</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_PortNADistanceUnit(short portNum, tagNADistanceUnit* *pVal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_PortNADistanceUnit(short portNum, tagNADistanceUnit newVal);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFixturing4</td>
</tr>
</tbody>
</table>
### PortExtState Property

**Description**
Turns Port Extension ON or OFF for all ports on the channel.

**VB Syntax**
`fixture.PortExtState = value`

**Variable**
- **(Type)** - Description
- **fixture** - A Fixturing (object)
- **value** - (Boolean)

**Examples**
- `fixture.PortExtState = 0`  
  `value = fixture.PortExtState`

**Return Type**
Boolean

**Default**
False

**C++ Syntax**
- `HRESULT get_PortExtState(VARIANT_BOOL *pVal)`
- `HRESULT put_PortExtState(VARIANT_BOOL newVal)`

**Interface**
IFixturing
PortExtUse1 Property

Description
Sets and returns the Use1 ON/OFF state for the use of the PortLoss1 and PortFreq1 values for the specified port number.

**Note:** This command affects ALL measurements on the channel.

VB Syntax

```vbnet
fixture.PortExtUse1(port) = value
```

Variable

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fixture</td>
<td>A Fixturing (object)</td>
</tr>
<tr>
<td>port</td>
<td>(Integer)  Port number to receive Use1 ON / OFF state.</td>
</tr>
<tr>
<td>value</td>
<td>(Boolean)</td>
</tr>
</tbody>
</table>

- False - Turns Use1 OFF
- True - Turns Use1 ON

Return Type

Boolean

Default

False

Examples

```vbnet
fixture.PortExtUse1(2) = False 'Write
value = fixture.PortExtUse1(1) 'Read
```

C++ Syntax

```c
HRESULT get_PortExtUse1(short port VARIANT_BOOL *pVal)
HRESULT put_PortExtUse1(short port VARIANT_BOOL newVal)
```

Interface

IFixturing
PortExtUse2 Property

**Description**
Sets and returns the Use2 ON/OFF state for the use of the PortLoss2 and PortFreq2 values for the specified port number.

**Note:** This command affects ALL measurements on the channel.

**VB Syntax**
```vbnet
fixture.PortExtUse2(port) = value
```

**Variable (Type) - Description**
- `fixture`  A Fixturing (object)
- `port`  (Integer) Port number to receive Use2 ON / OFF state.
- `value`  (Boolean)
  - False - Turns Use1 OFF
  - True - Turns Use1 ON

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
fixture.PortExtUse2(2) = False 'Write
value = fixture.PortExtUse2(1) 'Read
```

**C++ Syntax**
```csharp
HRESULT get_PortExtUse2(short port VARIANT_BOOL *pVal)
HRESULT put_PortExtUse2(short port VARIANT_BOOL newVal)
```

**Interface**
IFixturing
PortFreq1 Property

Description
Sets and returns Frequency1 value for the specified port number.

**Note:** This command affects ALL measurements on the channel.

VB Syntax

```
fixture.PortFreq1(port) = value
```

Variable

- **fixture** (Type) - Description
  A Fixturing (object)
- **port** (Integer) - Port number to receive extrapolated loss.
- **value** (Double) - Frequency1 value. Choose a frequency within the frequency span of the VNA.

Return Type

Double

Default

1 GHz

Examples

```
fixture.PortFreq1(2) = naFix2PD_USER 'Write
value = fixture.PortFreq1(1) 'Read
```

C++ Syntax

```
HRESULT get_PortFreq1(short port double *pVal)
HRESULT put_PortFreq1(short port double newVal)
```

Interface

IFixturing
PortFreq2 Property

**Description**  
Sets and returns Frequency2 value for the specified port number.

**Note:** This command affects ALL measurements on the channel.

**VB Syntax**  
`fixture.PortFreq2(port) = value`

**Variable**  

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fixture</code></td>
<td>A Fixturing (object)</td>
</tr>
<tr>
<td><code>port</code></td>
<td>(Integer) Port number to receive extrapolated loss.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Frequency2 value. Choose a frequency within the frequency span of the PNA.</td>
</tr>
</tbody>
</table>

**Return Type**  
Double

**Default**  
1 GHz

**Examples**  
`fixture.PortFreq2(2) = 10E9 'Write`
`value = fixture.PortFreq2(1) 'Read`

**C++ Syntax**  
`HRESULT get_PortFreq2(short port double *pVal)`
`HRESULT put_PortFreq2(short port double newVal)`

**Interface**  
IFixturing
**PortIFBandwidth Property**

**Description**
Sets or returns the IF Bandwidth per port property of the Segment. Enable the IF bandwidth per port property using the `PortIFBandwidthOption Property` of the Segments collection.

**VB Syntax**
```vbnet
seg.PortIFBandwidth(port) = value
```

**Variable (Type) - Description**
- **seg** A `Segment (object)`
- **port** (Long) Port number.
- **value** (Double) IF bandwidth.

**Return Type**
Double

**Default**
1000

**Examples**
```vbnet
seg.PortIFBandwidth(1) = 1000 'Write
value = seg.PortIFBandwidth(1) 'Read
```

**C++ Syntax**
```cpp
HRESULT get_PortIFBandwidth(long port, double* pVal);
HRESULT put_PortIFBandwidth(long port, double pVal);
```

**Interface**
ISegment3
**PortIFBandwidthOption Property**

**Description**
Enables or disables IF Bandwidth per port per segment. Value is set using the `PortIFBandwidth` Property of a Segment.

**VB Syntax**

```vb
segs.PortIFBandwidthOption = value
```

**Variable**

- **segs** - A `Segments` collection (`object`)
- **value** - (boolean)
  - **True** - Enable port IF bandwidth.
  - **False** - Disable port IF bandwidth.

**Return Type**
Boolean

**Default**
False

**Examples**

```vb
segs.PortIFBandwidthOption = True 'Write
IFBWstate = segs.PortIFBandwidthOption 'Read
```

**C++ Syntax**

```cpp
HRESULT get_PortIFBandwidthOption(VARIANT_BOOL* pVal)

HRESULT put_PortIFBandwidthOption(VARIANT_BOOL pVal)
```

**Interface**
ISegments6
## PortLabel Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the label on the calibration kit Port for the calibration wizard.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>calKit.PortLabel (portNum) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>calKit</code></td>
<td>A CalKit <em>(object)</em></td>
</tr>
<tr>
<td><code>portNum</code></td>
<td><em>(long integer)</em> - number of the port to be labeled. Choose either 1 or 2</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(string)</em> - Label that is visible in the calibration wizard.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Depends on the Cal Kit.</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>calKit.PortLabel = &quot;MyCalKit&quot; 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>kitLabel = calKit.PortLabel 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_PortLabel(long port, BSTR *pVal)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_PortLabel(long port, BSTR newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalKit</td>
</tr>
</tbody>
</table>
PortLogic Property

Description
Sets and returns the logic mode of data ports A-H on the HandlerIO connector. Port C of the Handler IO is connected internally to the Port C of the Aux IO connector. Therefore, it will have the same logic mode.

VB Syntax
handler.PortLogic = value

Variable (Type) - Description
handler (object) - A HandlerI/O object
value (enum as NaRearPanelIOLogic) - Choose from:

0 - naPositiveLogic - When a value of one is written, the associated line goes High
1 - naNegativeLogic - When a value of one is written, the associated line goes Low

For ports that are in output (write) mode, a change in logic causes the output lines to change state immediately. For example, Low levels change immediately to High levels.

For ports that are in input (read) mode (C,D,E only), a change in logic will be reflected when data is read from that port. For example, if a line read 0, the next read after a logic change will read 1.

Return Type
Long Integer

Default
1 - naNegativeLogic

Examples
handler.PortLogic = value 'Write
value = handler.PortLogic 'Read

C++ Syntax
HRESULT put_PortLogic( tagNARearPanelIOLogic Mode );

HRESULT get_PortLogic( tagNARearPanelIOLogic* Mode );

Interface
IHWMaterialHandlerIO
### PortLoss1 Property

**Description**
Sets and returns the Loss1 value for the specified port number.

**Note:** This command affects ALL measurements on the channel.

**VB Syntax**
```
fixture.PortLoss1(port) = value
```

**Variable**

- **fixture** (Type) - Description
  A Fixturing (object)
- **port** (Integer) - Port number to receive Loss value
- **value** (Double) - Loss1 value in dB. Choose a value between -90 and 90.

**Return Type**
Double

**Default**
0

**Examples**
```
fixture.PortLoss1(2) = .002 'Write
value = fixture.PortLoss1(1) 'Read
```

**C++ Syntax**
```
HRESULT get_PortLoss1(short port double *pVal)
HRESULT put_PortLoss1(short port double newVal)
```

**Interface**
IFixturing
# PortLoss2 Property

**Description**  Sets and returns the Loss2 value for the specified port number.

**Note:** This command affects ALL measurements on the channel.

**VB Syntax**  

```vbnet
fixture.PortLoss2(port) = value
```

**Variable**  

- **fixture**  A Fixturing (object)
- **port**  (Integer)  Port number to receive Loss value
- **value**  (Double)  Loss2 value in dB. Choose a value between -90 and 90.

**Return Type**  Double

**Default**  0

**Examples**  

```vbnet
gfixture.PortLoss2(2) = .002  'Write
value = fixture.PortLoss2(1)  'Read
```

**C++ Syntax**  

```c
HRESULT get_PortLoss2(short port double *pVal)
HRESULT put_PortLoss2(short port double newVal)
```

**Interface**  IFixturing
# PortLossDC Property

**Description**
Sets and returns the Loss value at DC for the specified port number.

Learn about Loss compensation values.

**Note:** This command affects ALL measurements on the channel.

**VB Syntax**

```vbnet
fixture.PortLossDC(port) = value
```

**Variable**

- **(Type)** - Description
- **fixture** - A Fixturing (object)
- **port** (Integer) - Port number to receive Loss value at DC.
- **value** (Double) - Loss value in ohms. Choose a value between -90 and 90

**Return Type**
Double

**Default**
0

**Examples**

```vbnet
fixture.PortLossDC(2) = .002  'Write
value = fixture.PortLossDC(1)  'Read
```

**C++ Syntax**

```csharp
HRESULT get_PortLossDC(short port double *pVal)
HRESULT put_PortLossDC(short port double newVal)
```

**Interface**
IFixturing
PortMatching_C Property

Description
Sets and returns the value for the 'C' (Capacitance) circuit element for a port matching circuit model to simulate on port 'n'. The port matching circuit model is selected using PortMatchingCktModel. You can specify C, C1, or C2 based on the selected port matching circuit model. Setting a value not used by the selected circuit will have no affect. Learn more.

There are three steps to set up a port matching circuit model simulation on a specified port:

1. Select the port matching circuit model to simulate using PortMatchingCktModel.
2. Set the values for R (Resistance), G (Conductance), C (Capacitance), and L (Inductance) corresponding to the selected port matching circuit model.
3. Turn the feature on using PortMatchingState to simulate the circuit and compute the measurement as if the circuit were attached to the port.

Note: This command affects ALL measurements on the channel.

VB Syntax
```
fixture.PortMatching_C(port) = value
```

Variable (Type) - Description
- `fixture` (A Fixturing (object))
- `port` (Integer) Port number to receive capacitance value
- `value` (Double) Capacitance value in farads. Choose a value between -1E18 to 1E18.

Return Type
Double

Default
0

Examples
```
fractionalPortMatching_C(2) = .00002 'Write
value = fixture.PortMatching_C(1) 'Read
```

C++ Syntax
```
HRESULT get_PortMatching_C(short port double *pVal)
HRESULT put_PortMatching_C(short port double newVal)
```

Interface
IFixturing7
PortMatching_G Property

Description
Sets and returns the value for the 'G' (Conductance) circuit element for a port matching circuit model to simulate on port 'n'. The port matching circuit model is selected using PortMatchingCktModel. You can specify G, G1, or G2 based on the selected port matching circuit model. Setting a value not used by the selected circuit will have no affect. Learn more.

There are three steps to set up a port matching circuit model simulation on a specified port:

1. Select the port matching circuit model to simulate using PortMatchingCktModel.
2. Set the values for R (Resistance), G (Conductance), C (Capacitance), and L (Inductance) corresponding to the selected port matching circuit model.
3. Turn the feature on using PortMatchingState to simulate the circuit and compute the measurement as if the circuit were attached to the port.

Note: This command affects ALL measurements on the channel.

VB Syntax
```
fixture.PortMatching_G(port) = value
```

Variable (Type) - Description
- `fixture` A Fixturing (object)
- `port` (Integer) Port number to receive conductance value.
- `value` (Double) Conductance value in siemens. Choose a value between -1E18 and 1E18.

Return Type Double
Default 0

Examples
```
fixture.PortMatching_G = .002 'Write
value = fixture.PortMatching_G 'Read
```

C++ Syntax
```
HRESULT get_PorMatching_G(short port double *pVal)
HRESULT put_PorMatching_G(short port double newVal)
```

Interface IFixturing7
PortMatching_L Property

Description
Sets and returns the value for the 'L' (Inductance) circuit element for a port matching circuit model to simulate on port 'n'. The port matching circuit model is selected using PortMatchingCktModel. You can specify L, L1, or L2 based on the selected port matching circuit model. Setting a value not used by the selected circuit will have no affect. Learn more.

There are three steps to set up a port matching circuit model simulation on a specified port:

1. Select the port matching circuit model to simulate using PortMatchingCktModel.
2. Set the values for R (Resistance), G (Conductance), C (Capacitance), and L (Inductance) corresponding to the selected port matching circuit model.
3. Turn the feature on using PortMatchingState to simulate the circuit and compute the measurement as if the circuit were attached to the port.

**Note:** This command affects ALL measurements on the channel.

**VB Syntax**

```vbnet
fixture.PortMatching_L(port) = value
```

**Variable Description**

- **fixture** (Fixturing object)
- **port** (Integer) Port number to receive inductance value
- **value** (Double) Inductance value in henries. Choose a value between -1E18 and 1E18.

**Return Type**
Double

**Default**
0

**Examples**

```vbnet
fixture.PortMatching_L = .002 'Write
value = fixture.PortMatching_L 'Read
```

**C++ Syntax**

```cpp
HRESULT get_PortMatching_L(short port double *pVal)
HRESULT put_PortMatching_L(short port double newVal)
```

**Interface**
IFixturing7
## PortMatching_R Property

**Description**
Sets and returns the value for the 'R' (Resistance) circuit element for a port matching circuit model to simulate on port 'n'. The port matching circuit model is selected using `PortMatchingCktModel`. You can specify R, R1, or R2 based on the selected port matching circuit model. Setting a value not used by the selected circuit will have no affect. Learn more.

There are three steps to set up a port matching circuit model simulation on a specified port:

1. Select the port matching circuit model to simulate using `PortMatchingCktModel`.
2. Set the values for R (Resistance), G (Conductance), C (Capacitance), and L (Inductance) corresponding to the selected port matching circuit model.
3. Turn the feature on using `PortMatchingState` to simulate the circuit and compute the measurement as if the circuit were attached to the port.

**Note:** This command affects ALL measurements on the channel.

### VB Syntax

```
fixture.PortMatching_R(port) = value
```

### Variable (Type) - Description

- **fixture**
  A Fixturing (object)
- **port**
  (Integer) Port number to receive resistance value.
- **value**
  (Double) Resistance value in ohms. Choose a value between -1E18 and 1E18.

### Return Type

Double

### Default

0

### Examples

```
fixture.PortMatching_R = .1  'Write
value = fixture.PortMatching_R  'Read
```

### C++ Syntax

```
HRESULT get_PortMatching_R(short port double *pVal)
HRESULT put_PortMatching_R(short port double newVal)
```

### Interface

IFixturing7
### PortMatchingCktModel Property

**Description:** Sets and returns the Port Matching circuit model for the specified port number.

**Note:** This command affects ALL measurements on the channel.

**VB Syntax**

```
fixture.PortMatchingCktModel(port) = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fixture</td>
<td>A Fixturing (object)</td>
</tr>
<tr>
<td>port</td>
<td>(Integer) Port number to receive circuit model.</td>
</tr>
<tr>
<td>value</td>
<td>(Enum as NAFixturingPortMatchCkt) Circuit model. Choose from</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>Circuit Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>naFixPMC_SLPC Series L - Parallel C</td>
</tr>
<tr>
<td>1</td>
<td>naFixPMC_PCSL Parallel C - Series L</td>
</tr>
<tr>
<td>2</td>
<td>naFixPMC_PLSC Parallel L - Series C</td>
</tr>
<tr>
<td>3</td>
<td>naFixPMC_SCPL Series C - Parallel L</td>
</tr>
<tr>
<td>4</td>
<td>naFixPMC_PLPC Parallel L - Parallel C</td>
</tr>
<tr>
<td>5</td>
<td>naFixPMC_USER Load S2P file - also set filename to load with strPortMatch_S2PFile Property</td>
</tr>
<tr>
<td>6</td>
<td>naFixPMC_NONE No circuit model</td>
</tr>
<tr>
<td>7</td>
<td>naFixPMC_SCPC Series C - Parallel C</td>
</tr>
<tr>
<td>8</td>
<td>naFixPMC_PCSC Parallel C - Series C</td>
</tr>
<tr>
<td>9</td>
<td>naFixPMC_SLPL Series L - Parallel L</td>
</tr>
<tr>
<td>10</td>
<td>naFixPMC_PLSL Parallel L - Series L</td>
</tr>
</tbody>
</table>

**Return Type**

Long Integer

**Default**

naFixPMC_NONE

**Examples**

```vbnet
fixture.PortMatchingCktModel(2) = naFixPMC_PLSC 'Write
value = fixture.PortMatchingCktModel(1) 'Read
```
C++ Syntax

HRESULT get_PortMatchingCktModel(short port tagNAFixturingPortMatchCkt *pVal)
HRESULT put_PortMatchingCktModel(short port tagNAFixturingPortMatchCkt newVal)

Interface
IFixturing
PortMatchingState Property

Description  Sets and returns the Port Matching State on the channel.

VB Syntax  

```vbnet
fixture.PortMatchingState = value
```

Variable  

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fixture</td>
<td>A Fixturing (object)</td>
</tr>
<tr>
<td>value</td>
<td>(boolean)</td>
</tr>
<tr>
<td>True</td>
<td>- Turns Port Matching ON</td>
</tr>
<tr>
<td>False</td>
<td>- Turns Port Matching OFF</td>
</tr>
</tbody>
</table>

Return Type  Boolean

Default  False

Examples  

```vbnet
define fixture.PortMatchingState = True 'Write
value = fixture.PortMatchingState 'Read
```

C++ Syntax  

```csharp
HRESULT get_PortMatchingState(VARIANT_BOOL *pVal)
HRESULT put_PortMatchingState(VARIANT_BOOL newVal)
```

Interface  IFixturing
PortMedium Property

Description  Sets and returns the media type of the added fixture or transmission line.

VB Syntax  fixture.PortMedium(port) = value

Variable  (Type) - Description

fixture  A Fixturing Object

port  (Integer) Port Number for which media type is being set.

value  (enum NACalStandardMedium) - Medium of the transmission line of the standard. Choose from:

0 - naCoax - Coaxial Cable

1 - naWaveGuide - Waveguide

Return Type  Enum

Default  0 - naCoax

Examples  fixture.PortMedium(2) = naCoax  'Write

value = fixture.PortMedium(2)  'Read

C++ Syntax  HRESULT get_PortMedium(short portNum, tagNACalStandardMedium* pVal);

HRESULT put_PortMedium(short portNum, tagNACalStandardMedium newVal);

Interface  IFixturing4
PortMode Property

Description
Sets and returns whether Port C or Port D is used for writing or reading data on the Handler IO connector. The Handler IO Port C is connected internally to the Port C of the Aux IO connector. Therefore, the Aux IO connector will have the same input/output mode.

VB Syntax
handler.PortMode (port) = value

Variable (Type) - Description
handler (object) - A Handler I/O object
port (enum as NAMatHandlerPort) Port to be changed. Choose from:
2 - naPortC
3 - naPortD

value (enum as NaPortMode) - Choose from:
0 - naInput - set the port for reading
1 - naOutput - set the port for writing

Return Type
Long Integer

Default
1 - naInput

Examples
handler.PortMode(naPortC) = naInput 'Write
value = handler.PortMode(naPortD) 'Read

C++ Syntax
HRESULT put_PortMode ( tagNAMatHandlerPort Port, tagNAPortMode Mode );

HRESULT get_PortMode ( tagNAMatHandlerPort Port, tagNAPortMode* Mode );

Interface
IHWMaterialHandlerI0
# PortNoiseEnabled Property

**Description**
Sets and returns the ON/OFF state of allowing noise data to contribute to the uncertainty of a calibration performed using Dynamic Uncertainty. Noise data must also be present for the ports at the time the calibration is performed.

**VB Syntax**

```vbnet
uncertMan.PortNoiseEnabled = value
```

**Variable (Type) - Description**

- `uncertMan` An `UncertaintyManager` Object
- `value` (Boolean) Noise contribution state. Choose from:
  - **True** - Noise uncertainty ON.
  - **False** - Noise uncertainty OFF.

**Return Type**
Boolean

**Default**
True

**Examples**

```vbnet
uncertMan.PortNoiseEnabled = True
```

See example program

**C++ Syntax**

```cpp
HRESULT get_PortNoiseEnabled([out,retval] VARIANT_BOOL* pState);
HRESULT put_PortNoiseEnabled([in] VARIANT_BOOL state);
```

**Interface**
`IUncertaintyManager`
## PortsNeedingDeltaMatch Property

**Description**
Returns the port numbers for which delta match correction is required. 0 (zero) is returned if the Cal does NOT require Delta Match correction for one of the following reasons:

- The Cal does NOT involve Unknown Thru or TRL. You specify this using `ThruCalMethod` Property.
- The Cal DOES involve Unknown Thru or TRL, but the delta match data can be calculated by the Unknown Thru or TRL Cal. Learn how this is possible. However, you can force the Cal to use the Delta Match data from a Cal Set.

**VB Syntax**
```vbnet
value = guided.PortsNeedingDeltaMatch
```

**Variable**
- **Type**: Variant
- **Description**: Variable to store the returned list of port numbers.

**guided**
GuidedCalibration (object)

**Return Type**
Variant

**Default**
Not Applicable

**Examples**
```vbnet
Dim ports As Variant
ports = guided.PortsNeedingDeltaMatch
```

**C++ Syntax**
```cpp
HRESULT get_PortsNeedingDeltaMatch (VARIANT* portList);
```

**Interface**
IGuidedCalibration2
PortNumber Property

Sets and reads the port number for power range data. When two sources are available in combiner mode, refer to the following table.

<table>
<thead>
<tr>
<th>Port Number property of power range object</th>
<th>Power returned for a two port instrument</th>
<th>Power returned for a four port instrument</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Port 1</td>
<td>Port 1</td>
</tr>
<tr>
<td>2</td>
<td>Port 2</td>
<td>Port 2</td>
</tr>
<tr>
<td>3</td>
<td>Src2-Out1(^1)</td>
<td>Port 3</td>
</tr>
<tr>
<td>4</td>
<td>Src2-Out2(^1)</td>
<td>Port 4</td>
</tr>
<tr>
<td>5</td>
<td>Port1-Src2(^1)</td>
<td>Port1-Src2(^1)</td>
</tr>
</tbody>
</table>

\(^1\) Indicates that the port requires option 224 or 423.

**VB Syntax**

`powerRange.PortNumber = value`

**Variable**

(Variable) - Description

`powerRange` A PowerRange (object)

`value` (Integer) Port number for power range data.

**Return Type**

Integer

**Default**

Not Applicable

`powerRange.PortNumber = 3 'Write value = powerRange.PortNumber 'Read`

**C++ Syntax**

HRESULT get_PortNumber(short* port);

HRESULT put_PortNumber(short port);

**Interface**

IPowreRange
## SourceStateProperty

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the ON / OFF state of the source specified by <em>(port)</em>. At the top of the Source Configuration dialog this is the Source State setting.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><em>DIQ</em>.SourceState*(port)* = <em>value</em></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td>DIQ</td>
<td>A Differential I/Q <em>(object)</em></td>
</tr>
<tr>
<td>port</td>
<td><em>(String)</em> Source port name. Use SourcePortNames to read a list of valid source ports.</td>
</tr>
<tr>
<td>value</td>
<td><em>(Enum as NAPortState)</em> Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>naPortAUTO</strong> (or 0) - Source power is turned ON at the specified test port when required by the measurement. This is the most common (default) setting. Auto sources are turned OFF when other sources are performing Match Correction sweeps.</td>
</tr>
<tr>
<td></td>
<td><strong>naPortON</strong> (or 1) - Source power is ALWAYS ON, regardless of measurements that are in process. Use this setting to supply source power to a DUT port that always requires power, such as an LO port. This could turn OFF power at another test port. Learn about internal second source restrictions.</td>
</tr>
<tr>
<td></td>
<td><strong>naPortOFF</strong> (or 2) - Source power is never ON, regardless of the measurement requirements. Use this setting to prevent damage to a sensitive DUT test port.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum</td>
</tr>
<tr>
<td>Default</td>
<td><strong>naPortAUTO</strong> (or 0)</td>
</tr>
<tr>
<td>Examples</td>
<td><code>diq.PortState(&quot;Port 2&quot;) = naPortAUTO 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>Value = oDIQ.PortState(&quot;port 2&quot;) 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_PortState(BSTR port, tagNASourceState* SourceState);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_PortState(BSTR port, tagNASourceState SourceState);</td>
</tr>
<tr>
<td>Interface</td>
<td>IDIQ</td>
</tr>
</tbody>
</table>
PortVelocityFactor Property

Description: Sets and returns the Port Extensions Velocity Factor value for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax: `fixture.PortVelocityFactor(port) = value`

Variable (Type) - Description:
- `fixture` A Fixturing (object)
- `port` (Integer) Port number to receive velocity factor value.
- `value` (Double) Velocity Factor value. Choose a number between: 0 and 10 (.66 polyethylene dielectric; .7 PTFE dielectric)

Return Type: Double

Default: 1

Examples:
- `fixture.PortVelocityFactor(2) = .6` 'Write
- `value = fixture.PortVelocityFactor(1)` 'Read

C++ Syntax:
- `HRESULT get_PortVelocityFactor(short port double *pVal)`
- `HRESULT put_PortVelocityFactor(short port double newVal)`

Interface: IFixturing4
## PortWGcutFFreqProperty

**Description**
Sets and returns the cuttoff (minimum) frequency of the added waveguide fixture or transmission line.

**VB Syntax**
```
fixture.PortWGcutFFreq(port) = value
```

**Variable**
- **fixture** - A Fixturing Object
- **port** - (Integer) Port Number for which media type is being set.
- **value** - (Double) Cutoff frequency in Hz.

This value is ignored when **PortMedium Property** is set to **COAX** for the same port.

**Return Type**
Double

**Default**
System Media Cutoff Frequency

**Examples**
```
fixture.PortWGcutFFreq(2) = 1e9 'Write
value = fixture.PortWGcutFFreq(2) 'Read
```

**C++ Syntax**
```
HRESULT get_PortWGcutFFreq(short portNum, double *pVal);
HRESULT put_PortWGcutFFreq(short portNum, double newVal);
```

**Interface**
IFixturing4
### POut Property

**Description**
Returns the POut result of the PNOP or PSAT marker search.

- PSAT Out = Marker 2 Y-axis value
- PNOPOut = Marker 4 Y-axis value

**VB Syntax**

```vbnet
pOut = pMark.POut
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pOut</td>
<td>double</td>
<td>Variable to store returned value</td>
</tr>
<tr>
<td>pMark</td>
<td>A PNOP (Object) or PSaturation (Object)</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
Not applicable

**Examples**

```vbnet
pOut = pMark.POut  'Read
```

See example program

**C++ Syntax**

```cpp
HRESULT get_POut(double* pNewVal)
```

**Interface**
IPNOP or IPSaturation
PowerSlope Property

Description: Sets or returns the Power Slope value. Power Slope function increases or decreases the output power over frequency. Units are db/GHz. For example: PowerSlope = 2 will increase the power 2db/1GHz.

VB Syntax: 
```
object.PowerSlope = value
```

Variable: (Type) - Description
- `object` Channel (object)

  or

- `CalSet (object)` - Read-only property

  `value` (double) - Power Slope. Choose any number between -2 and 2.
  No slope = 0

  Return Type: Double
  Default: 0

Examples: 
```
chan.PowerSlope = 2 'Write
pwrslp = chan.PowerSlope 'Read
```

C++ Syntax: 
```
HRESULT get_PowerSlope(double *pVal)
HRESULT put_PowerSlope(double newVal)
```

Interface: IChannel

| CalSet3 |
**PowerAcquisitionDevice Property**

**Description**

Returns the power sensor channel (A or B) that is currently selected for use at a specific frequency.

If `UsePowerSensorFrequencyLimits` is set to False, this property will return the sensor channel last used for a source power calibration. This setting corresponds to the **Use this sensor only** checkbox in the Power Sensor Settings dialog.

When performing an SMC calibration, use `SetPowerAcquisitionDevice Method` to set the power sensor channel.

**VB Syntax**

```
sensor = pwrCal.PowerAcquisitionDevice(dFreq)
```

**Variable**

* (Type) - Description

- `sensor` (enum NAPowerAcquisitionDevice) The currently selected sensor channel for the specified frequency. Choose from:
  - 0 – naPowerSensor_A
  - 1 – naPowerSensor_B

- `pwrCal` A SourcePowerCalibrator (object)
- `dFreq` (double) Frequency (Hz) for the power reading of interest.

**Return Type**

enum NAPowerAcquisitionDevice

**Default**

Not Applicable

**Examples**

```
selectedSensor = pwrCal.PowerAcquisitionDevice(1.E9) 'Write
```

**C++ Syntax**

```
HRESULT get_PowerAcquisitionDevice(double dFreq, tagNAPowerAcquisitionDevice* enumAcqDevice);
```

**Interface**

ISourcePowerCalibrator2
## PowerAsFixture Property

**Description**
Sets and reads the state of power offset as a fixture with True Mode balanced measurements. Learn more about iTMSA power and power offset.

**VB Syntax**
```
balStim.PowerAsFixture = value
```

**Variable (Type) - Description**
- **balStim** A BalancedStimulus (object)
- **value** (Boolean) State of power offset as a fixture.

- **False** Offset is applied but is NOT included as a fixture in the output calculations.
- **True** Offset is applied and included as a fixture in the output calculations.

**Return Type** Boolean

**Default** False

**Examples**
```
balStim.PowerAsFixture = False 'Write
var = balStim.PowerAsFixture 'Read
```

**C++ Syntax**
```
HRESULT get_PowerAsFixture (VARIANT_BOOL *bVal)
HRESULT put_PowerAsFixture (VARIANT_BOOL bVal)
```

**Interface** IBalancedStimulus
**PowerCalibrationLevel Property**

**Description**
Set and read the power level at which to perform the source cal during an Enhanced Power Cal or during the power cal portion of some Noise Figure Cals.

**VB Syntax**
```
object.PowerCalibrationLevel (port) = value
```

**Variable**
- **Type** - Description
  - `object`: A GuidedCalibration (Object)
  - `port`: (Long) VNA Port number to connect the power sensor.
  - `value`: (Double) - Power level in dB. Choose a value from +30 to (-30).

**Return Type**
Double

**Default**
0

**Examples**
```
cal.PowerCalibrationLevel(1) = -5 'Write
pLevel = nfx.PowerCalibrationLevel(1) 'Read

See enhanced power cal example
```

**C++ Syntax**
```
HRESULT get_PowerCalibrationLevel(long port, double* pVal)
HRESULT put_PowerCalibrationLevel(long port, double* pVal)
```

**Interface**
IGuidedCalibration6
### PowerCalibrationPowerLevel Property

**Description**
Set and read the power level at which to perform the source cal during an Guided Power Cal or during an NFX Cal.

When used with Guided Power Cal, first enable a power cal using PerformPowerCalibration Property.

**VB Syntax**
```
object.PowerCalibrationPowerLevel (port) = value
```

**Variable (Type) - Description**
- `object`: A GuidedCalibration (Object)
- `port`: (Long) VNA Port number to connect the power sensor.
- `value`: (Double) - Power level in dB. Choose a value from +30 to (-30).

**Return Type**
Double

**Default**
0

**Examples**
```
cal.PowerCalibrationPowerLevel(1) = -5 'Write
pLevel = nfx.PowerCalibrationPowerLevel(1) 'Read
```

**C++ Syntax**
```
HRESULT get_PowerCalibrationPowerLevel(long port, double* pVal)
HRESULT put_PowerCalibrationPowerLevel(long port, double* pVal)
```

**Interface**
IGuidedCalibration6
### PowerCalRange Property

<table>
<thead>
<tr>
<th>Description</th>
<th>This property returns a handle to the PowerCalRange object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>IndependentPwrrCalPort. <strong>PowerCalRange</strong>()</td>
</tr>
<tr>
<td>Variable (Type) - Description</td>
<td>A IndependentPowerCalibrationPort (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>IPowerCalRange</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Example</td>
<td>CalibrateAllChannels.IndependentPowerCalibrationPort(3).PowerCal</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT PowerCalRange( IPowerCalRange** value);</td>
</tr>
<tr>
<td>Interface</td>
<td>IIndependentPowerCalibrationPort</td>
</tr>
</tbody>
</table>
PowerForBestAccuracy Property

Description
Returns the power level associated with the best accuracy for a specific power meter.

VB Syntax
plevel = pwrSensor.PowerForBestAccuracy

Variable
(Type) - Description
plevel (Double) - Frequency in Hertz.
pwrSensor A PowerSensorUncertainty (Object)

Return Type
(Double) Power in dBm.

Default
Not Applicable

Examples
plevel = pwrSensor.PowerForBestAccuracy

C++ Syntax
HRESULT PowerForBestAccuracy(double *plevel);

Interface
IPowerSensorUncertainty
### PowerLevel Property

#### Description
Set and read the power level at which to perform the Source Power Cal portion of a Gain Compression Calibration or a SweptIMD Cal.

#### VB Syntax
```
object.PowerLevel = value
```

#### Variable (Type) - Description
- `object` A GainCompressionCal (object)
- `A SweptIMDCal (object)`
- `value` (Double) - Power level in dB. Choose a value from +30 to (-30). Learn about choosing a power level.

#### Return Type
Double

#### Default
0

#### Examples
```
gca.PowerLevel = -5 'Write
pLevel = imd.PowerLevel 'Read
```

#### C++ Syntax
```
HRESULT get_PowerLevel(double* pVal)

HRESULT put_PowerLevel(double newVal)
```

#### Interface
- IGainCompressionCal
- ISweptIMDCal
PowerLevel (Cal All) Property

Description
Sets and returns the power level at which a Cal All calibration is to be performed.

VB Syntax
\[ \text{calAll}.\text{PowerLevel} \ (\text{port}) = \text{value} \]

Variable
(Type) - Description
- \text{calAll}: A \text{CalibrateAllChannels (object)}
- \text{port}: (Long) Source port number.
- \text{value}: (Double) Power level at which the calibration is to be performed.

Return Type
Double

Default

Examples
\[ \text{calAll}.\text{PowerLevel} = 0 \ '\text{Power Level of cal} \]
\[ \text{value} = \text{calAll}.\text{PowerLevel} \ '\text{returns the power level of the cal} \]

C++ Syntax
HRESULT get_PowerLevel (long port, double
\[ \text{val}) \];
HRESULT put_PowerLevel long port, double* newVal);

Interface
ICalibrateAllChannels
# PowerMax Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the maximum power level for Safe Mode.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>RxLevel.PowerMax(srcPort) = value</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>RxLevel</code></td>
<td>A <code>ReceiverLeveling</code> Object</td>
</tr>
<tr>
<td><code>srcPort</code> (Long Integer)</td>
<td>Source port for which to set the max power for Receiver Leveling.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use <code>chan.getPortNumber</code> to translate the string into a port number. To learn more see <a href="#">Remotely Specifying a Source Port</a>.</td>
</tr>
<tr>
<td><strong>value</strong> (Double)</td>
<td>Maximum power level in dB.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>(Double)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>+ 30</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td></td>
</tr>
<tr>
<td><code>rxLevel.PowerMax (1) = 25</code></td>
<td>Write</td>
</tr>
<tr>
<td><code>value = rxLevel.PowerMax 2</code></td>
<td>Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td></td>
</tr>
<tr>
<td>HRESULT <code>get_PowerMax(long port, double* pVal);</code></td>
<td></td>
</tr>
<tr>
<td>HRESULT <code>put_PowerMax(long port, double newVal);</code></td>
<td></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IReceiverLevelingConfiguration</td>
</tr>
</tbody>
</table>
## PowerMeterChannel Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Identifies which channel of the power meter the power sensor is connected to.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan = powerSensor.PowerMeterChannel</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>chan</code></td>
<td><code>(enum NAPowerAcquisitionDevice) – Power meter channel identifier for sensor. Choose from:</code></td>
</tr>
<tr>
<td></td>
<td>0 – <code>naPowerSensor_A</code></td>
</tr>
<tr>
<td></td>
<td>1 – <code>naPowerSensor_B</code></td>
</tr>
<tr>
<td><code>pwrSensor</code></td>
<td>A <code>PowerSensor</code> (Object) or A <code>PowerSensorAsReceiver</code> (Object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>NAPowerAcquisitionDevice</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meterChannel = pwrSensor.PowerMeterChannel</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT PowerMeterChannel(tagNAPowerAcquisitionDevice *pSensor);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPowerSensor</td>
</tr>
<tr>
<td></td>
<td>IPowerSensorAsReceiver</td>
</tr>
</tbody>
</table>
# PowerMeterGPIBAddress Property  

**About Source Power Cal**

<table>
<thead>
<tr>
<th>Description</th>
<th>This command is replaced with PowerMeterInterface Object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the GPIB address of the power meter that will be referenced by the SourcePowerCalibrator object.</td>
<td></td>
</tr>
<tr>
<td>When performing a source power cal, the VNA will search VISA interfaces that are configured in the Keysight IO Libraries on the VNA.</td>
<td></td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
global powerCalibrator
powerCalibrator.PowerMeterGPIBAddress = value
```

**Variable (Type) - Description**

- `powerCalibrator` (**object**) - A SourcePowerCalibrator (object)
- `value` (**long integer**) – Power meter GPIB address. Choose any number between 0 and 30.

**Return Type**

Long integer

**Default**

13

**Examples**

```vbnet
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.PowerMeterGPIBAddress = 13 'Write
pwrMtrAddress = powerCalibrator.PowerMeterGPIBAddress 'Read
```

**C++ Syntax**

```c++
HRESULT put_PowerMeterGPIBAddress(long newVal);
HRESULT get_PowerMeterGPIBAddress(long *pVal);
```

**Interface**

ISourcePowerCalibrator
### PowerMin Property

**Description**
Sets and returns the minimum power level for Safe Mode.

**VB Syntax**
```
RxLevel.PowerMin(srcPort) = value
```

**Variable**
- **RxLevel** - Description: A ReceiverLeveling Object
- **srcPort** - Description: (Long Integer) Source port for which to set the min power for Receiver Leveling.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

**value**
- (Double) Minimum power level in dB.

**Return Type**
- (Double)

**Default**
- -95 dB

**Examples**
```
rxLevel.PowerMin (1) = -50 ' Write
value = rxLevel.PowerMin 2' Read
```

**C++ Syntax**
```
HRESULT get_PowerMin(long port, double* pVal);
HRESULT put_PowerMin(long port, double newVal);
```

**Interface**
- IReceiverLevelingConfiguration
PowerMtrReadingUncertainty Property

Description: Returns the power uncertainty associated with the specific power meter at the specified frequency and power.

VB Syntax: `pwrUnc = pwrSensor.PowerMtrReadingUncertainty(freq,power)`

Variable (Type) - Description:
- `pwrUnc` (Double): Variable to store the returned power meter reading uncertainty.
- `pwrSensor` (A PowerSensorUncertainty (Object))
- `freq` (Double): Frequency (Hz).
- `power` (Double): Power (dBm).

Return Type: Double (Watts)
Default: Not Applicable

Examples: `pwrUnc = pwrSensor.PowerMtrReadingUncertainty(10e9,0.0)`

C++ Syntax: `HRESULT PowerMtrReadingUncertainty(double freq, double power, double *pwrUnc);`

Interface: IPowerSensorUncertainty
### PowerOffset Property

**Description**  
Sets and returns the power level offset value.

**VB Syntax**  
```vb
RxLevel.PowerOffset(srcPort) = value
```

**Variable**  
**(Type) - Description**
- `RxLevel`  
  A `ReceiverLeveling` Object
- `srcPort`  
  (Long Integer) Source port for which to set the offset power for Receiver Leveling.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port

**value**  
(Double) Power level Offset in dB.

**Return Type**  
(Double)

**Default**  
0 dB

**Examples**  
```vb
rxLevel.PowerOffset (1) = 10 ' Write
value = rxLevel.PowerOffset 2' Read
```

**C++ Syntax**  
```c++
HRESULT get_PowerOffset(long port, double* pVal);
HRESULT put_PowerOffset(long port, double newVal);
```

**Interface**  
`IReceiverLevelingConfiguration`
PowerOffset (Cal All) Property

Description
Sets and returns the power offset value for a Cal All calibration.

Power Offset provides a method of compensating port power for added attenuation or amplification in the source path. The result is that power at the specified port reflects the added components.

VB Syntax
`calAll.PowerOffset (port) = value`

Variable (Type) - Description
- `calAll` A `CalibrateAllChannels` object
- `port` (Long) Source port number.
- `value` (Double) Power offset value in dB for a Cal All calibration.

- For amplification, use positive offset.
- For attenuation, use negative offset.

Return Type
Double

Default
0 (zero dB)

Examples
`calAll.PowerOffset = 0 'Power Level of cal`
`value = calAll.PowerOffset 'returns the power offset of the cal`

C++ Syntax
`HRESULT get_PowerOffset (long port, double val);`
`HRESULT put_PowerOffset long port, double* newVal);`

Interface
CalibrateAllChannels
## PowerOnDuringRetraceMode Property

<table>
<thead>
<tr>
<th>Description</th>
<th>For single-band frequency or segment sweeps ONLY, specify whether to turn RF power ON or OFF during a retrace. This setting remains until changed using this command, or until the hard drive is changed or reformatted.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>pref.PowerOnDuringRetraceMode = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>pref</code></td>
<td>A Preferences (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Enum) - Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <code>naRetracePowerMode_Auto</code> Power is left ON during retrace of single-band frequency or segment sweeps ONLY.</td>
</tr>
<tr>
<td></td>
<td>1 - <code>naRetracePowerMode_OFF</code> Power is turned OFF during retrace of single-band frequency or segment sweeps ONLY.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum</td>
</tr>
<tr>
<td>Default</td>
<td>0 - <code>naRetracePowerMode_Auto</code></td>
</tr>
<tr>
<td>Examples</td>
<td><code>pref.PowerOnDuringRetraceMode = naPowerSweepRetraceMode_Start </code>Write <code>psMode = pref.naPowerOnDuringRetraceMode_Start </code>Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_PowerOnDuringRetraceMode (tagNARetracePowerMode* preference); HRESULT put_PowerOnDuringRetraceMode (tagNARetracePowerMode val)</td>
</tr>
<tr>
<td>Interface</td>
<td>IPreferences4</td>
</tr>
</tbody>
</table>
### PowerRangeType Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the type of power range data (specified or typical) to be returned.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>powerRange.PowerRangeType = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><em>powerRange</em></td>
<td>A <strong>PowerRange</strong> <em>(object)</em></td>
</tr>
<tr>
<td><em>value</em></td>
<td>(Enum NAType) Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 or <strong>naSpecifiedPower</strong></td>
</tr>
<tr>
<td></td>
<td>1 or <strong>naTypicalPower</strong></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>powerRange.PowerRangeType = naSpecifiedPower</code> <em>(Write)</em></td>
</tr>
<tr>
<td></td>
<td><code>value = powerRange.PowerRangeType</code> <em>(Read)</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_PowerRangeType(enum NAType *val);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_PowerRangeType(enum NAType val);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPowreRange</td>
</tr>
</tbody>
</table>
### PowerSensorCalKitType Property

**Description**
Set and read the cal kit to be used for calibrating at the reference plane when the power sensor connector is different from the DUT port.

When used with Guided Power Cal, first enable a power cal using PerformPowerCalibration Property.

**VB Syntax**

```
object.PowerSensorCalKitType (n) = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>A GuidedCalibration (object)</td>
</tr>
<tr>
<td>n</td>
<td>(Long) VNA port number for which cal kit is specified.</td>
</tr>
<tr>
<td>value</td>
<td>(String) - Cal Kit. Use CompatibleCalKits Property to return a list of valid cal kits.</td>
</tr>
</tbody>
</table>

**Return Type**
String

**Default**
Not Applicable

**Examples**

```vbnet
gguided.PowerSensorCalkitType(1) = "85052B" 'Write
ctype =guided.PowerSensorCalkitType(1) 'Read
```

**C++ Syntax**

```
HRESULT get_PowerSensorCalkitType(long port, BSTR* Val)
HRESULT put_PowerSensorCalkitType(long port, BSTR  newVal)
```

**Interface**
IGuidedCalibration6
### PowerSensorCalkitType Property

**Description**  
Set and read the cal kit to be used for calibrating at the adapter when the power sensor connector is different from the DUT. Use `PowerSensorConnectorType` to specify the connector type of the adapter.

**VB Syntax**  
```vbnet
object.PowerSensorCalkitType = value
```

**Variable**  
*(Type)* - Description

- `object`  
  A `GainCompressionCal` (object)
  
  A `SweptIMDCal` (object)
  
  A `GuidedCalibrationPowerSensor` (object)

- `value`  
  *(String)* - Cal Kit. Use `CompatibleCalKits` Property to return a list of valid cal kits.

**Return Type**  
String

**Default**  
Not Applicable

**Examples**  
```vbnet
gca.PowerSensorCalkitType = "85052B" 'Write
ctype = imd.PowerSensorCalkitType 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_PowerSensorCalkitType(BSTR* Val)
HRESULT put_PowerSensorCalkitType(BSTR newVal)
```

**Interface**  
`IGainCompressionCal2`

`ISweptIMD`

`IGuidedCalibrationPowerSensor`
## PowerSensorConnectorType Property

**Description**
Set and read the power sensor connector type which is used to perform the Source Power Cal. Use `PowerSensorCalKitType` to specify the Cal Kit to use for the cal.

**VB Syntax**
```vb
object.PowerSensorConnectorType = value
```

**Variable**
- **(Type)** - Description
  - `object` - A `GainCompressionCal` (object)
  - A `SweptIMDCal` (object)
  - A `GuidedCalibrationPowerSensor` (object)

- **value** - (String) - Power sensor connector type. Use `ValidConnectorType Property` to return a list of valid connector types.
  
  Select "Ignored" to NOT compensate for the adapter.

**Return Type**
String

**Default**
Not applicable

**Examples**
```vb
(gca.PowerSensorConnectorType = "APC3.5 male"  'Write
cType = imd.PowerSensorConnectorType  'Read
```

**C++ Syntax**
```cpp
HRESULT get_PowerSensorConnectorType(BSTR* Val)
HRESULT put_PowerSensorConnectorType(BSTR newVal)
```

**Interface**
- `IGainCompressionCal2`
- `ISweptIMDCal`
- `IGuidedPowerCalPowerSensor`
**PowerSensorConnectorType Property**

**Description**
Set and read the power sensor connector type which is used to perform a Power Cal during an S-parameter calibration or during an NFX Cal.

When used with Guided Power Cal, first enable a power cal using PerformPowerCalibration Property.

**VB Syntax**
```vbnet
guided.PowerSensorConnectorType (n) = value
```

**Variable**
*(Type) - Description*

- `guided` A GuidedCalibration *(object)*
- `n` *(Long)* VNA port number to connect power sensor to.
- `value` *(String)* - Power sensor connector type. Use ValidConnectorType Property to return a list of valid connector types.

Set to "Ignored" to NOT compensate for the adapter.

**Return Type**
String

**Default**
Not applicable.

**Examples**
```vbnet
guided.PowerSensorConnectorType(1) = "APC3.5 male" 'Write
cType = guided.PowerSensorConnectorType(1) 'Read
```

**C++ Syntax**
```cpp
HRESULT get_PowerSensorConnectorType (long port, BSTR* val);

HRESULT put_PowerSensorConnectorType (long port, BSTR newVal);
```

**Interface**
IGuidedCalibration6
# PowerSlopeState Property

**Description**
Turns point slope ON or OFF for all measurements on the channel. Use PowerSlope to set the slope value.

**VB Syntax**

```
chan.PowerSlopeState = state
```

**Variable (Type) - Description**

- `chan` *(A Channel (object))*
- `state` *(boolean)*

- **False** - Turns power slope OFF
- **True** - Turns point slope ON

**Return Type**
Boolean

**Default**
False

**Examples**

```
chan.PowerSlopeState = True  'Write
state = chan.PowerSlopeState  'Read
```

**C++ Syntax**

```
HRESULT get_PowerSlopeState(VARIANT_BOOL *pVal)
HRESULT put_PowerSlopeState(VARIANT_BOOL newVal)
```

**Interface**
IChannel18
PowerSpinResolution Property

Description
Sets and returns the resolution of the front panel knob when it is used to adjust Source Power manually.

VB Syntax

\[ disp.PowerSpinResolution = value \]

Variable
\( disp \) - A Display (object)
\( value \) - (Double) - Power level knob resolution. The range of acceptable values is 0.01 to 100.

Return Type
Double

Default
0.1

Examples

\[ app.disp.PowerSpinResolution = 0.01 \text{'Write - Every tick of the front panel knob will change the Power Level by 0.01 dBm.'} \]

\[ \text{resolution} = \text{app.disp.PowerSpinResolution} \text{'Read - Get the current knob resolution.'} \]

C++ Syntax

HRESULT get_PowerSpinResolution(double* resolution)
HRESULT put_PowerSpinResolution(double resolution)

Interface
IDisplay
PowerStep Property

Description: Sets and returns the safe mode power step value.

**VB Syntax**

\[ RxLevel.\text{PowerStep}(srcPort) = value \]

**Variable**

- **RxLevel** (Type) - Description
  - A ReceiverLeveling Object
- **srcPort** (Long Integer) - Source port for which to set the power step for Receiver Leveling.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port

**value** (Double) - Power step in dB.

**Return Type** (Double)

**Default** 1 dB

**Examples**

- `rxLevel.PowerStep (1) = 2` \(\text{'Write}\)
- `value = rxLevel.PowerStep 2` \(\text{'Read}\)

**C++ Syntax**

- `HRESULT get_PowerStep(long port, double* pVal);`
- `HRESULT put_PowerStep(long port, double newVal);`

**Interface** IReceiverLevelingConfiguration
## PowerStepsIn3DSweep Property

**Description**  
Set and read the number of power steps for a 3D sweep.

**VB Syntax**  
`HotS22.PowerStepsIn3DSweep = value`

**Variable**  
- **Type**: ActiveParametersApp (object)
- **value**:  
  - **(Long)** - Number of power steps. The range is 2 to 20001.

**Return Type**  
Long

**Default**  
201

**Examples**  
- `HotS22.PowerStepsIn3DSweep = 201 'Write`
- `value = HotS22.PowerStepsIn3DSweep 'Read`

**C++ Syntax**  
- `HRESULT get_PowerStepsIn3DSweep(long* value)`
- `HRESULT put_PowerStepsIn3DSweep(long value)`

**Interface**  
IActiveChannelSettings
# PowerSweepRetracePowerMode Property

**Description**  
At the end of a power sweep, while waiting to trigger the next sweep, maintain source power at either the start power level or at the stop power level.

**VB Syntax**  
`pref.PowerSweepRetracePowerMode = value`

**Variable**  
- **`pref`** - A Preferences (object)
- **`value`** - (Enum) - Choose from:
  
  0 - `naPowerSweepRetraceMode_Start` - maintain source at start power level.
  
  1 - `naPowerSweepRetraceMode_Stop` - maintain source at stop power level.

**Return Type**  
Enum

**Default**  
0 - `naPowerSweepRetraceMode_Start`

**Examples**  
```vbnet
 pref.PowerSweepRetracePowerMode = naPowerSweepRetraceMode_Start 'Write
psMode = pref.PowerSweepRetracePowerMode 'Read
```

**C++ Syntax**  
```c++
HRESULT get_PowerSweepRetracePowerMode (tagNAPowerSweepRetraceMode* preference);
HRESULT put_PowerSweepRetracePowerMode (tagNAPowerSweepRetraceMode val)
```

**Interface**  
IPreferences3
## PowerTableFilename Property

**Description**  
Loads a file that defines a power table to be used during a SMC Guided Power Cal or Cal All Channels on a mmWave system. This feature is available because power sensors are NOT typically available at mmWave frequencies. [Learn more.](#)

**VB Syntax**  
```
guided.PowerTableFilename (n) = value
```

**Variable**  
*Type* - Description

- `guided`  
  A GuidedCalibration *(object)*

- `n`  
  *(Long)*  
  Source port being calibrated.

- `value`  
  *(String)*  
  Full path and filename of a *.prn file that defines the power table. An error is returned if the file is not found.

**Return Type**  
String

**Default**  
Not applicable.

**Examples**
```
guided.PowerTableFilename(1) = "c:\powertable1.prn"  'Write
value = guided.PowerTableFilename(1)  'Read
```

**C++ Syntax**
```c++
HRESULT get_PowerTableFilename (long port, BSTR* val);
```
```
HRESULT put_PowerTableFilename (long port, BSTR newVal);
```

**Interface**  
IGuidedCalibration9
### PreciseTuningTolerance Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the tuning tolerance for precise tuning.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>obj.PreciseTuningTolerance = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>obj</code></td>
<td>An <code>EmbeddedLO (object)</code> or</td>
</tr>
<tr>
<td></td>
<td>A <code>ConverterEmbeddedLO (object)</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Double)</em> Tuning tolerance in Hz.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td><em>(Double)</em></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1 Hz</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>embedLO.PreciseTuningTolerance = .5</code> 'write'</td>
</tr>
<tr>
<td></td>
<td><code>value = embedLO.PreciseTuningTolerance</code> 'read'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_PreciseTuningTolerance(double* tolerance);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_PreciseTuningTolerance(double tolerance);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IEmbededLO</code></td>
</tr>
</tbody>
</table>
PreferInternalTriggerOnChannelSingle Property

Description
Set and read the preference for the chan.Single trigger behavior. This setting persists until changed.

These preferences are important when performing a Guided calibration, as the VNA uses the chan.Single trigger command to measure standards.

- set PreferInternalTriggerOnChannelSingle = False to use an External trigger sweep to measure a cal standard.
- set PreferInternalTriggerOnChannelSingle = True to use an External sweep for the measurement, but rely on the VNA to send Internal trigger signals for calibrating.

To set this preference for an Unguided Calibration, use PreferInternalTriggerOnUnguidedCal Property

The chan.Single trigger command NEVER respects the Trigger Source = Manual setting. It always switches to Internal for one trigger, then back to Manual, regardless of this preference command.

VB Syntax

```vbnet
pref.PreferInternalTriggerOnChannelSingle = bool
```

Variable (Type) - Description

pref A Preferences object

bool (Boolean) - Choose from:

- 0 - False - the Single trigger property does respect the Trigger Source = External setting. For example, if Trigger source = External, the single trigger method will wait for the External trigger signal and then allow only one sweep.

- 1 - True - the Single trigger command does NOT respect the Trigger Source = External setting. For example, when the Single method is sent, the VNA immediately switches to Internal sweep, responds to one trigger signal, then switches back to External.

Return Type
Boolean

Default
0 - False

Examples

```vbnet
pref.PreferInternalTriggerOnChannelSingle = False 'Write
prefer = pref.PreferInternalTriggerOnChannelSingle 'Read
```

C++ Syntax

```cpp
HRESULT put_PreferInternalTriggerOnChannelSingle( VARIANT_BOOL value )
```
HRESULT get_PreferredInternalTriggerOnChannelSingle(VARIANT_BOOL *bprefSingle)

Interface IPreferences2
## PreferInternalTriggerOnUnguidedCal Property

**Description**: Set and read the preference for the trigger behavior when performing an Unguided calibration.

**VB Syntax**: 
```
pref.PreferInternalTriggerOnUnguidedCal = bool
```

**Variable**
- **pref**: A `Preferences` object
- **bool**: A `Boolean`

**Return Type**: `Boolean`

**Default**: 0 - False

**Examples**
```
pref.PreferInternalTriggerOnUnguidedCal = False 'Write
prefer = pref.PreferInternalTriggerOnUnguidedCal 'Read
```

**C++ Syntax**
```
HRESULT put_PreferInternalTriggerOnUnguidedCal(VARIANT_BOOL bprefUnguided)

HRESULT get_PreferInternalTriggerOnUnguidedCal(VARIANT_BOOL *bprefUnguided)
```

**Interface**: `IPreferences2`

---

**Description**
- **0 - False**: The trigger behavior during an Unguided calibration DOES respect the setting of the `Trigger source` command. For example, during an Unguided Cal, when `Trigger source = External`, the VNA will wait for the External trigger signal and then allow only one sweep.

- **1 - True**: The trigger behavior during an Unguided calibration does NOT respect the `Trigger Source = External` setting. For example, during an Unguided Cal, when `Trigger source = External`, the VNA immediately switches to Internal sweep, measures the standard with one trigger signal, then switches back to External trigger.

**Note**: When `Trigger Source = Manual` during a calibration, the VNA ALWAYS switches to Internal for one trigger to measure a standard, then back to Manual, regardless of this preference command.
PreferSourcePowerCalFromCalset Property

**Description**
Specifies if the source power cal in the calset linked to a measurement cal should be enabled or disabled with that cal.

**VB Syntax**
```
pref.PREFERSOURCEPOWERCALFROMCALSET = bool
```

**Variable (Type) - Description**
- **pref**  A Preferences **(object)**
- **bool**  **(Boolean)** - Choose from:
  - **0 - False**  - Disable source power cal in calset.
  - **1 - True**  - Enable source power cal in calset.

**Return Type**
Boolean

**Default**
0 - False

**Examples**
```
pref.PREFERSOURCEPOWERCALFROMCALSET = False  'Write
prefer = pref.PREFERSOURCEPOWERCALFROMCALSET  'Read
```

**C++ Syntax**
```
HRESULT put_PREFERSOURCEPOWERCALFROMCALSET( VARIANT_BOOL prefPowerCal)
HRESULT get_PREFERSOURCEPOWERCALFROMCALSET( VARIANT_BOOL *prefPowerCal)
```

**Interface**
IPreferences
## PresetMaxFrequency Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the maximum specified frequency of the analyzer. Does not include any over-sweep. See also: MaximumFrequency Property.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.PresetMaxFrequency</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Variable to store the returned value.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.PresetMaxFrequency 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_PresetMaxFrequency(Double *value);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities13</td>
</tr>
</tbody>
</table>
### PresetMinFrequency Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the minimum specified frequency of the analyzer. Does not include any over-sweep. See also: MinimumFrequency Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.PresetMinFrequency</code></td>
</tr>
</tbody>
</table>

**Variable**

- **Value** (Double) Variable to store the returned value.
- **cap** A Capabilities (object)

**Default**

Not Applicable

**Examples**

`value = cap.PresetMinFrequency 'Read`

**C++ Syntax**

```cpp
HRESULT get_PresetMinFrequency(Double *value);
```

**Interface**

ICapabilities13
PresetPowerState Property

Description  Set and return the Preset Power ON/OFF state.

VB Syntax  \textit{\texttt{pref.PresetPowerState = value}}

Variable  \begin{itemize}
\item \texttt{pref} - A Preferences \texttt{(object)}
\item \texttt{value} - (Enum as NAPowerStates) - Choose from:
\end{itemize}

\begin{itemize}
\item \texttt{naPowerON (0)} - Instrument Preset always turns RF power ON.
\item \texttt{naPowerAUTO (1)}:
\begin{itemize}
\item When the current power setting is OFF, leave power OFF at Preset.
\item When the current power setting is ON, turn power ON at Preset.
\end{itemize}
\end{itemize}

Return Type  enum

Default  \texttt{naPowerON (0)}

Examples  \begin{itemize}
\item \texttt{pref.PresetPowerState = naPowerON} \texttt{'}Write\texttt{'}
\item \texttt{pwrState = pref.PresetPowerState} \texttt{'}Read\texttt{'}
\end{itemize}

C++ Syntax  \begin{itemize}
\item \texttt{HRESULT get_PresetPowerState(tagNAPowerStates* pVal)};
\item \texttt{HRESULT put_PresetPowerState(tagNAPowerStates pVal)};
\end{itemize}

Interface  IPreferences11
### PrimaryFrequency Property

**Description**
Sets the pulse repetition frequency (PRF) for ALL internal pulse generators.

This setting is equal to 1/period which is set with `PrimaryPeriod Property`.

**Note:** On the **Pulse Setup dialog**, this command is a "Basic" setting, intended to be used with the 'Auto' selections set to ON.

#### VB Syntax

```vbnet
pulse.PrimaryFrequency = value
```

#### Variable (Type) - Description
- `pulse` A `PulseMeasurementControl (object)`
- `value` (Double) PRF in Hz.

#### Return Type
Double

#### Default
1 kHz

#### Examples

```vbnet
pulse.PrimaryFrequency = 1e4 'Write
value = pulse.PrimaryFrequency 'Read
```

#### C++ Syntax

```cpp
HRESULT get_PrimaryFrequency(double* value);
HRESULT put_PrimaryFrequency(double value);
```

#### Interface
`IPulseMeasurementControl2`
**PrimaryMode Property**

**Description**
Sets and returns the primary (On/Off) setting of the external pulse generator. This setting allows the external pulse generator to set the primary clock frequency for the other pulse generators.

**VB Syntax**
`extPulseGen.PrimaryMode = value`

**Variable**
(Type) - Description
- `extPulseGen` An `ExternalPulseGenerator (object)`
- `value` (Boolean) Primary setting. Choose from:
  - **True** - Use the external pulse generator becomes the primary clock frequency.
  - **False** - Use the internal pulse generator as the primary clock frequency.

**Return Type**
Boolean

**Default**
False

**Examples**
```
extPulseGen.PrimaryMode = True 'Write
primary = extPulseGen.PrimaryMode 'Read
```

**C++ Syntax**
```
HRESULT get_PrimaryMode (VARIANT BOOL *pValue)
HRESULT put_PrimaryMode (VARIANT BOOL newVal)
```

**Interface**
`IExternalPulseGenerator2`
**PrimaryPeriod Property**

**Description**
Sets the period for ALL internal pulse generators.

This setting is equal to 1/PRF which is set with MasterFrequency Property.

*Note:* On the Pulse Setup dialog, this command is a 'Basic setting, which is intended to be used with the 'Auto' selections set to ON.

**VB Syntax**

```vbnet
pulse.PrimaryPeriod = value
```

**Variable (Type) - Description**

- `pulse` A PulseMeasurementControl Object *(object)*
- `value` *(Double)* Period in seconds.

**Return Type**

Double

**Default**

1 msec

**Examples**

```vbnet
pulse.PrimaryPeriod = 1e-4 'Write
value = pulse.PrimaryPeriod 'Read
```

**C++ Syntax**

```cpp
HRESULT get_PrimaryPeriod(double* Period);
HRESULT put_PrimaryPeriod(double Period);
```

**Interface**

IPulseMeasurementControl2
## PrimaryWidth Property

**Description**
Sets the pulse width for ALL internal pulse generators.

*Note:* On the Pulse Setup dialog, this command is a 'Basic setting, which is intended to be used with the 'Auto' selections set to ON.

**VB Syntax**

```vbnet
pulse.PrimaryWidth = value
```

**Variable**

- **pulse** (Type) - Description
  - A PulseMeasurementControl Object (Object)
- **value** (Double) - Pulse width in seconds.

**Return Type**
Double

**Default**
100 microseconds

**Examples**

```vbnet
pulse.PrimaryWidth = 1e-3 'Write

value = pulse.PrimaryWidth 'Read
```

**C++ Syntax**

```cpp
HRESULT get_PrimaryWidth(double* Width);
HRESULT put_PrimaryWidth(double Width);
```

**Interface**
IPulseMeasurementControl2
<table>
<thead>
<tr>
<th>PropertyNames Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
</tr>
<tr>
<td><code>props</code></td>
</tr>
<tr>
<td><code>calAll</code></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
</tr>
<tr>
<td><strong>Default</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
</tr>
</tbody>
</table>
### PropertyNamesByMeasurementClass Property

**Description**  Returns the list of valid property names by measurement class.

> See a list of valid properties and values for each measurement class.

**VB Syntax**  
```vbnet
propNames = calAll.PropertyNamesByMeasurementClass (MeasurementClass)
```

**Variable**  
- **propNames**  (Variant Array) Variable to store the returned property names.
- **calAll**  A `CalibrateAllChannels (object)`

**MeasurementClass**  (String) Measurement class name for which valid names are to be returned.

**Return Type**  Variant Array

**Default**  Not Applicable

**Examples**  
```vbnet
props = calAll.PropertyNamesByMeasurementClass ("SMC")
```

**C++ Syntax**  
```cpp
HRESULT get_PropertyNamesByMeasurementClass(BSTR MeasurementClass, VARIANT* propNames);
```

**Interface**  `CalibrateAllChannels`
## PropertyValue Property

**Description**
Sets and returns the current property value for a specific property name.

See a list of valid properties and values for each measurement class.

Use `PropertyValues` to query a list of valid values.

### VB Syntax
```
calAll.PropertyValue (propName) = value
```

### Variable (Type) - Description
- **calAll**
  - A `CalibrateAllChannels` (object)
- **propName**
  - (String) Property name for which value is to be set or returned.
- **value**
  - (String) Property value.

### Return Type
String

### Default
Not Applicable

### Examples

**Example 1:**
calAll.PropertyValue ("Noise Cal Method") = "Scalar"

**Example 2:**
calAll.PropertyValue ("Enable Extra Power Cals") = "Port 1 Src2,Port3"

**Example 3:**
calAll.PropertyValue ("Port 1 Src2 Cal Power") = "-20"

### C++ Syntax
```
HRESULT get_PropertyValue (BSTR propName, BSTR* propValue);

HRESULT put_PropertyValue (BSTR propName, BSTR propValue);
```

### Interface
`CalibrateAllChannels`
### PropertyValues Property

**Description**
Returns the valid property values for a specific property name.

See a list of valid properties and values for each measurement class.

**VB Syntax**

```vbnet
props = calAll.PropertyValues (propName)
```

**Variable**

- **Type** - Description
- **props** (Variant Array) Variable to store the returned property values that can be set.
- **calAll** A CalibrateAllChannels *(object)*
- **propName** (String) Property name for which valid values are to be returned.

**Return Type**

- Variant Array
- **Default** Not Applicable

**Examples**

```vbnet
props = calAll.PropertyValues ("Noise Cal Method")
```

**C++ Syntax**

```cpp
HRESULT get_PropertyValues (BSTR propName, VARIANT* propValues);
```

**Interface**

- CalibrateAllChannels
Pulse4OutAsADCAcivity Property

**Description**
Turns pulse4 output ON and OFF. Enable pulse4 to use an oscilloscope connected to pin 13 of the PULSE I/O connector on the rear panel of the VNA to display when the ADC is making measurements.

**VB Syntax**
`pulse.Pulse4OutAsADCAcivity = value`

**Variable (Type) - Description**
- `pulse` A `PulseGenerator (object)`
- `value` (Boolean) Choose from:
  - **ON** (or 1) - Pulse 4 output pin indicates ADC activity.
  - **OFF** (or 0) - Pulse 4 output pin indicates legacy behavior (pulse generator number 4 output).

**Return Type**
Boolean

**Default**
False

**Examples**
- `pulse.Pulse4OutAsADCAcivity = True`  `'Write`
- `bool = pulse.Pulse4OutAsADCAcivity`  `'Read`

**C++ Syntax**
HRESULT get_Pulse4OutAsADCAcivity(VARIANT_BOOL* ADCActivity);
HRESULT put_Pulse4OutAsADCAcivityit (VARIANT_BOOL ADCActivity);

**Interface**
IPulseGenerator5
**PulseGeneratorID Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the ID of the specified External Pulse Generator name. Use this ID number when setting properties on the PulseGenerator Object. Use PulseGeneratorNames to read the names of the internal and configured external pulse generators.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = chan.PulseGeneratorID (name)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(Long Integer)</strong> - Variable to store the returned ID.</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>A Channel (object)</td>
</tr>
<tr>
<td><code>name</code></td>
<td>Name of the pulse generator. Use PulseGeneratorNames to read the names of configured pulse generators.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>id = chan.PulseGeneratorID &quot;My81110&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_PulseGeneratorID(long* count, BSTR name)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel23</td>
</tr>
</tbody>
</table>
### PulseGeneratorNames Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the string names of internal and configured external pulse generators.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>names = chan.PulseGeneratorNames</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>names</code></td>
<td>Variant array to store the returned string names.</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>A Channel <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant Array of string names</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>names = extPulseGen.PulseGeneratorNames  'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_PulseGeneratorNames (VARIANT *pNames)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel</td>
</tr>
</tbody>
</table>
## PulseMeasMode Property

Sets the pulse measurement state for the channel.

**Description**

Sets the pulse measurement state for the channel.

**VB Syntax**

```vb
pulseMeas.PulseMeasMode = value
```

**Variable**

(Types) - Description

- `pulseMeas` A `PulseMeasurementControl` (object)
- `value` (Enum as `NAPulseMeasurementMode`) Choose from:
  - (0) `naPulseMeasurementOff` - Turn OFF pulse measurements.
  - (1) `naPulseStandardMeasurement` - Turn ON standard pulse measurements.
  - (2) `naPulseProfileMeasurement` - Turn ON pulse profile measurements.

**Return Type**

Enum

**Default**

(0) `naPulseMeasurementOff`

**Examples**

```vb
pulse.PulseMeasMode = naPulseProfileMeasurement 'Write
value = pulse.PulseMeasMode 'Read
```

**C++ Syntax**

```cpp
HRESULT get_PulseMeasMode(tagNAPulseMeasurementMode* pVal);
HRESULT put_PulseMeasMode(tagNAPulseMeasurementMode newVal);
```

**Interface**

`IPulseMeasurementControl`
## PulseOffAlcMode Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Sets and returns ALC mode that the Pulse Setup will use when the pulse is turned off.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pulseMeas.PulseOffAlcMode = value</code></td>
</tr>
</tbody>
</table>
| **Variable**    | `{**Type** - Description}

- `pulseMeas` A `PulseMeasControl` `object`

- `value` `{**Enum as NAALCLevelingMode} Choose from:}

  - `(0) naALCInternal` - Set ALC to internal when the pulse is turned off.
  - `(1) naALCOpenLoop` - Set ALC to open loop to use no ALC when the pulse is turned off.

<table>
<thead>
<tr>
<th><strong>Return Type</strong></th>
<th>Enum</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td><code>(0) naALCInternal</code></td>
</tr>
</tbody>
</table>

| **Examples** | `pulseMeas.PulseOffAlcMode = naALCOpenLoop` `Write`
|              | `value = pulseMeas.PulseOffAlcMode` `Read` |
| **C++ Syntax**| `HRESULT get_PulseOffAlcMode(tagNAALCLevelingMode* pVal);` |
|              | `HRESULT put_PulseOffAlcMode(tagNAALCLevelingMode newVal);` |
| **Interface** | IPulseMeasControl4 |
PulseProfileStart Property

Sets and returns the start time of the pulse. Pulse profile measurements provide a time domain (CW frequency) view of the pulse envelope. Profiling is performed using a measurement technique that "walks" a narrow receiver "snapshot" across the width of the pulse. This is analogous to using a camera to take many small snapshots of a wide image, then piecing them together to form a single, panoramic view.

**VB Syntax**

```
pulseMeas.PulseProfileStart = value
```

**Variable**

- *pulseMeas*: A PulseMeasurementControl (object)
- *value*: (double) - Start time in seconds.

**Note:** The start value cannot be negative.

**Return Type**

Double

**Default**

0

**Examples**

```
pulseMeas.PulseProfileStart = 3e-3 'Write
startprofile = pulseMeas.PulseProfileStart 'Read
```

**C++ Syntax**

```
HRESULT get_PulseProfileStart(double *pVal)
HRESULT put_PulseProfileStart(double pVal)
```

**Interface**

IPulseMeasControl3
PulseProfileStop Property

Description
Sets and returns the stop time of the pulse. Pulse profile measurements provide a time domain (CW frequency) view of the pulse envelope. Profiling is performed using a measurement technique that "walks" a narrow receiver "snapshot" across the width of the pulse. This is analogous to using a camera to take many small snapshots of a wide image, then piecing them together to form a single, panoramic view.

VB Syntax
pulseMeas.PulseProfileStop = value

Variable (Type) - Description
pulseMeas
A PulseMeasurementControl (object)
value (double) - Stop time in seconds.

Note: The stop value cannot be negative.

Return Type
Double

Default
N/A

Examples
pulseMeas.PulseProfileStop = 3e-3 'Write
startprofile = pulseMeas.PulseProfileStop 'Read

C++ Syntax
HRESULT get_PulseProfileStop(double *pVal)
HRESULT put_PulseProfileStop(double pVal)

Interface
IPulseMeasControl3
### RangeCount (Independent Power Cal Port) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Queries how many ranges are included in the calibration for source port ( &lt;n&gt; ).</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = IndependentPwrrCalPort.RangeCount</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long)</em> - Variable to store the returned range count.</td>
</tr>
<tr>
<td><code>IndependentPwrrCalPort</code></td>
<td>A <code>IndependentPowerCalibrationPort (object)</code></td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Example</td>
<td><code>value = CalibrateAllChannels.IndependentPowerCalibrationPort.RangeCount</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT RangeCount(Int* ranges);</td>
</tr>
<tr>
<td>Interface</td>
<td>IIndependentPowerCalibrationPort</td>
</tr>
</tbody>
</table>
## RangeCount Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Returns the number of ranges that are available in the VNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>To see the range names, query the Name property of each range in the FOM collection.</td>
</tr>
</tbody>
</table>

**VB Syntax**

```
value = FOM.RangeCount
```

**Variable**

<table>
<thead>
<tr>
<th><strong>(Type)</strong></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>An FOM (collection object)</td>
</tr>
<tr>
<td>value</td>
<td>(long) - Variable to store the returned number of ranges.</td>
</tr>
</tbody>
</table>

**Return Type**

Long Integer

**Default**

Not Applicable

**Examples**

```vbnet
NumRanges = fom.RangeCount 'Read
```

**C++ Syntax**

```
HRESULT get_RangeCount(long *count)
```

**Interface**

IFOM
# rangeNumber Property

Returns the index number of the range within the FOM collection.

**VB Syntax**

```vbnet
value = FOMRange.rangeNumber
```

**Variable**

- **value** (Long) - Variable to store the returned range number.
- **object** An FOMRange (object)

**Return Type** Long

**Default** Not Applicable

**Examples**

```vbnet
num = fomRange.rangeNumber 'Read
```

**C++ Syntax**

```cpp
HRESULT get_rangeNumber(long *pVal)
```

**Interface** IFOMRange
RangeOffset Property

Sets and reads the frequency range number to be used as an offset. The frequencies of the range \(<n>\) will be offset from the 'coupled to' range by this frequency range. The RangeOffsetUp Property command determines whether the offset is positive or negative.

On the Frequency Range dialog under Coupling, this is the Offset setting.

VB Syntax

```vbnet
DIQ.RangeOffset (n) = value
```

Variable (Type) - Description

- **DIQ**: A Differential I/Q (object)
- **n**: (Long Integer) Frequency range number.
- **value**: (Long Integer) Offset range number. The resulting range must be within the frequency range of the analyzer.

Return Type

Long

Default

1

Examples

```vbnet
diq.RangeOffset 3 = 2 'Write, range 3 is offset from the 'coupled to' range by the frequencies defined by range 2
value = diq.RangeOffset 2 'Read
```

C++ Syntax

```c++
HRESULT get_RangeOffset(long range, long* RangeOffset);
HRESULT put_RangeOffset(long range, long RangeOffset);
```

Interface

IDIQ
**RangeOffsetUp Property**

**Description**
Sets and reads the state of the Up / Down conversion setting.

On the Frequency Range dialog under Coupling, this is the Up (On|Off) setting.

**VB Syntax**

```
DIQ.RangeOffsetUp (n) = value
```

**Variable**

- **DIQ** (Type) - Description
  - A Differential I/Q (object)
- **n** (Long Integer) Frequency range number.
- **value** (Boolean) Choose from the following:
  - True - The offset range is ADDED to the 'coupled to' frequency range.
  - False - The offset range is SUBTRACTED from the 'coupled to' frequency range.

**Return Type**
Boolean

**Default**
False

**Examples**

- `diq.RangeOffsetUp 2 = True` 'Write
- `value = diq.RangeOffsetUp 2` 'Read

**C++ Syntax**

```cpp
HRESULT get_RangeOffsetUp(long range, VARIANT_BOOL* RangeOffsetUp);

HRESULT put_RangeOffsetUp(long range, VARIANT_BOOL RangeOffsetUp);
```

**Interface**
IDIQ
# RangeStartFrequency Property

Sets and reads the start value for the specified frequency range.

On the **Frequency Range** dialog under **Frequency**, this is the **Start** setting.

**Description**

Sets and reads the start value for the specified frequency range.

**VB Syntax**

```vbnet
dIQ.RangeStartFrequency(n) = value
```

**Variable**

- `DIQ` - A Differential I/Q object
- `n` - Frequency range number. (Long)
- `value` - Frequency range start value. Choose a value within the frequency range of the analyzer. (Double)

**Return Type**

Double

**Default**

Start frequency of the analyzer.

**Examples**

```vbnet
diq.RangeStartFrequency(2) = 1e9 'Write
value = diq.RangeStartFrequency(1) 'Read
```

**C++ Syntax**

```cpp
HRESULT get_RangeStartFrequency(long range, double* RangeStartFrequency);

HRESULT put_RangeStartFrequency(long range, double RangeStartFrequency);
```

**Interface**

IDIQ
# RangeStartFrequency (PowerRange) Property

**Description**
Sets and reads the lower bound of the frequency range used for range based power min and max.

**VB Syntax**
```vbnet
powerRange.RangeStartFrequency = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>powerRange</code></td>
<td>A PowerRange (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>A (Double) Start frequency. Choose a number within the frequency limits of the analyzer. Units are Hz.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
Not Applicable

**C++ Syntax**
```cpp
HRESULT get_RangeStartFrequency(double* startfreq);
HRESULT put_RangeStartFrequency(double startfreq);
```

**Interface**
IPowerRange
**RangeStopFrequency Property**

**Description**
Sets and reads the Stop value for the specified frequency range.

On the Frequency Range dialog under Frequency, this is the Stop setting.

**VB Syntax**
```
DIQ.RangeStopFrequency (n) = value
```

**Variable (Type) - Description**
- **DIQ** (A Differential I/Q (object))
- **n** (Long) Frequency range number.
- **value** (Double) Frequency range stop value. Choose a value within the frequency range of the analyzer.

**Return Type**
Double

**Default**
Maximum frequency of the analyzer.

**Examples**
```
diq.RangeStopFrequency(2) = 1e9 'Write
value = diq.RangeStopFrequency(1) 'Read
```

**C++ Syntax**
```
HRESULT get_RangeStopFrequency(long range, double* RangeStopFrequency);
HRESULT put_RangeStopFrequency(long range, double RangeStopFrequency);
```

**Interface**
IDIQ
### RangeStopFrequency (PowerRange) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the upper bound of the frequency range used for range based power min and max.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>powerRange.RangeStopFrequency = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>powerRange</code></td>
<td>A PowerRange (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Stop frequency. Choose a number within the frequency limits of the analyzer. Units are Hz.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><em>Write</em></td>
<td><code>powerRange.RangeStopFrequency = 10e9</code></td>
</tr>
<tr>
<td><em>Read</em></td>
<td><code>value = powerRange.RangeStopFrequency</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_RangeStopFrequency(double* stopfreq);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_RangeStopFrequency(double stopfreq);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPowreRange</td>
</tr>
</tbody>
</table>
## RatioedPowerCorrectionData Property

**Description**  
Write and read an array of ratioed power offsets. This allows the setting of arbitrary impedance, which is used for active load applications. The number of offset values must be the same as the number of data points. Use RatioedPowerCorrectionEnabled to use the offset values.

**VB Syntax**  

```vbnet
phase.RatioedPowerCorrectionData(srcPort) = value
```

**Variable**  
*(Type)* - Description  

- `phase` A `PhaseControl` Object  
- `srcPort` (Long Integer) Source port for which to make phase control settings.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

**value** (Long values) Ratioed amplitude offset data array.

**Return Type**  
Long data array

**Default**  
Not Applicable

**Examples**  

```vbnet
phase.RatioedPowerCorrectionData 1 = .1,.2,.3 ' Write 3 power offset values
value = phase.RatioedPowerCorrectionData 2 ' Read
```

**C++ Syntax**  

```c++
HRESULT get_RatioedPowerCorrectionData(long port, long* pVals);
HRESULT put_RatioedPowerCorrectionData(long port, long newVals);
```

**Interface**  
`IPhaseControl`
# RatioedPowerCorrectionEnabled Property

## Description
Write and read whether to use the ratioed power offset array. Use `RatioedPowerCorrectionData Property` to write or read the offset data.

## VB Syntax
```
phase.RatioedPowerCorrectionEnabled(srcPort) = value
```

## Variable
- **(Type) - Description**
  - **phase**: A `PhaseControl Object`
  - **srcPort**: (Long Integer) Source port for which to make phase control settings.

- **Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

- **value**: (Boolean) Ratioed power offset array state.
  - **True** – Apply offset array.
  - **False** – Do NOT apply offset array.

## Return Type
Boolean

## Default
False

## Examples
```
phase.RatioedPowerCorrectionEnabled 1 = True ' Write
value = phase.RatioedPowerCorrectionEnabled 2 ' Read
```

## C++ Syntax
```
HRESULT get_RatioedPowerCorrectionEnabled(long port, VARIANT_BOOL* pVal);

HRESULT put_RatioedPowerCorrectionEnabled(long port, VARIANT_BOOL newVal);
```

## Interface
`IPhaseControl`
## RcvCharMethod Property

**Description**  
Set and read the method used to characterize the noise receivers.

**VB Syntax**  

```vbscript
noise.RcvCharMethod = value
```

**Variable (Type) - Description**

- **noise**  
  A `NoiseCal (object)`

- **value**  
  (string) Receiver characterization method. NOT case-sensitive. Choose from:

  - **"NoiseSource"** - Use a noise source. This selection is NOT allowed when a standard VNA receiver is used as the noise receiver. (`noise.NoiseReceiver = naStandardReceiver`).

  - **“PowerMeter”** - Use a power meter.

  **NOTE:** The Power Meter selection is **NOT** allowed when the Noise Bandwidth is **8 MHz** or **24 MHz**.

**Return Type**  
String

**Default**  
"NoiseSource"

**Examples**

```vbscript
noise.RcvCharMethod = "PowerMeter" 'Write
receiverMethod = noise.RcvCharMethod 'Read
```

**C++ Syntax**  

```csharp
HRESULT get_RcvCharMethod(BSTR* pValue)
HRESULT put_RcvCharMethod(BSTR pNewValue)
```

**Interface**  
INoiseCal3
### ReadDCAtCompression Property

**Description**  Set and read the DC readings at the compression point in the last iteration of a smart sweep. Taking only these DC readings improves measurement speed.

**VB Syntax**  
\[ gca.ReadDCAtCompression = value \]

**Variable**  
- **Type** - Description
  - \( gca \)  A **GainCompression** (object)
  - \( value \)  (Boolean)  Choose from:
    - **True**  Enable reading DC value at compression point in the last iteration of a smart sweep.
    - **False**  Disable reading DC value at compression point in the last iteration of a smart sweep.

**Return Type**  Boolean

**Default**  False

**Examples**  
- \[ gca.ReadDCAtCompression = True \]  'Write
- \[ enable = gca.ReadDCAtCompression \]  'Read

**C++ Syntax**  
- `HRESULT get_ReadDCAtCompression(VARIANT_BOOL* enable)`
- `HRESULT put_ReadDCAtCompression(VARIANT_BOOL enable)`

**Interface**  IGainCompression5
R1inputPath Property

**Description**
All VNA models (except the N523xA models) have a switch in the test set that allows access to the port 1 reference receiver through the front panel Reference 1 connectors. This command throws that switch between the internal path to the receiver, or through the external connectors.

See other Frequency Offset properties.

**VB Syntax**
```vbnet
chan.R1InputPath = value
```

**Variable (Type) - Description**
- **chan** A Channel (object)
- **value** (Enum as naInputPath) - Choose from:
  - **naPathInternal** - (0) - internal path to the reference receiver
  - **naPathExternal** - (1) - path through external connectors

**Return Type**
Enum

**Default**
aPathInternal - (0)

**Examples**
```vbnet
chan.R1InputPath = naPathInternal 'Write
Inpath = chan.R1InputPath 'Read
```

**C++ Syntax**
```csharp
HRESULT get_R1InputPath (tag NAInputPath *pPath);
HRESULT put_R1InputPath (tag NAInputPath newPath);
```

**Interface**
IChannel2
ReadingsPerPoint Property

This command, along with ReadingsTolerance, allows for settling of the power sensor READINGS.

Specifies the maximum number of power readings that are taken at each stimulus point to allow for power meter settling. Each reading is averaged with the previous readings at that stimulus point. When this average meets the ReadingsTolerance value or this number of readings has been made, the average is returned as the valid reading.

VB Syntax

\[ \text{pwrSensor.ReadingsPerPoint} = \text{value} \]

Variable (Type) - Description

- \( \text{pwrCal} \): A SourcePowerCalibrator (Object) or A PowerSensorAsReceiver (Object)

- \( \text{value} \): (long integer) – Number of power readings. Choose any number between 3 and 100.

Return Type

Long Integer

Default

3

Examples

\[ \text{pwrSensor.ReadingsPerPoint} = 3 \] 'Write
\[ \text{numReadings} = \text{pwrSensor.ReadingsPerPoint} \] 'Read

C++ Syntax

HRESULT put_ReadingsPerPoint(long newVal);

HRESULT get_ReadingsPerPoint(long *pVal);

Interface

ISourcePowerCalibrator

IPowerSensorAsReceiver
ReadingsTolerance Property

Description
This command, along with ReadingsPerPoint Property allows for settling of the power sensor READINGS.

Each power reading is averaged with the previous readings at each stimulus point. When the average meets this tolerance value or the maximum ReadingsPerPoint has been made, the average is returned as the valid reading.

VB Syntax
`pwrSens.ReadingsTolerance = value`

Variable (Type) - Description
`pwrCal` A SourcePowerCalibrator (Object) or
A PowerSensorAsReceiver (Object)

`value` (Double) – Power meter settling tolerance value in dB. Choose any number between 0 and 5.

Return Type
Double

Default
.05 dB

Examples
`pwrSens.ReadingsTolerance = .1 'Write ReadTol = pwrSensor.ReadingsTolerance 'Read`

C++ Syntax
HRESULT get_ReadingsTolerance( double *pVal);
HRESULT put_ReadingsTolerance( double newVal);

Interface
ISourcePowerCalibrator3
IPowerSensorAsReceiver
### ReadyForTriggerPolarity Property

**Description**

Specifies the polarity of Ready for Trigger output.

All existing Ready for Trigger outputs for PNA-X and PNA-L models are configured simultaneously with this command. See Capabilities Summary.

The Ready for Trigger polarity can NOT be configured for E836x models.

**VB Syntax**

```vbnet
trigsetup.ReadyForTriggerPolarity = value
```

**Variable (Type) - Description**

- `trigsetup` A TriggerSetup (object)
- `value` (Enum as NALevel)

Choose from:

- **0 - naLow** - Outputs a TTL low when the VNA is ready for trigger.
- **1 - naHigh** - Outputs a TTL high when the VNA is ready for trigger.

**Return Type**

Enum

**Default**

0 - naLow

**Examples**

```vbnet
trigsetup.ReadyForTriggerPolarity = naLow 'Write
pol = trigsetup.ReadyForTriggerPolarity 'Read
```

**C++ Syntax**

```cpp
HRESULT get_ReadyForTriggerPolarity(tagNALevel *pVal);

HRESULT put_ReadyForTriggerPolarity(tagNALevel newVal);
```

**Interface**

ITriggerSetup3
Read/Write

About the Handler I/O Connector

ReadyForTriggerState Property

Description
Determines the control of Material Handler connector Pin 21.

VB Syntax
handler.ReadyForTriggerState = value

Variable
- **handler** *(object)* - A Handler I/O object
- **value** *(boolean)*

- False - Pin 21 is controlled by Output Port B7
- True - Pin 21 is controlled by the Ready for Trigger signal

Return Type
Boolean

Default
False

Examples
handler.ReadyForTriggerState = False 'Write
bState = handler.ReadyForTriggerState 'Read

C++ Syntax
HRESULT put_ReadyForTriggerState (BOOL *pVal);

HRESULT get_ReadyForTriggerState (BOOL newVal);

Interface
IHWMaterialHandlerIO2
ReadyForTriggerStatus Property

Description
Checks if the PNA is ready for a hardware trigger.

This command is not intended to be used in a dynamic triggering situation where the ready status is constantly changing. Instead, the expected use is a more static situation where you are expecting the PNA to transition from not ready to ready, and then wait for a trigger. The PNA is polled until it becomes ready and then an operation that triggers the PNA is performed.

Note: This command is only supported on the PNA-L, PNA, and PNA-X with DSP5 installed. Any other model will return an error.

VB Syntax
value = trigsetup.ReadyForTriggerStatus

Variable
(trigsetup) - A TriggerSetup (object)
(value) - (Enum as NATriggerReadyStatusSource)

Return Type
Boolean

Default
Not applicable

C++ Syntax
HRESULT get_ReadyForTriggerStatus(tagNATriggerReadyStatusSource whichTrigger, VARIANT_BOOL* val);

Interface
ITriggerSetup4
**RecallSoftkeysMostRecent Property**

**Description**
Set and return whether to list files for recall on softkeys by most-recently used or alphabetically.

**VB Syntax**

```
pref.RecallSoftkeysMostRecent = value
```

**Variable (Type) - Description**

- `pref` - A Preferences (object)
- `value` - (Boolean) – Choose from:
  - **True** – Recall softkeys show most recently-used files.
  - **False** – Recall softkeys show alphabetically-ordered files.

**Return Type**
Boolean

**Default**
False

**Examples**

```
pref.RecallSoftkeysMostRecent = False  'Write
recallPref = pref.RecallSoftkeysMostRecent 'Read
```

**C++ Syntax**

```c++
HRESULT get_RecallSoftkeysMostRecent(VARIANT_BOOL * pVal);
HRESULT put_RecallSoftkeysMostRecent(VARIANT_BOOL pVal);
```

**Interface**
IPreferences13
ReceiverAttenuator Property

Description  Sets or returns the value of the specified receiver attenuator control.

VB Syntax  

\[ \text{object.ReceiverAttenuator}(\text{rec}) = \text{value} \]

Variable  

\( \text{(Type)} \) - Description

- \( \text{object} \) Channel (object)

- \( \text{or} \)

CalSet (object) - Read-only property

- \( \text{rec} \) (long integer) - Receiver with attenuator control to be changed. Choose from any of the available receivers in your VNA
  1 - Receiver A
  2 - Receiver B
  3 - Receiver C
  4 - Receiver D

Receiver attenuation can not be set using logical receiver notation.

- \( \text{value} \) (double) - Attenuator value in dB. Choose any Long Integer between 0 and 35 in 5dB steps:

  If an invalid value is entered, the analyzer will select the next lower valid value. For example, if 19.9 is entered the analyzer will select 15 dB attenuation.

Return Type  Double

Default  0 db

Examples  

\begin{align*}
\text{chan.ReceiverAttenuator}(1) &= 5 \quad \text{\texttt{Write}} \\
\text{attn} &= \text{chan.ReceiverAttenuator}(2) \quad \text{\texttt{Read}}
\end{align*}

C++ Syntax  

\begin{verbatim}
HRESULT get_ReceiverAttenuator(long lport, double* pVal)
HRESULT put_ReceiverAttenuator(long lport, double newVal)
\end{verbatim}

Interface  IChannel

| CalSet3

2207
### ReceiverAttenuator (Cal All) Property

**Description**
Sets and returns the Receiver Attenuator setting for a Cal All calibration.

**VB Syntax**
```vbnet
calAll.ReceiverAttenuator (port) = value
```

**Variable (Type) - Description**
- `calAll` A `ICalibrateAllChannels` object
- `port` (Long) Receiver port number.
- `value` (Double) Attenuation value in dB for a Cal All calibration. Choose a valid value for the VNA model. See valid settings.

**Return Type**
Double

**Default**
0 (zero dB)

**Examples**
```vbnet
calAll.ReceiverAttenuator = 0 'Set value
value = calAll.ReceiverAttenuator 'Return value
```

**C++ Syntax**
```cpp
HRESULT get_ReceiverAttenuator (long port, double val);
HRESULT put_ReceiverAttenuator (long port, double* newVal);
```

**Interface**
ICalibrateAllChannels
### ReceiverCount Property

**Description**
Returns the number of receivers in the remote VNA. The returned number includes both test port receivers and reference receivers. See the number of reference receivers in your VNA.

**VB Syntax**
```
value = cap.ReceiverCount
```

**Variable**
- **Type**: Long
- **Description**: Variable to store the returned number of receivers.
- **value**: A Capabilities (object)

**Return Type**: Long

**Default**: Not Applicable

**Examples**
```
value = cap.ReceiverCount  'Read
```

**C++ Syntax**
```
HRESULT get_ReceiverCount(long * receiverCount);
```

**Interface**: ICapabilities
**ReceiverRatio Property**

**Description**
Sets and returns the receiver ratio pair to be used with Receiver Leveling.

To perform receiver leveling with a ratioed receiver, use `ReceiverRatio Property`.

**VB Syntax**

```
RxLevel.ReceiverRatio(srcPort) = value
```

**Variable**

- **RxLevel**: A `ReceiverLeveling` Object
- **srcPort**: (Long Integer) Source port for which to set the receiver for Receiver Leveling.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number.

To learn more see [Remotely Specifying a Source Port](#).

**value**: (String) Receiver to use for the leveling sweeps. (Not case sensitive). Choose any receiver in your VNA. See the block diagram of your VNA, located at the bottom of all `Specs` documents.

**Return Type**: String

**Default**: Not applicable

**Examples**

```
rxLevel.ReceiverRatio (1) = "R1" ' Write

rxLevel.ReceiverRatio (1) = "b2" ' Write

value = rxLevel.ReceiverRatio 2' Read
```

**C++ Syntax**

```
HRESULT get_ReceiverRatio( long port, BSTR* pVal);

HRESULT put_ReceiverRatio( long port, BSTR newVal);
```

**Interface**

`IReceiverLevelingConfiguration`
**ReceiverStepAttenuatorStepSize Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a value indicating the step size of the attenuator.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.ReceiverStepAttenuatorStepSize (n)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(Double) -</strong> Variable to store the returned value of the attenuator step size.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities <strong>(object)</strong></td>
</tr>
<tr>
<td><code>n</code></td>
<td><strong>(Long) -</strong> port number to query for the value of the attenuator step size.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.ReceiverStepAttenuatorStepSize(1)</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_ReceiverStepAttenuatorStepSize(long portNumber, double * stepSize );</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities</td>
</tr>
</tbody>
</table>
### ReceiverTemperature Property

**Description:** Returns the current temperature on the VNA receiver board.

**VB Syntax:**

```vbnet
value = cap.ReceiverTemperature (unit)
```

**Variable**

- **value** *(Double)* - Variable to store the returned temperature value.
- **cap** A Capabilities *(object)*
- **unit** *(Enum as NATempScale)* - Units in which temperature is returned. Choose from:
  - 0 - naTempScale_Fahrenheit
  - 1 - naTempScale_Celsius

**Return Type:** Double

**Default:** Not Applicable

**Examples:**

```vbnet
value = cap.ReceiverTemperature naTempScale_Celsius 'Read
```

**C++ Syntax:**

```c
HRESULT get_ReceiverTemperature(NATempScale val, long temperature);
```

**Interface:** ICapabilities9
ReceivePort Property

Description
Returns the receiver (response) port number of measurement. To understand how this property is useful, see IMeasurement2 Interface.

Note: Returning a receiver port is only supported for S-Parameter measurements. If the measurement is not an S-Parameter, then E_NA_BAD_PARAMETER is returned.

VB Syntax
value = meas.ReceivePort

Variable (Type) - Description
value (Long) - Variable to store the returned value
meas A Measurement (object)

Return Type
Long Integer

Default
Not Applicable

Examples
rp = meas.ReceivePort

C++ Syntax
HRESULT ReceivePort(Long* revPort);

Interface IMeasurement2
### RecordSize Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the current DFT record size. This value is based on the ForceADCRecordSize and dft.Type settings.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>\texttt{value} = \texttt{sa.dft.RecordSize}</td>
</tr>
</tbody>
</table>
| Variable    | \begin{itemize} 
  \item \texttt{sa.dft} \quad \textbf{(Type)} - A \texttt{SpectrumAnalyzerDFT} \textbf{(object)}
  \item \texttt{value} \quad \textbf{(Long)} - Variable to store the returned value. \end{itemize} |
| Return Type | Long                                                                                             |
| Example     | \texttt{value = sa.dft.RecordSize \ 'Read}                                                         |
| C++ Syntax  | \texttt{HRESULT get_RecordSize(long* val);}                                                       |
| Interface   | ISpectrumAnalyzerDFT                                                                            |
## RedTraceOnFail Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and return whether to display limit line failures as red trace segments or red data points (dots).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pref.RedTraceOnFail = bool</code></td>
</tr>
</tbody>
</table>
| **Variable** | (Type) - Description  
`pref` - A `Preferences (object)`  
`bool` - (Boolean) - Choose from:  
  - **False** - Display failures as red data points (dots).  
  - **True** - Display failures as red trace segments. (Red Trace On Fail). |
| **Return Type** | Boolean |
| **Default** | False |
| **Examples** | `pref.RedTraceOnFail = False` 'Write  
`prefer = pref.RedTraceOnFail` 'Read |
| **C++ Syntax** | `HRESULT put_RedTraceOnFail( VARIANT_BOOL bValue)`  
`HRESULT get_RedTraceOnFail( VARIANT_BOOL *bValue)` |
| **Interface** | IPreferences10 |
## ReduceIFBandwidth Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the state of the Reduced IF Bandwidth at Low Frequencies setting.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan.ReduceIFBandwidth = state</code></td>
</tr>
<tr>
<td><strong>Variable (Type)</strong></td>
<td>A Channel (object)</td>
</tr>
<tr>
<td><code>chan</code></td>
<td></td>
</tr>
<tr>
<td><code>state</code></td>
<td>(Enum as naStates)</td>
</tr>
<tr>
<td>0 - naOFF</td>
<td>Turns Reduce IFBW OFF</td>
</tr>
<tr>
<td>1 - naON</td>
<td>Turns Reduce IFBW ON</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum as naStates</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>naON</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chan.ReduceIFBandwidth = naOFF'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>reduce = chan.ReduceIFBandwidth 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_ReduceIFBandwidth (tagNAStates *pState)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_ReduceIFBandwidth (tagNAStates newState)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel5</td>
</tr>
</tbody>
</table>
ReferenceCalFactor Property

**Description**
Sets and returns the Cal Factor (%) for the 50 MHz reference signal associated with this power sensor. Use this property **only** if the power sensor does not contain cal factors in EPROM (for example, HP/Keysight 848x sensors).

**VB Syntax**
```
pwrSensor.ReferenceCalFactor = value
```

**Variable**
**(Type) - Description**
- `pwrSensor` A `PowerSensor` (Object)
- `value` *(double)* – Cal factor in units of percent. This can be any value between 1 and 150.

**Return Type**
Double

**Default**
100

**Examples**
```
pwrSensor.ReferenceCalFactor = 98 'R

RefFact = pwrSensor.ReferenceCalFactor 'Read
```

**C++ Syntax**
```
HRESULT put_ReferenceCalFactor(double newVal);

HRESULT get_ReferenceCalFactor(double *pVal);
```

**Interface**
IPowerSensor

IPowerSensorAsReceiver

---

**About Source Power Cal**

**About PMAR**
# ReferenceMarkerState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turn ON or OFF the reference marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.ReferencemarkerState = state</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>app</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><code>state</code></td>
<td>(boolean) -</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>turns the reference marker ON</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>turns the reference marker OFF</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td><strong>False</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meas.ReferencemarkerState = True</code></td>
</tr>
<tr>
<td><code>reference = meas.ReferencemarkerState</code></td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_ReferenceMarkerState(VARIANT_BOOL bState)</code></td>
</tr>
<tr>
<td><code>HRESULT put_ReferenceMarkerState(VARIANT_BOOL* bState)</code></td>
<td></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
# ReferenceReceiver Property

**Description**
Sets and returns the receiver to be used with Receiver Leveling.

To perform receiver leveling with a ratioed receiver, use `ReceiverRatio Property`.

<table>
<thead>
<tr>
<th>VB Syntax</th>
<th>RxLevel.<strong>ReferenceReceiver</strong>(srcPort) = value</th>
</tr>
</thead>
</table>

**Variable** *(Type) - Description*

- **RxLevel** A `ReceiverLeveling` Object
- **srcPort** (Long Integer) Source port for which to set the receiver for Receiver Leveling.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number.

To learn more see [Remotely Specifying a Source Port](#).

**value** *(String) Receiver to use for the leveling sweeps. (Not case sensitive). Choose any receiver in your VNA. see the block diagram of your VNA, located at the bottom of all Specs documents.*

Receivers can also be referred to using logical receiver notation. This notation makes it easy to refer to receivers with an external test set connected to the VNA. You do not need to know which physical receiver is used for each test port. Learn more.

**Return Type** String

**Default** Not applicable

**Examples**

```vbnet
rxLevel.ReferenceReceiver (1) = "R1" ' Write

rxLevel.ReferenceReceiver (1) = "b2" ' Write

value = rxLevel.ReferenceReceiver 2 ' Read
```

**C++ Syntax**

```csharp
HRESULT get_ReferenceReceiver( long port, BSTR* pVal);

HRESULT put_ReferenceReceiver( long port, BSTR newVal);
```

**Interface** `IReceiverLevelingConfiguration`
ReferenceValue Property

Description: Sets or returns the value of the Y-axis Reference Level of the active trace.

VB Syntax: `tree.ReferenceValue = value`

Variable (Type) - Description:
- `tree` (object) - A Trace
- `value` (double) - Reference Value. Units and range depend on the current data format.

Return Type: Double

Default: Not applicable

Examples:
```
meas.ReferenceValue = 0 'Write
rlev = meas.ReferenceValue 'Read
```

C++ Syntax:
- `HRESULT get_ReferenceValue(double *pVal)`
- `HRESULT put_ReferenceValue(double newVal)`

Interface: ITrace
### ReferencePosition Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the Reference Position of the active trace.</th>
</tr>
</thead>
</table>

**VB Syntax**

```vbnet
trace.ReferencePosition = value
```

**Variable (Type) - Description**

- `trace` A Trace (object)
- `value` (double) - Reference position on the screen measured in horizontal graticules from the bottom of the screen. Choose from any number between: 0 and 10.

**Return Type**

Double

**Default**

0

**Examples**

```vbnet
meas.ReferencePosition = 5 'Middle of the screen -Write
rpos = meas.ReferencePosition -Read
```

**C++ Syntax**

- HRESULT get_ReferencePosition(double *pVal)
- HRESULT put_ReferencePosition(double newVal)

**Interface**

ITrace
### RelativeExtractionToneLevel Property

**Description**: Set and read the tone power relative to the input power (dBc).

**VB Syntax**: `HotS22.RelativeExtractionToneLevel = level`

**Variable**
- **(Type)** - Description
  - `HotS22` A [ActiveParametersApp](#) *(object)*
  - `level` *(Double)* - Relative tone power level

**Return Type**: Double

**Default**: -15 dBc

**Examples**
- `HotS22.RelativeExtractionToneLevel = -5 'Write`
- `level = HotS22.RelativeExtractionToneLevel 'Read`

**C++ Syntax**
- `HRESULT get_RelativeExtractionToneLevel(double* level)`
- `HRESULT put_RelativeExtractionToneLevel(double level)`

**Interface**: IActiveChannelSettings
### RemoteCalStoragePreference Property

**Description**
Specifies the default manner in which calibrations performed using COM or SCPI are to be stored.

Cal data is always stored to the channel’s Cal Register regardless of this setting.

This setting survives instrument preset and reboot. It remains until changed by another invocation of this property.

**VB Syntax**

```vbnet
pref.RemoteCalStoragePreference = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cal</code></td>
<td>A Preferences (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Enum) - Choose from:</td>
</tr>
</tbody>
</table>

- **0 - naPreferCalRegister** - Cal is saved ONLY to the channel’s Cal Register.

- **1 - naPreferNewUserCalSet** - Cal is automatically saved to a new User Cal Set file when performing a calibration using COM. The Cal Set name is automatically generated. This corresponds to pre-6.0 behavior. Use the Name property to change the name after the cal is complete.

- **2 - naPreferReuseCurrentCalSet** - The cal is saved to the Cal Set is that is currently selected on the specific channel. This could be the channel’s Cal Register. If the channel does not yet have a selected Cal Set, the cal will be saved to a new User Cal Set with an automatically-generated name.

**Return Type**

Enum

**Default**

0 - naPreferCalRegister

**Examples**

```vbnet
pref.RemoteCalStoragePreference = naPreferNewUserCalSet  'Write
calStorageMode = pref.RemoteCalStoragePreference  ' Read
```

**C++ Syntax**

```cpp
HRESULT get_RemoteCalStoragePreference(enum NARemoteCalStoragePreference* preference);

HRESULT put_RemoteCalStoragePreference(enum NARemoteCalStoragePreference val);
```

**Interface**

IPreferences7
## ReportReceiverOverload Property

**Description**
Set and return whether to display receiver overload warnings.

**VB Syntax**
```
pref.ReportReceiverOverload = value
```

**Variable**
- **pref**  
  A Preferences (object)
- **value**  
  (Boolean) - Choose from:
  - **True**  Display overload warnings.
  - **False**  Do NOT display overload warnings.

**Return Type**
Boolean

**Default**
True

**Examples**
```
pref.ReportReceiverOverload = True 'Write
value = pref.ReportReceiverOverload 'Read
```

**C++ Syntax**
```
HRESULT get_ReportReceiverOverload (VARIANT_BOOL PowerSweepRetraceMode* preference);
HRESULT put_ReportReceiverOverload (VARIANT_BOOL PowerSweepRetraceMode val)
```

**Interface**
IPreferences12
ResBWList Property

**Description**
Returns a list of Res BW values that are supported by the IM Spectrum apps.

**VB Syntax**
```vbnet
value = cap.ResBWList
```

**Variable**
- `(Type)` - Description
- `value` (Variant) - Variable to store the returned array of Res BW values
- `cap` A Capabilities (Object)

**Return Type**
Variant array

**Default**
Not Applicable

**Examples**
```
'Read the supported IFBW values
Set app = CreateObject("AgilentPNA835x.Application")
Set cap = app.Capabilities
list=cap.ResBWList
For i = 0 To UBound(list)
    msg = msg & list(i) & vbCrLf
Next
MsgBox msg
```

**C++ Syntax**
```c++
HRESULT get_ResBWList(Variant *value);
```

**Interface**
ICapabilities8
### ResBWListSA Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the list of supported Resolution BW values for the SA channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( value = cap.ResBWListSA )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>( value )</td>
<td>(Variant) - Variable to store the returned array of Res BW values</td>
</tr>
<tr>
<td>( cap )</td>
<td>A Capabilities (Object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant array</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>'Read the supported IFBW values</td>
</tr>
<tr>
<td></td>
<td>Set app = CreateObject(&quot;AgilentPNA835x.Application&quot;)</td>
</tr>
<tr>
<td></td>
<td>Set cap = app.Capabilities</td>
</tr>
<tr>
<td></td>
<td>list=cap.ResBWListSA</td>
</tr>
<tr>
<td></td>
<td>For i = 0 To UBound(list)</td>
</tr>
<tr>
<td></td>
<td>msg = msg &amp; list(i) &amp; vbCrLf</td>
</tr>
<tr>
<td></td>
<td>Next</td>
</tr>
<tr>
<td></td>
<td>MsgBox msg</td>
</tr>
</tbody>
</table>

**C++ Syntax**

```
HRESULT get_ResBWListSA(Variant *value);
```
ResBWList (Spectrum Analyzer) Property

Description
Read the list of all Resolution Bandwidth values currently supported with spectrum analyzer current settings.

VB Syntax
```
sa.ResBWList = value
```

Variable (Type) - Description
- `sa` A `SpectrumAnalyzer` (object)
- `value` (Variant) Variable to store the returned Resolution Bandwidth values.

Return Type
Variant array

Default
N/A

Examples
```
value = sa.ResBWList 'Read
```

C++ Syntax
```
HRESULT get_ResBWList(VARIANT* resolution);
```

Interface
ISpectrumAnalyzer5
### Resolution Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the DFT resolution.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>sa.dft.Resolution = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>sa.dft</code></td>
<td>A <code>SpectrumAnalyzerDFT</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Variable to store the returned DFT resolution.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>N/A</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = sa.dft.DFTResolution</code> 'Read'</td>
</tr>
</tbody>
</table>

**C++ Syntax**

```cpp
HRESULT get_Resolution(double* resolution);
```

**Interface**

`ISpectrumAnalyzerDFT`
# ResolutionBW Property

Sets and returns the Resolution Bandwidth for the IM Spectrum measurement.

**VB Syntax**

```vbnet
ims.ResolutionBW = value
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ims</code></td>
<td>An IMSpectrum Object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Resolution BW in Hz. Choose from:</td>
</tr>
</tbody>
</table>

- 60k | 100k | 150k | 300k, 600k | 1.0M | 3.0M

If an invalid number is specified, the VNA will round up to the closest valid number.

**Return Type**

Double

**Default**

600 kHz

**Examples**

```vbnet
ims.ResolutionBW = 150e3 'Write
value = ims.ResolutionBW 'Read
```

**C++ Syntax**

```cpp
HRESULT get_ResolutionBW(double *pVal)
HRESULT put_ResolutionBW(double newVal)
```

**Interface**

IIMSpectrum
## ResolutionBW Property

**Description**
Set and read the resolution bandwidth. Also set ResolutionBWMode to naManual.

**VB Syntax**

\[ sa\_ResolutionBW = value \]

**Variable**

1. **sa**
   - A SpectrumAnalyzer (object)
2. **value**
   - (Double) Choose a value between 6 Hz and 3 MHz. Attempting to set the bandwidth outside these bounds will force the bandwidth to the nearest bound.

**Learn about these settings.**

**Return Type**
Double

**Default**
100 kHz

**Examples**

\[ sa\_ResolutionBW = 1e6 \]  
\[ value = sa\_ResolutionBW \]

See an example program.

**C++ Syntax**

```c++
HRESULT put_ResolutionBW(double freq);
HRESULT get_ResolutionBW(double* freq);
```

**Interface**
ISpectrumAnalyzer
<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Read the maximum resolution bandwidth.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = sa.ResolutionBWMax</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>sa</code></td>
<td>A SpectrumAnalyzer <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><code>value = sa.ResolutionBWMax</code></td>
<td><em>Read</em></td>
</tr>
</tbody>
</table>

See an example program.

**C++ Syntax**

```c++
HRESULT get_ResolutionBWMax(tagNASAResolutionBWMax* maxbw);
```

**Interface**

ISpectrumAnalyzer
### ResolutionBWMin

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the minimum resolution bandwidth.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = sa.ResolutionBWMin</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td></td>
<td><code>sa</code> A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td></td>
<td><code>value = sa.ResolutionBWMin 'Read</code></td>
</tr>
</tbody>
</table>

See an example program.

<table>
<thead>
<tr>
<th>C++ Syntax</th>
<th>HRESULT get_ResolutionBWMin(tagNASAResolutionBWMin* minbw);</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>ISpectrumAnalyzer</td>
</tr>
</tbody>
</table>
## ResolutionBWMode Property

**Description**
Set and read how the resolution bandwidth is set. When ON, resolution bandwidth is set based on Span/RBW ratio.

**VB Syntax**

```vbnet
sa.ResolutionBWMode = value
```

**Variable**

- **Type**
  - `sa` A SpectrumAnalyzer (`object`)
  - `value` (`Enum as NAModes`) Choose from:
    - **1** - `naMANUAL` - Res. BW is set manually using `ResolutionBW` Property
    - **0** - `naAUTO` - Res. BW is set automatically and will not set resolution bandwidth above 300 kHz.

Learn about these settings.

**Return Type**
Enum

**Default**

- **0** - `naAUTO`

**Examples**

```vbnet
sa.ResolutionBWMode = 1 'Write
value = sa.ResolutionBWMode 'Read
```

See an example program.

**C++ Syntax**

```cpp
HRESULT put_ResolutionBWMode(tagNAModes mode);
HRESULT get_ResolutionBWMode(tagNAModes* mode);
```

**Interface**
ISpectrumAnalyzer
# ResolutionBWVideoBWRatio Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the RBW / VBW ratio.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>\texttt{sa.ResolutionBWVideoBWRatio = value}</td>
</tr>
<tr>
<td>Variable</td>
<td>\texttt{(Type)} - Description \texttt{sa A SpectrumAnalyzer (object)} \texttt{value (Double) RBW / VBW ratio.}</td>
</tr>
</tbody>
</table>

Learn about these settings.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Double</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>1.0</td>
</tr>
</tbody>
</table>

| Examples        | \texttt{sa.ResolutionBWVideoBWRatio = 1} \texttt{ 'Write} \texttt{value = sa.ResolutionBWVideoBWRatio 'Read} |

See an example program.

<table>
<thead>
<tr>
<th>C++ Syntax</th>
<th>HRESULT put_ResonatBWVideoBWRatio(double ratio);</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HRESULT get_ResonatBWVideoBWRatio(double* ratio);</td>
</tr>
<tr>
<td>Interface</td>
<td>ISpectrumAnalyzer</td>
</tr>
</tbody>
</table>


ResponseCorrectedPorts Property

Description
Sets and returns the selected ports to be corrected with enhanced response calibration. Learn more.

Note: The CorrectionSubsettingState must be set to ON to enable the response command.

VB Syntax
value = corrMethods.ResponseCorrectedPorts

Variable
(value) - Description

value (Variant) Variable to store a comma separated list of ports to include for enhanced response correction.

corrMethods (Type) - CorrectionMethods (object)

Return Type
Variant

Default
All ports included

Examples
Example #1:
16-port VNA with an active 16-port calibration

corrMethods.CorrectionSubsettingState = True
fullportlist = Array(1,2,3,4,5,6)
fullportlist = corrMethods.FullyCorrectedPorts
responseportlist = Array(7,8)
responseportlist = corrMethods.ResponseCorrectedPorts

Result: Full correction on ports 1-6
Enhanced response corrected for parameters involving ports 7 and 8
No correction for ports 9-16

Example #2:
16-port VNA with an active 16-port calibration

corrMethods.CorrectionSubsettingState = True
fullportlist = Array(0)
fullportlist = corrMethods.FullyCorrectedPorts
responseportlist = Array(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16)
responseportlist = corrMethods.ResponseCorrectedPorts

Result: Enhanced response correction for parameters involving any ports

C++ Syntax
HRESULT get_ResponseCorrectedPorts(VARIANT *portList);
HRESULT put_ResponseCorrectedPorts(VARIANT portList);
Interface  ICorrectionMethods2
### Reverse2PortAdapter Property

**Description**  
Set and read whether or not to reverse ports on a 2-port fixture or adapter to be de-embedded.

**VB Syntax**  
```
fixture.Reverse2PortAdapter (port) = bool
```

**Variable (Type) - Description**
- `fixture`  
  A Fixturing (object)
- `port`  
  VNA port number for which SNP file is to be de-embedded.
- `bool`  
  `True` - Reverse ports.
  
  `False` - Do NOT reverse ports.

**Return Type**  
Boolean

**Default**  
False

**Examples**
```
fixture.Reverse2PortAdapter = True
value = fixture.Reverse2PortAdapter  'Read
```

**C++ Syntax**
```
HRESULT get_Reverse2PortAdapter(short portNum, VARIANT_BOOL *pRev);
HRESULT put_Reverse2PortAdapter(short portNum, VARIANT_BOOL bRev);
```

**Interface**  
IFixturing6
### ReverseLinearPowerLevel Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the reverse power level to the DUT. This is applied to the DUT output port when making reverse measurements like S22.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>gca.ReverseLinearPowerLevel = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) A GainCompression (object) &lt;br&gt; (double) Reverse power level in dBm. Choose a value from +30 to (-30).</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>-5</td>
</tr>
<tr>
<td>Examples</td>
<td><code>gca.ReverseLinearPowerLevel = -10 'Write</code> &lt;br&gt;<code>LinPwr = gca.ReverseLinearPowerLevel 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_ReverseLinearPowerLevel(double* pVal) &lt;br&gt;HRESULT put_ReverseLinearPowerLevel(double newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IGainCompression</td>
</tr>
</tbody>
</table>
### RFOffOnReceiverOverload Property

**Description**
Set and return whether to turn source power OFF when a receiver is overloaded.

**VB Syntax**
```vbnet
pref.RFOffOnReceiverOverload = value
```

**Variable**
- **pref** (Type) - Description
  - A Preferences (object)
- **value** (Boolean) - Choose from:
  - **True**  Turn OFF source power to ALL ports when a receiver is overloaded.
  - **False** Power remains ON when a receiver is overloaded.

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
pref.RFOffOnReceiverOverload = True 'Write
value = pref.RFOffOnReceiverOverload 'Read
```

**C++ Syntax**
```cpp
HRESULT get_RFOffOnReceiverOverload (VARIANT_BOOL* preference);
HRESULT put_RFOffOnReceiverOverload (VARIANT_BOOL val)
```

**Interface**
IPreferences12
# RoleDevice Property

This command replaces AssignSourceToRole Method and GetSourceByRole Method. Set and return the source to be used in the specified role. For example, use this command to set a source name to be used as the RF2 tone for a Swept IMD channel.

**VB Syntax**

\[\text{chan.RoleDevice}(\text{role}) = \text{source}\]

**Variable (Type) - Description**

- **chan** (A Channel (object))
- **role** (String) Role of the source. Not context-sensitive. Use DefinedRoles to read the valid roles for the channel.
- **source** (String) Source name to be used in the specified role. Use extDevices.Items to read a list of configured sources.

**Return Type** String

**Default** Not Applicable

**Examples**

- \[\text{chan.RoleDevice}("RF2") = "MyEsg" \] 'Write
- \[\text{source} = \text{chan.RoleDevice}("RF2")\] 'Read

**C++ Syntax**

```cpp
HRESULT get_RoleDevice(BSTR role BSTR *source);
HRESULT put_RoleDevice(BSTR role BSTR source);
```

**Interface** IChannel22
### SADDataThreshold Property

**Description**
Sets or returns the SA data threshold.

**VB Syntax**

```vbnet
seg.SADDataThreshold = value
```

**Variable**
- **seg**: A Segment (object)
- **value**: (Double) Data threshold (in dBm).

**Return Type**
Double

**Default**
-60 dBm

**Examples**

```vbnet
seg.SADDataThreshold = -60 'Write
value = seg.SADDataThreshold 'Read
```

**C++ Syntax**

```cpp
HRESULT get_SADDataThreshold(double* pVal);
HRESULT put_SADDataThreshold(double pVal);
```

**Interface**
ISegment5
## SADDataThresholdOption Property

**Description**
Specifies whether SA Data Threshold can be set independently for each segment.

**VB Syntax**
```
segs.SADDataThresholdOption = value
```

**Variable**

- `segs` A Segments collection *(object)*
- `value` *(boolean)*

- **True** - turns SA Data Threshold control ON.
- **False** - turns SA Data Threshold control OFF.

**Return Type**
Boolean

**Default**
False

**Examples**
```
segs.SADDataThresholdOption = True 'Write
value = SADDataThresholdOption 'Read
```

**C++ Syntax**

- `HRESULT get_SADDataThresholdOption(VARIANT_BOOL *value)`
- `HRESULT put_SADDataThresholdOption(VARIANT_BOOL value)`

**Interface**
ISegments8
## SafeMode Property

**Description**
Sets and returns the state of Safe Mode.

**VB Syntax**
\[ RxLevel.SafeMode(srcPort) = value \]

**Variable**
(Type) - Description

- **RxLevel**
  A ReceiverLeveling Object

- **value**
  (Boolean) Choose from:

  - **True** - Safe mode ON
  - **False** - Safe mode OFF

**srcPort**
(Long Integer) Source port for which to set the Safe Mode state for Receiver Leveling.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

**Return Type**
Variant Boolean

**Default**
False

**Examples**
\[
\begin{align*}
\text{rxLevel.SafeMode}(1) &= \text{True} \quad \text{Write} \\
\text{value} &= \text{rxLevel.SafeMode}(2) \quad \text{Read}
\end{align*}
\]

**C++ Syntax**

\[
\begin{align*}
\text{HRESULT get_SafeMode(long port, VARIANT_BOOL* pLevelingSafeMode);} \\
\text{HRESULT put_SafeMode(long port, VARIANT_BOOL LevelingSafeMode);} \\
\text{IReceiverLevelingConfiguration}
\end{align*}
\]
**SafeSweepDCParameter Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the name of the external DC device.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>gca.SafeSweepDCParameter = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td></td>
<td><code>gca</code> - A GainCompression (object)</td>
</tr>
<tr>
<td></td>
<td><code>value</code> - (string) - Name of the external DC device.</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>gca.SafeSweepDCParameter = &quot;MyDCDevice&quot;</code></td>
</tr>
<tr>
<td></td>
<td><code>value = gca.SafeSweepDCParameter</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_SafeSweepDCParameter(BSTR* dcname)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_SafeSweepDCParameter(BSTR dcname)</td>
</tr>
<tr>
<td>Interface</td>
<td>IGainCompression6</td>
</tr>
</tbody>
</table>
### SafeSweepCoarsePowerAdjustment Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the Safe Sweep COARSE power adjustment.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>gca.SafeSweepCoarsePowerAdjustment = value</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>gca</td>
<td>A GainCompression (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Double) Coarse power adjustment setting in dBm. Choose a value from +30 to (-30).</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>3.0</td>
</tr>
<tr>
<td>Examples</td>
<td>gca.SafeSweepCoarsePowerAdjustment = 2.0 'Write</td>
</tr>
<tr>
<td></td>
<td>SSCourse = gca.SafeSweepCoarsePowerAdjustment 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_SafeSweepCoarsePowerAdjustment(double* value)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_SafeSweepCoarsePowerAdjustment(double value)</td>
</tr>
<tr>
<td>Interface</td>
<td>IGainCompression</td>
</tr>
</tbody>
</table>
### SafeSweepEnable Property

**Description**
Set and read the (ON | OFF) state of Safe Sweep mode.

**VB Syntax**
```vbnet
gca.SafeSweepEnable = value
```

**Variable**
- `gca` A `GainCompression (object)`
- `value` (Boolean) Safe Sweep state. Choose from:
  - **False** - Disable Safe Sweep
  - **True** - Enable Safe Sweep

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
gca.SafeSweepEnable = True 'Write
SSEnable = gca.SafeSweepEnable 'Read
```

**C++ Syntax**
```cpp
HRESULT get_SafeSweepEnable(VARIANT_BOOL* value)
HRESULT put_SafeSweepEnable(VARIANT_BOOL  value)
```

**Interface**
IGainCompression
<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the Safe Sweep FINE power adjustment.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>gca.SafeSweepFinePowerAdjustment = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>gca</code></td>
<td>A <code>GainCompression</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Fine power adjustment setting in dBm. Choose a value from +30 to (-30).</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1.0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>gca.SafeSweepFinePowerAdjustment = 0.1</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>SSfine = gca.SafeSweepFinePowerAdjustment</code> 'Read'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT <code>get_SafeSweepFinePowerAdjustment(double* value)</code></td>
</tr>
<tr>
<td></td>
<td>HRESULT <code>put_SafeSweepFinePowerAdjustment(double value)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IGainCompression</code></td>
</tr>
</tbody>
</table>
### SafeSweepFineThreshold Property

**Description**  
Set and read the compression level at which Safe Sweep changes from the COARSE power adjustment to the FINE power adjustment.

**VB Syntax**  
gca.SafeSweepFineThreshold = value

**Variable**  
**gca** - GainCompression (object)  
**value** - (Double) Threshold setting in dBm. Choose a value from +30 to (-30).

**Return Type**  
Double

**Default**  
0.5 dBm

**Examples**  
gca.SafeSweepFineThreshold = 0.1  
SSThresh = gca.SafeSweepFineThreshold

**C++ Syntax**  
HRESULT get_SafeSweepFineThreshold(double* value)  
HRESULT put_SafeSweepFineThreshold(double value)

**Interface**  
IGainCompression
### SafeSweepMaximumDCLimit Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the maximum limit of the external DC device.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>gca.SafeSweepMaximumDCLimit = value</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>gca</code></td>
<td>A <code>GainCompression</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Maximum DC level.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>-5</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td></td>
</tr>
<tr>
<td><code>gca.SafeSweepMaximumDCLimit = -5</code></td>
<td>'Write'</td>
</tr>
<tr>
<td><code>maxLimit = gca.SafeSweepMaximumDCLimit</code></td>
<td>'Read'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td></td>
</tr>
<tr>
<td>HRESULT get_SafeSweepMaximumDCLimit(double* value)</td>
<td></td>
</tr>
<tr>
<td>HRESULT put_SafeSweepMaximumDCLimit(double value)</td>
<td></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IGainCompression6</td>
</tr>
</tbody>
</table>
### SafeSweepMaximumLimit Property

**Description**
When the VNA port that is connected to the DUT Output measures the specified value, the input power to the DUT is no longer incremented at that frequency.

**VB Syntax**
```vbnet
gca.SafeSweepMaximumLimit = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>gca</code></td>
<td>A GainCompression (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Maximum power level in dBm. Choose a value from -100 to +100.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
100

**Examples**
```vbnet
gca.SafeSweepMaximumLimit = 23 'Write
maxPwr = gca.SafeSweepMaximumLimit 'Read
```

**C++ Syntax**
```cpp
HRESULT get_SafeSweepMaximumLimit(double* value)
HRESULT put_SafeSweepMaximumLimit(double value)
```

**Interface**
IGainCompression
**SAMTReference Property**

**Description**
Sets or returns the SA multitone reference.

**VB Syntax**
```vbnet
seg.SAMTReference = value
```

**Variable (Type) - Description**
- `seg` A Segment (object)
- `value` (Double) Multitone reference (in dBm).

**Return Type**
Double

**Default**
0

**Examples**
```vbnet
seg.SAMTReference = 0 'Write
value = seg.SAMTReference 'Read
```

**C++ Syntax**
```cpp
HRESULT get_SAMTReference(double* pVal);

HRESULT put_SAMTReference(double pVal);
```

**Interface**
ISegment4
## SAMTReferenceFreqOption Property

**Description**: Specifies whether SA Reference Tone can be set independently for each segment.

**VB Syntax**: `segs.SAMTReferenceFreqOption = value`

**Variable (Type) - Description**
- `segs` - A Segments collection (object)
- `value` - (boolean)
  - **True** - turns SA Reference Tone control ON.
  - **False** - turns SA Reference Tone control OFF.

**Return Type**: Boolean

**Default**: False

**Examples**
```
segs.SAMTReferenceFreqOption = True 'Write
value = SAMTReferenceFreqOption 'Read
```

**C++ Syntax**
```
HRESULT get_SAMTReferenceFreqOption(VARIANT_BOOL *value)
HRESULT put_SAMTReferenceFreqOption(VARIANT_BOOL value)
```

**Interface**: ISegments7
## SAResBWList Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a list of Res BW values that are supported by the Spectrum Analyzer app.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = cap.SAResBWList</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td></td>
<td>(Variant) - Variable to store the returned array of Res BW values.</td>
</tr>
<tr>
<td></td>
<td><code>cap</code> A Capabilities (Object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Variant array</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>'Read the supported IFBW values'</td>
</tr>
<tr>
<td></td>
<td>Set <code>app = CreateObject(&quot;AgilentPNA835x.Application&quot;)</code></td>
</tr>
<tr>
<td></td>
<td>Set <code>cap = app.Capabilities</code></td>
</tr>
<tr>
<td></td>
<td><code>list=cap.SAResBWList</code></td>
</tr>
<tr>
<td></td>
<td>For <code>i = 0 To UBound(list)</code></td>
</tr>
<tr>
<td></td>
<td><code>msg = msg &amp; list(i) &amp; vbCrLf</code></td>
</tr>
<tr>
<td></td>
<td>Next</td>
</tr>
<tr>
<td></td>
<td><code>MsgBox msg</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_SAResBWList(Variant *value);</td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities8</td>
</tr>
</tbody>
</table>
## SaturationLevel Property

**Description**
Set and read the deviation dB from the maximum Pout. This is the point of saturation.

This value is used for **Compression Method**: Compression from Saturation.

**VB Syntax**

```vbnet
gca.SaturationLevel = value
```

**Variable**

<table>
<thead>
<tr>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>gca</code></td>
<td>A GainCompression <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Double)</em> - Saturation level in dB. Choose a value greater than 0.01 dB.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
.1 dB

**Examples**

```vbnet
gca.SaturationLevel = .5 'Write
satLevel = gca.SaturationLevel 'Read
```

**C++ Syntax**

```
HRESULT get_SaturationLevel(double* pVal)
HRESULT put_SaturationLevel(double newVal)
```

**Interface**
IGainCompression3
### SAVectorAverage Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the SA vector average points.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>seg.SAVectorAverage = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description <strong>seg</strong> - A Segment <em>(object)</em> <strong>value</strong> - <em>(Long)</em> Vector average points.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>seg.SAVectorAverage = 10</code> 'Write'</td>
</tr>
<tr>
<td><strong>value</strong></td>
<td><code>value = seg.SAVectorAverage</code> 'Read'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_SAVectorAverage(long* pVal);</code> <code>HRESULT put_SAVectorAverage(long pVal);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISegment4</td>
</tr>
</tbody>
</table>
### SAVectorAverageOption Property

**Description**: Specifies whether SA Video Bandwidth can be set independently for each segment.

**VB Syntax**: `segs.SAVectorAverageOption = value`

**Variable**
- **Type**: Segment collection (`object`)
- **Value**: Boolean
  - `True` - turns SA Vector Averaging control ON.
  - `False` - turns SA Vector Averaging control OFF.

**Return Type**: Boolean

**Default**: False

**Examples**
- `segs.SAVectorAverageOption = True 'Write`
- `value = SAVectorAverageOption 'Read`

**C++ Syntax**
- `HRESULT get_SAVectorAverageOption(VARIANT_BOOL *value)`
- `HRESULT put_SAVectorAverageOption(VARIANT_BOOL value)`

**Interface**: ISegments7
### SAVideoAverageOption Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies whether SA Video Bandwidth can be set independently for each segment.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>segs.SAVideoAverageOption = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>segs</code></td>
<td>A Segments collection <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(boolean)</em></td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>turns SA Video Bandwidth control ON.</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>turns SA Video Bandwidth control OFF.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>segs.SAVideoAverageOption = True 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = SAVideoAverageOption 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_SAVideoAverageOption(VARIANT_BOOL *value)`</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_SAVideoAverageOption(VARIANT_BOOL value)`</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISegments8</td>
</tr>
</tbody>
</table>
## SAVideoBandwidth Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the SA video bandwidth.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>seg.SAVideoBandwidth = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td>seg</td>
<td>A Segment <em>(object)</em></td>
</tr>
<tr>
<td>value</td>
<td><em>(Double)</em> Video bandwidth (in Hz).*</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1E6 Hz</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>seg.SAVideoBandwidth = 1E6 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = seg.SAVideoBandwidth 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_SAVideoBandwidth(double* pVal);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_SAVideoBandwidth(double pVal);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISegment5</td>
</tr>
</tbody>
</table>
### SB_BalPortNegative Property

**Description**  
With a Single-ended - Balanced topology, returns the VNA port number that is connected to the Negative side of the DUT's Balanced Port.

Use **SetSBPorts Method** to set the port mapping for a Single-Ended - Balanced topology.

**VB Syntax**  
```
var = balTopology.SB_BalPortNegative
```

**Variable (Type) - Description**  
- **var** (Long Integer) Variable to store the returned value.

**balTopology**  
A **BalancedTopology (object)**

**Return Type**  
Long Integer

**Default**  
Not Applicable

**Examples**  
```
variable = balTop.SB_BalPortNegative  'Read
```

**C++ Syntax**  
```
HRESULT get_SB_BalPortNegative(long *bVal)
```

**Interface**  
IBalancedTopology
SB_BalPortPositive Property

Description
With a Single-ended - Balanced topology, returns the VNA port number that is connected to the Positive side of the DUT's Balanced Port.

Use SetSBPorts Method to set the port mapping for a Single-Ended - Balanced topology.

VB Syntax
```vbnet
var = balTopology.SB_BalPortPositive
```

Variable (Type) - Description
- `var` (Long Integer) Variable to store the returned value.

`balTopology` A BalancedTopology (object)

Return Type Long Integer

Default Not Applicable

Examples
```vbnet
variable = balTop.SB_BalPortPositive  'Read
```

C++ Syntax
```c++
HRESULT get_SB_BalPortPositive(long *bVal)
```

Interface IBalancedTopology
**SB_SEPort Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>With a Single-ended - Balanced topology, returns the VNA port number that is connected to the DUT's Single-ended port.</th>
</tr>
</thead>
</table>

Use **SetSBPorts Method** to set the port mapping for a Single-Ended - Balanced topology.

**VB Syntax**

```
var = balTopology.SB_SEPort
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>var</td>
<td>Long Integer</td>
<td>Variable to store the returned value.</td>
</tr>
</tbody>
</table>

**balTopology**

A **BalancedTopology (object)**

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Long Integer</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Default</th>
<th>Not Applicable</th>
</tr>
</thead>
</table>

**Examples**

```
variable = balTopology.SB_SEPort 'Read
```

**C++ Syntax**

```
HRESULT get_SB_SEPort(long *bVal)
```

**Interface**

IBalancedTopology
 SBalMeasurement Property

**Description**
Sets and returns the measurement for the Single-Ended - Balanced topology.

**VB Syntax**

```vbnet
balMeas.SBalMeasurement = value
```

**Variable (Type) - Description**

- `balMeas` A `BalancedMeasurement` (object)
- `value` (String) - Single-ended - Balanced Measurement parameter. Not case-sensitive. Choose from:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sss11</td>
<td>Ssd12</td>
<td>Ssc12</td>
</tr>
<tr>
<td>Sds21</td>
<td>Sdd22</td>
<td>Sdc22</td>
</tr>
<tr>
<td>Scs21</td>
<td>Scd22</td>
<td>Scc22</td>
</tr>
<tr>
<td>Imb</td>
<td>CMRR1</td>
<td>CMRR2</td>
</tr>
<tr>
<td></td>
<td>(Sds21/Scs21)</td>
<td>(Ssd12/Ssc12)</td>
</tr>
</tbody>
</table>

**Return Type**
Sss11

**Default**
Not Applicable

**Examples**

```vbnet
balMeas.SBalMeasurement = "Ssd12" "Write
variable = balMeas.SBalMeasurement "Read
```

**C++ Syntax**

```c++
HRESULT get_SBalMeasurement(BSTR *pVal)
HRESULT put_SBalMeasurement(BSTR newVal)
```

**Interface**
IBalancedMeasurement
## ScaleCouplingMethod Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the method of scale coupling.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>win.ScaleCouplingMethod = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>Variable (Type) - Description</strong></td>
</tr>
<tr>
<td><code>win</code></td>
<td>An NAWindow (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum NAScaleCouplingMethod)</td>
</tr>
<tr>
<td><strong>0</strong></td>
<td><code>naScaleCouplingOff</code> - Scale Coupling is Off</td>
</tr>
<tr>
<td><strong>1</strong></td>
<td><code>naScaleCouplingWindow</code> - Traces within selected windows share scaling</td>
</tr>
<tr>
<td><strong>2</strong></td>
<td><code>naScaleCouplingAll</code> - Scaling is shared among traces in all selected windows</td>
</tr>
<tr>
<td>Select windows using ScaleCouplingState Property</td>
<td></td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum</td>
</tr>
<tr>
<td>Default</td>
<td>0 - <code>naScaleCouplingOff</code></td>
</tr>
<tr>
<td>Examples</td>
<td><code>win.ScaleCouplingMethod = naScaleCouplingWindow</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>method = app.ActiveNAWindow.ScaleCouplingMethod</code> 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_ScaleCouplingMethod(tagNAScaleCouplingMethod* couplingMethod);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_ScaleCouplingMethod(tagNAScaleCouplingMethod couplingMethod);</td>
</tr>
<tr>
<td>Interface</td>
<td>INAWindow2</td>
</tr>
</tbody>
</table>
## ScaleCouplingState Property

**Description**
Enables and disables scale coupling for the window.

Use `ScaleCouplingMethod` to select the coupling method.

**VB Syntax**

```vbnet
win.ScaleCouplingState = bool
```

**Variable**

- **(Type)** - Description
  - `win` - An `NAWindow` (object).
  - `bool` - (Boolean)
    - **False** - NO scale coupling for this window.
    - **True** - Scale coupling enabled for this window.

**Return Type**
Boolean

**Default**
True

**Examples**

```vbnet
win.ScaleCouplingState = false 'Write
coupled = app.ActiveNAWindow.ScaleCouplingState 'Read
```

**C++ Syntax**

```cpp
HRESULT get_ScaleCouplingState(VARIANT_BOOL *pVal);

HRESULT put_ScaleCouplingState(VARIANT_BOOL newVal)
```

**Interface**
INAWindow2
# Scope Property

**Description**
Sets or returns the scope of a trigger signal. This determines whether a trigger signal affects a single channel or all channels in the VNA.

**Note:** Trigger Modes Point and EverySweep require that Trigger.Scope be set to naChannelTrigger.

**VB Syntax**
```
trigsetup.Scope = value
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trigsetup</td>
<td>A TriggerSetup (object)</td>
</tr>
<tr>
<td>value</td>
<td>(enum NATriggerType) - Trigger type. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naGlobalTrigger - a trigger signal is applied to all triggerable channels</td>
</tr>
<tr>
<td></td>
<td>1 - naChannelTrigger - a trigger signal is applied to the current channel. The next trigger signal will be applied to the next channel; not necessarily the next channel in numeric sequence (1-2-3-4 and so forth).</td>
</tr>
<tr>
<td></td>
<td>2 - naActiveChannelTrigger - a trigger signal is sent only to the active channel.</td>
</tr>
</tbody>
</table>

**Return Type**
Long Integer

**Default**
naGlobalTrigger

**Examples**
```
trigsetup.Scope = naGlobalTrigger 'Write
trigtyp = trigsetup.Scope 'Read
```

**C++ Syntax**
```
HRESULT get_Scope(tagNATriggerType *pTrigger)
HRESULT put_Scope(tagNATriggerType trigger)
```

**Interface**
ITriggerSetup
SearchFunction Property

Description
Emulates the Tracking function in the marker search dialog box. The value you choose for SearchFunction will determine the type of search that takes place when the Tracking property is set true.

The tracking function finds the selected search function every sweep. In effect, turning Tracking ON is the same as executing one of the "Search..." methods (such as SearchMin, SearchMax) for every sweep.

VB Syntax
mark/SearchFunction = value

Variable
(mark) - Description
mark A Marker (object)
value (enum NAMarkerFunction) - search function. Choose from:

0 - naMarkerFunction_None
1 - naMarkerFunction_Min
2 - naMarkerFunction_Max
3 - naMarkerFunction_Target
4 - naMarkerFunction_NextPeak
5 - naMarkerFunction_PeakRight
6 - naMarkerFunction_PeakLeft
7 - naMarkerFunction_Compression

Return Type
Long Integer

Default
0 - naMarkerFunction_None

Examples
mark/SearchFunction = naMarkerFunction_Target 'When this marker is set to track, it will track the Target value.

searchfunction = mark/SearchFunction 'Read

C++ Syntax
HRESULT get_SearchFunction(tagNAMarkerFunction *pVal)
HRESULT put_SearchFunction(tagNAMarkerFunction newVal)

Interface
IMarker
SearchFailures Property

Description
Returns a comma-separated list of the frequency indexes that were out of
tolerance for SMART Sweep mode, or at the power limit for 2D Sweep mode.
Zero (0) is the first frequency data point.

Must be Single triggered. Invalid results occur if the GCA channel is
continuously sweeping.

VB Syntax
value = gca.SearchFailures

Variable (Type) - Description
value (Double) Variable to store the returned data.
gca A GainCompression (object)

Return Type
Returns a comma-separated list of frequency indexes.

Default
Not applicable

Examples
SFA = gca.SearchFailures 'Read

C++ Syntax
HRESULT get_SearchFailures(VARIANT* value)

Interface
IGainCompression
## SearchOccupiedBWMinFreq Property

**Description**  
Set and read the minimum search frequency to use during an Occupied BW search measurement. Power below this frequency is ignored.

**VB Syntax**  
`sas .SearchOccupiedBWMinFreq = value`

**Variable** (Type) - Description  
- `sa`  
  A `SpectrumAnalyzer` (object)
- `value`  
  `(Double)` Minimum search frequency value.

Learn about these settings.

**Return Type**  
Double

**Default**  
250 MHz

**Examples**  
- `sa .SearchOccupiedBWMinFreq = 300e6`  
  *Write*
- `value = sa .SearchOccupiedBWMinFreq`  
  *Read*

**C++ Syntax**  
```cpp
HRESULT put_SearchOccupiedBWMinFreq(double minfreq);
HRESULT get_SearchOccupiedBWMinFreq(double* minfreq);
```

**Interface**  
`ISpectrumAnalyzer4`

## SearchSummary Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the status of a compression search. This command can be used to indicate when a GCA search is complete.</th>
</tr>
</thead>
</table>

**Note:** The returned value reflects the current state of the GCA compression search which can vary when in continuous sweep. This command is intended to be used with `chan.Single` (trigger).

### VB Syntax

```vbnet
value = gca.SearchSummary
```

### Variable (Type) - Description

**value** *(Enum)* Variable to store the returned value.

- **0 - naSearchNotDone** - Acquisition is still in process.
- **1 - naSearchSucceeded** - Acquisition is complete and compression value found for all frequency points.
- **2 - naSearchFailed** - Acquisition is complete and unable to find the compression value at one or more frequency points.

### gca

A `GainCompression` *(object)*

### Return Type

Enum

### Default

Not Applicable

### Examples

```vbnet
sum = gca.SearchSummary
```

### C++ Syntax

```c++
HRESULT get_SearchSummary(enum naGCASearchSummary* value)
```

### Interface

`IGainCompression`
### SecurityLevel Property

**Description**
Controls the display of frequency information on the VNA screen and printouts.

**VB Syntax**
```vbnet
app.SecurityLevel value
```

**Variable**
- **Type** - Description
- **app** An Application (object)
- **value** (enum NASecurityLevel) - Choose from:
  - **0 - naNoSecurity** ALL frequency information is displayed.
  - **1 - naLowSecurity** NO frequency information is displayed. Frequency information can be redisplayed using the Security Setting dialog box or this command.
  - **2 - naHighSecurity** LOW setting plus GPIB console is disabled. Frequency information can be redisplayed ONLY by performing a Preset, recalling an instrument state with None or Low security settings, or using this command.
  - **3 - naExtraSecurity** HIGH setting plus:
    - ASCII data saving is disabled. Same method to redisplay frequency information as HIGH setting.
    - Mixer setup files (*.mxr) can NOT be saved.

**Return Type**
Long Integer

**Default**
0 - None

**Examples**
```vbnet
app.SecurityLevel = naLowSecurity 'Write
level = app.SecurityLevel 'Read
```

**C++ Syntax**
```cpp
HRESULT get_NASecurityLevel(tagNASecurityLevel *level);
HRESULT put_NASecurityLevel(tagNASecurityLevel level);
```

**Interface**
IApplication4
### SegmentNumber Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of the current segment, PowerSensorCalFactorSegment or PowerLossSegment object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>seg.SegmentNumber</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>seg</code></td>
<td>Any of the following objects:</td>
</tr>
<tr>
<td></td>
<td>A <code>Segment</code> (Object)</td>
</tr>
<tr>
<td></td>
<td>A <code>PowerSensorCalFactorSegment</code> (Object)</td>
</tr>
<tr>
<td></td>
<td>A <code>PowerSensorCalFactorSegmentPMAR</code> Object (Object)</td>
</tr>
<tr>
<td></td>
<td>A <code>PowerLossSegment</code> (Object)</td>
</tr>
<tr>
<td></td>
<td>A <code>PowerLossSegmentPMAR</code> (Object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long Integer</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>segNum = seg.SegmentNumber</code> <code>returns the segment number -Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_SegmentNumber(long *pVal)</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ISegment</td>
</tr>
<tr>
<td></td>
<td>IPowerSensorCalFactorSegment</td>
</tr>
<tr>
<td></td>
<td>IPowerLossSegment</td>
</tr>
</tbody>
</table>
### SegmentCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of segments on the <code>Applied mixer</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = conv.SegmentCount</code></td>
</tr>
<tr>
<td>Variable (Type)</td>
<td><code>value</code> (Long integer) Variable in which to store the returned segment count.</td>
</tr>
<tr>
<td></td>
<td><code>conv</code> A <code>Converter Object</code></td>
</tr>
<tr>
<td>Return Type</td>
<td>Long integer</td>
</tr>
<tr>
<td>Default</td>
<td>1 Segment is created on new converter objects.</td>
</tr>
<tr>
<td>Examples</td>
<td><code>count=mxr.SegmentCount</code></td>
</tr>
<tr>
<td></td>
<td><a href="#">See example program</a></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_SegmentCount(long *value);</code></td>
</tr>
<tr>
<td>Interface</td>
<td><code>IConverter5</code></td>
</tr>
</tbody>
</table>
## SegmentCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of segments used in a limit test. All segments are counted, whether they are on or not.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = limitst.SegmentCount</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long integer) Variable in which to store the returned segment count.</td>
</tr>
<tr>
<td><code>limitst</code></td>
<td>A LimitTest (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1 Segment is created on new limit test objects.</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>count=limitst.SegmentCount</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_SegmentCount(long *value);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ILimitTest</td>
</tr>
</tbody>
</table>
SegmentFixedFrequency Property

Description
Set and read the CW Frequency for mixer segments. The specified SegmentRangeMode must be set to Fixed.

Send Apply before sending a query (read). Learn more.

VB Syntax

```
conv.SegmentFixedFrequency(index,range) = value
```

Variable (Type) - Description

- **conv** (A Converter Object)
- **index** (Long integer) Segment for which fixed frequency is being set. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer.
- **range** (Enum as ConverterFrequencyRange) Range for which fixed frequency is being set. Choose from:
  - 0 - naInputFrequencies - set input frequency
  - 1 - naOutputFrequencies - set output frequency
  - 2 - naLO1Frequencies - set LO1 frequency
  - 3 - naLO2Frequencies - set LO2 frequency
  - 4 - naIFFrequencies - set IF frequency
- **value** (Double) CW Frequency. Choose a value within the frequency range of the VNA.

Return Type
Double

Default
Center frequency of the VNA

Examples
```
mxr.SegmentFixedFrequency(1,0)=1e9 'sets the input frequency to 1 GHz
```

See example program

C++ Syntax

```
HRESULT get_SegmentFixedFrequency(long index, tagConverterFrequencyRange range, double *value);

HRESULT put_SegmentFixedFrequency(long index, tagConverterFrequencyRange value, double value);
```
<table>
<thead>
<tr>
<th>Interface</th>
<th>IConverter5</th>
</tr>
</thead>
</table>

2277
### SegmentFixedPower Property

**Description**  
Set and read the fixed power level for the mixer segment.

**VB Syntax**  
```vbnet
conv.SegmentFixedPower(index, range) = value
```

**Variable (Type) - Description**

- **conv**  
  A Converter Object

- **index**  
  (Long integer) Segment for which power is being set. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer.

- **range**  
  (Enum as ConverterFrequencyRange) Range for which power is being set. Choose from:

  0 - naInputFrequencies - set input power

  1 - naOutputFrequencies - set output power

  2 - naLO1Frequencies - set LO1 power

  3 - naLO2Frequencies - set LO2 power

  4 - naIFFrequencies - set IF power

- **value**  
  (Double) Power level in dBm. Choose a value within the power/attenuation range of the VNA.

**Return Type**  
Double

**Default**

**Examples**  
```vbnet
mxr.SegmentFixedPower(1, 0) = 0 'sets the input power level to 0 dBm
```

**See example program**

**C++ Syntax**

```cpp
HRESULT get_SegmentFixedPower(long index, tagConverterFrequencyRange range, double *value);

HRESULT put_SegmentFixedPower(long index, tagConverterFrequencyRange value, double value);
```

**Interface**  
IConverter5
# SegmentIFBandwidth Property

Sets and returns the IF Bandwidth for the sweep segment.

Send **Apply** before sending a query (read). Learn more.

**VB Syntax**

```
conv.SegmentIFBandwidth(index, value)
```

**Variable (Type) - Description**

- **conv**: A Converter Object
- **index**: (Long integer) Segment for which IF Bandwidth is to be set. Choose a segment between 1 and the current segment count. Use **SegmentCount Property** to read the current count in the Applied Mixer.
- **value**: (Long integer) IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the VNA model. See the lists. If an invalid number is specified, the analyzer will round up to the closest valid number.

**Return Type**: Long integer

**Default**: 100 kHz

**Examples**

```
mxr.SegmentIFBandwidth(1,3)=10e3 'Sets IFBW to 10 kHz
```

See example program

**C++ Syntax**

```
HRESULT get_SegmentIFBandwidth(long index, long * val);
HRESULT put_SegmentIFBandwidth(long index, long val);
```

**Interface**: IConverter5
## SegmentIsInputGreaterThanLO Property

**Description**
Set and read whether to use the Input frequency that is greater than the LO or less than the LO.

Send Apply before sending a query (read). Learn more.

**VB Syntax**
```vbnet
conv.SegmentIsInputGreaterThanLO(index, range) = value
```

**Variable**
- **conv**  
  A Converter Object
- **index**  
  (Long integer) Segment which is being set. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer.
- **range**  
  (Enum as ConverterFrequencyRange) Range for which value is being set. Choose from:
  - 0 - **naInputFrequencies** - set input power
  - 1 - **naOutputFrequencies** - set output power
  - 2 - **naLO1Frequencies** - set LO1 power
  - 3 - **naLO2Frequencies** - set LO2 power
  - 4 - **naIFFrequencies** - set IF power
- **value**  
  (Boolean) - Choose from the following:
  - **True** - Use the Input that is Greater than the specified LO.
  - **False** - Use the Input that is Less than the specified LO.

**Return Type**
Boolean

**Default**
True

**Examples**
```vbnet
mxr.SegmentIsInputGreaterThanLO(1,2=1)'sets segment 1, LO1 to Input greater
```

**C++ Syntax**
```cpp
HRESULT get_SegmentIsInputGreaterThanLO(long index,  
tagConverterFrequencyRange range, VARIANT_BOOL *value);
```

```cpp
HRESULT put_SegmentIsInputGreaterThanLO(long index,  
tagConverterFrequencyRange range, VARIANT_BOOL *value);
```
tagConverterFrequencyRange value, VARIANT_BOOL value);

Interface IConverter5
### SegmentMixingMode Property

**Description**  
Set and read whether the mixing mode (high side or low side) for the mixer segment.

Send **Apply** before sending a query (read). Learn more.

**VB Syntax**  
`conv.SegmentMixingMode(index, range) = value`

**Variable (Type) - Description**

- **conv**  
  A Converter Object

- **index**  
  (Long integer) Segment for which mixing mode being set. Choose a segment between 1 and the current segment count. Use **SegmentCount Property** to read the current count in the Applied Mixer.

- **range**  
  (Enum as ConverterFrequencyRange) Range for which mixing mode is being set. Choose from:
  
  1 - **naOutputFrequencies** - sets output frequencies for 1-stage or 2-stage mixers.

  4 - **naIFFrequencies** - sets IF frequencies for the first LO in 2-stage mixers.

- **value**  
  (Enum as ConverterSideBand) Choose from:

  0 - **naLowSide**  
  Input minus LO

  1 - **naHighSide**  
  Input plus LO

**Return Type**  
Enum as ConverterSideBand

**Default**  
0 - **naLowSide**

**Examples**

```vbnet
mxr.SegmentMixingMode(1,1)=1 'sets segment 1 output frequencies to Highside mixing.
```

See example program

**C++ Syntax**

```cpp
HRESULT get_SegmentMixingMode(long index, tagConverterFrequencyRange range, tagConverterSideBand *value);

HRESULT put_SegmentMixingMode(long index, tagConverterFrequencyRange value, tagConverterSideBand value);
```

**Interface**  
IConverter5
# SegmentPoints Property

**Description**
Sets and returns the number of data points to be measured in the sweep segment.

Send Apply before sending a query (read). Learn more.

**VB Syntax**

```vbnet
conv.SegmentPoints(index) = value
```

**Variable** *(Type) - Description*

- **conv** A Converter Object
- **index** (Long integer) Segment for which points is to be set. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer.
- **value** (Long integer) - Choose a value between 1 and the maximum number of data points allowed in the VNA. This is also the total number of points allowed for ALL segments.

**Return Type**
Long integer

**Default**
21

**Examples**

```vbnet
mxr.SegmentPoints(1)=3  'Sets 3 points for segment 1.
```

See example program

**C++ Syntax**

```cpp
HRESULT get_SegmentPoints(long index, long * val);

HRESULT put_SegmentPoints(long index, long val);
```

**Interface**
IConverter5
## SegmentRangeMode Property

**Description**
Sets or returns the segment sweep mode (Swept or Fixed) for the specified range (Input/LO/Output).

Send `Apply` before sending a query (read). [Learn more.](#)

### VB Syntax

```
conv.SegmentRangeMode(index,range) = value
```

### Variable *(Type) - Description*

- **conv**
  - A `Converter Object`

- **index**
  - (Long integer) Segment for which mixing mode being set. Choose a segment between 1 and the current segment count. Use `SegmentCount Property` to read the current count in the **Applied Mixer**.

- **range**
  - (Enum as `ConverterFrequencyRange`) Range for which sweep mode is being set. Choose from:
    - 0 - `naInputFrequencies` - set input sweep mode.
    - 1 - `naOutputFrequencies` - set output sweep mode.
    - 2 - `naLO1Frequencies` - set LO1 sweep mode.
    - 3 - `naLO2Frequencies` - set LO2 sweep mode.
    - 4 - `naIFFrequencies` - set IF sweep mode.

- **value**
  - (Enum as `NARangeMode`) Choose from:
    - 0 - `naSwept` Range is swept
    - 1 - `naFixed` Range is fixed.

### Return Type

Enum as `NARangeMode`

### Default

- Input and Output - `naSwept`
- LO (1 and 2) - `naFixed`

### Examples

```
mxr.SegmentRangeMode(1,1)=0  'sets segment 1 output range to swept.
```

[See example program](#)

### C++ Syntax

```
HRESULT get_SegmentRangeMode(long index, tagConverterFrequencyRange range, tagNARangeMode *value);
```
HRESULT put_SegmentRangeMode(long index,  
tagConverterFrequencyRange value, tagNARangeMode value);

Interface  IConverter5
### SegmentStartFrequency Property

**Description**
Set and read the start frequency for the mixer segment. The specified `SegmentRangeMode` must be set to Swept.

Send **Apply** before sending a query (read). Learn more.

**VB Syntax**

```vbnet
conv.SegmentStartFrequency(index, range) = value
```

**Variable (Type) - Description**

- **conv** (A Converter Object)
- **index** (Long integer) Segment for which start frequency is being set. Choose a segment between 1 and the current segment count. Use `SegmentCount Property` to read the current count in the Applied Mixer.
- **range** (Enum as `ConverterFrequencyRange`) Range for which start frequency is being set. Choose from:
  - 0 - naInputFrequencies - set input frequency
  - 1 - naOutputFrequencies - set output frequency
  - 2 - naLO1Frequencies - set LO1 frequency
  - 3 - naLO2Frequencies - set LO2 frequency
  - 4 - naIFFrequencies - set IF frequency
- **value** (Double) Start frequency. Choose a value within the frequency range of the VNA.

**Return Type**
Double

**Default**
Start frequency of the VNA

**Examples**

```vbnet
mxr.SegmentStartFrequency(1, 0)=1e9 'sets the input start frequency to 1 GHz
```

See example program

**C++ Syntax**

```c++
HRESULT get_SegmentStartFrequency(long index, tagConverterFrequencyRange range, double *value);

HRESULT put_SegmentStartFrequency(long index, tagConverterFrequencyRange value, double value);
```
Interface  IConverter5
## SegmentState Property

**Description**
Sets and returns the ON|OFF state of a sweep segment. Off segments are not included in a segment sweep. Send **Apply** before sending a query (read). Learn more.

**VB Syntax**

```
conv.SegmentState(index) = state
```

**Variable** (Type) - Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>conv</code></td>
<td>A Converter Object</td>
<td></td>
</tr>
<tr>
<td><code>index</code></td>
<td>(Long integer) Segment to set ON or OFF. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer.</td>
<td></td>
</tr>
<tr>
<td><code>state</code></td>
<td>(Boolean) - Choose from:</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Segment ON</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Segment OFF</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**
Boolean

**Default**
ON when added.

**Examples**

```vba
mxr.SegmentState(1)=False 'Turns segment 1 OFF.
```

See example program

**C++ Syntax**

```cpp
HRESULT get_SegmentState(long index, VARIANT_BOOL * val);

HRESULT put_SegmentState(long index, VARIANT_BOOL val);
```

**Interface**
IConverter5
**SegmentStopFrequency Property**

**Description**
Set and read the Stop frequency for the mixer segment. The specified `SegmentRangeMode` must be set to Swept.

Send **Apply** before sending a query (read). Learn more.

**VB Syntax**
```
conv.SegmentStopFrequency(index, range) = value
```

**Variable**
- **`conv`** - Description
  A Converter Object
- **`index`** - (Long integer) Segment for which Stop frequency is being set. Choose a segment between 1 and the current segment count. Use `SegmentCount Property` to read the current count in the Applied Mixer.
- **`range`** - (Enum as ConverterFrequencyRange) Range for which Stop frequency is being set. Choose from:
  - **0 - naInputFrequencies** - set input frequency
  - **1 - naOutputFrequencies** - set output frequency
  - **2 - naLO1Frequencies** - set LO1 frequency
  - **3 - naLO2Frequencies** - set LO2 frequency
  - **4 - naIFFrequencies** - set IF frequency
- **`value`** - (Double) Stop frequency. Choose a value within the frequency range of the VNA.

**Return Type**
Double

**Default**
Stop frequency of the VNA

**Examples**
```
mxr.SegmentStopFrequency(1,0)=1e9 'sets the input Stop frequency to 1 GHz
```

See example program

**C++ Syntax**
```
HRESULT get_SegmentStopFrequency(long index, tagConverterFrequencyRange range, double *value);

HRESULT put_SegmentStopFrequency(long index, tagConverterFrequencyRange value, double value);
```
Interface  IConverter5
SelectPort Property

Description
Sets and returns a port mapping for a single port. If this command creates a conflict with an existing port, the VNA will resolve the conflict.

Note: This command is currently not supported for the Z5623AK44.

VB Syntax
```vbnet
  tset.SelectPort(chNum, portNum) = portValue
```

Variable (Type) - Description
- `tset` A `TestsetControl` object.
- `chNum` (Long) Channel number of the measurement.
- `portNum` (Long) Physical port number to map.
- `portValue` (Long) Logical port value to assign

Return Type
Long

Default
Not Applicable

Examples
See External Testset Program

C++ Syntax
```cpp
  HRESULT get_SelectPort(long channelNum, long PortNum long *outPort);
  HRESULT put_SelectPort(long channelNum, long PortNum long outPort);
```

Interface
`ITestsetControl`
## SensorIndex Property

**Description**  
For dual sensor power meters, sets and returns the power sensor channel (1 or 2) to be used.

**VB Syntax**  
`pwrSensor.SensorIndex = value`

**Variable**  
**Type** - Description  
pwrSensor  
A `PowerSensor` (Object) or  
a `PowerSensorAsReceiver` (Object)  

**value**  
**Long** - Power Meter channel. Choose from:  
1 - Sensor A  
2 - Sensor B

**Return Type**  
Long

**Default**  
1

**Examples**  
`pwrSensor.SensorIndex = False` 'Write

`sensor = pwrSensor.SensorIndex` 'Read

**C++ Syntax**  
HRESULT put_SensorIndex(long newVal);

HRESULT get_SensorIndex(long* pVal);

**Interface**  
IPowerSensor  
IPowerSensorAsReceiver
## SeparatePowerCal Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies whether to use a Thru standard or to use two power sensor connections during the power cal of an SMC calibration. <a href="#">Learn more.</a></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command must be sent immediately after the Initialize command, but before all other calibration properties.</td>
</tr>
</tbody>
</table>

### VB Syntax

```vbnet
smc.SeparatePowerCal = bool
```

### Variable

**Type** - Description

- **smc**  
  An SMCType (object)

- **bool**  
  (Boolean)

**True** - Do NOT use a Thru, but instead perform separate power cals on Input and Output reference planes.

**False** - Perform Cal with Thru standard.

### Return Type

Boolean

### Default

False

### Example

```vbnet
FCAppLib.ISMCType4 SMC = (FCAppLib.ISMCType4)CalMgr.CreateCustomCal("SMC");
SMC.Initialize(chan, true);
if (separatePowerCalIsDesired)
    SMC.SeparatePowerCal = true;
```

### C++ Syntax

```cpp
HRESULT put_SeparatePowerCal(VARIANT_BOOL bValue);
HRESULT get_SeparatePowerCal(VARIANT_BOOL *bValue);
```

### Interface

SMCType4
### ShiftLO Property *(PNA does not support this setting)*

**Description** Sets or returns the Shift LO Property of a Segment. Enable the Shift LO Property setting using the `ShiftLOOption Property` of the Segments collection.

#### VB Syntax

```vbnet
seg.ShiftLO = value
```

#### Variable *(Type) - Description*

- **seg**
  - A Segment *(object)*
- **value** *(boolean)*
  - **True** - Enable shift LO state.
  - **False** - Disable shift LO state.

**Return Type** Boolean

**Default** False

#### Examples

- `seg.ShiftLO = True` *Write*
- `shiftstate = seg.ShiftLO` *Read*

#### C++ Syntax

- `HRESULT get_ShiftLO(VARIANT_BOOL* pVal)`
- `HRESULT put_ShiftLO(VARIANT_BOOL pVal)`

**Interface** ISegment3
ShiftLOOption Property (This command is not supported on PNA-L, PNA, PNA-X)

Description
Enables or disables ShiftLO per segment. Value is set using ShiftLO Property of a segment.

VB Syntax
```
segs.ShiftLOOption = value
```

Variable
(Type) - Description
```
segs  A Segments collection (object)
value (boolean)
  True - Enable shift LO state.
  False - Disable shift LO state.
```

Return Type
Boolean

Default
False

Examples
```
segs.ShiftLOOption = True 'Write
shiftstate = segs.ShiftLOOption 'Read
```

C++ Syntax
```
HRESULT get_ShiftLOOption(VARIANT_BOOL* pVal)
HRESULT put_ShiftLOOption(VARIANT_BOOL pVal)
```

Interface
ISegments6
## ShowKeysToolbarAtPowerOn Property

**Write/Read**

<table>
<thead>
<tr>
<th>Description</th>
<th>This command controls the on/off state of the preference, &quot;On Power-on show Keys toolbar&quot;.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pref.ShowKeysToolbarAtPowerOn = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>pref</code></td>
<td>A <code>Preferences</code> <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Boolean) - Choose from:</em></td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>Display the keys toolbar on power-on.</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>Do not display the keys toolbar on power-on.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pref.ShowKeysToolbarAtPowerOn = True 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = pref.ShowKeysToolbarAtPowerOn 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_ShowKeysToolbarAtPowerOn (VARIANT_BOOL* preference);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_ShowKeysToolbarAtPowerOn (VARIANT_BOOL val);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IPreferences16</code></td>
</tr>
</tbody>
</table>
# ShowStatistics Property

**Description**
Displays and hides the measurement (Trace) statistics (peak-to-peak, mean, standard deviation) on the screen. To display measurement statistics for a narrower band of the X-axis, use `StatisticsRange`.

The analyzer will display either measurement statistics or Filter Bandwidth statistics; not both.

**VB Syntax**
```vbnet
meas.ShowStatistics = value
```

**Variable (Type) - Description**
- `meas` (A Measurement object)
- `value` (boolean) - Boolean value:
  - **True** - Show statistics
  - **False** - Hide statistics

**Return Type**
Boolean

**Default**
**False**

**Examples**
```vbnet
meas.ShowStatistics = True
```

**C++ Syntax**
```cpp
HRESULT put_ShowStatistics(VARIANT_BOOL bState)
```

**Interface**
IMeasurement
ShowProperties Property

Description: Turns ON and OFF the display of the test set control status bar. This status bar indicates the test set that is being controlled and the current port mappings. This setting is turned ON and OFF automatically when the test set is enabled or disabled.

VB Syntax: `tset.ShowProperties = value`

Variable (Type) - Description

- `tset` A TestsetControl object.
- OR
- An E5091Testset object.

- `value` (Boolean)
  - **True** - Turns display of testset properties ON.
  - **False** - Turns display of testset properties OFF.

Return Type: Boolean

Default: **False** (True when test set control is enabled.)

Examples:

- See E5091A Example Program
- See External Testset Program

C++ Syntax:

- `HRESULT get_ShowProperties(VARIANT_BOOL *state);`
- `HRESULT put_ShowProperties(VARIANT_BOOL state);`

Interface: IE5091Testsets

ITestsetControl
**ShowQuickStartOnPreset Property**

**Description**
This command controls the on/off state of the preference, "On PRESET show Quick Start dialog".

**VB Syntax**
```
pref.ShowQuickStartOnPreset = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pref</td>
<td>A Preferences (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Boolean) - Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Display the Quick Start dialog on PRESET.</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Do not display the Quick Start dialog on PRESET.</td>
</tr>
</tbody>
</table>

**Return Type**
Boolean

**Default**
False

**Examples**
```
pref.ShowQuickStartOnPreset = True 'Write
value = pref.ShowQuickStartOnPreset 'Read
```

**C++ Syntax**
```
HRESULT get_ShowQuickStartOnPreset (VARIANT_BOOL* preference);
HRESULT put_ShowQuickStartOnPreset (VARIANT_BOOL val);
```

**Interface**
IPreferences13
SICL Property

Description Allows you to control the VNA via SICL (standard instrument control library). In this mode, the analyzer can receive SCPI commands from the LAN interface or from a program residing on the VNA itself. This command performs the same function as the SICL / GPIB dialog box - SICL Enabled checkbox. See Configuring the analyzer for SICL/VISA.

When SICL is enabled, the VNA VXI-11.2 interface is enabled, and if the VNA hard disk image is new enough to have the VXI-11.3 interface, it also enables that. Learn more about LXI / VXI.

With this method you can augment a test program written using SICL that resides on the VNA so that it will run unattended. An automation script can be written to start the VNA, enable SICL (using the SICL property), and then start the SICL based program.

VB Syntax

```vbnet
app.SICL value
```

Variable (Type) - Description

- **app** An Application (object)
- **value** (Boolean) Choose from:
  - **True** - enable SICL
  - **False** - disable SICL

Return Type Boolean

Default False

Examples

```vbnet
Dim Pna as AgilentPNA835x.Application
Dim siclState as Boolean
Set Pna = CreateObject("AgilentPNA835x.Application")
Pna.SICL = true 'write
siclState = Pna.SICL 'Read
```

C++ Syntax

```cpp
HRESULT get_SICL(VARIANT_BOOL *pVal)
HRESULT put_SICL(VARIANT_BOOL newVal)
```

Interface IApplication5
## SICLAddress Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the VNA SICL address. This is the address used for SICL over LAN.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.SICLAddress = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Integer) SICL Address of the VNA. Choose a value between 0 and 30.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Short Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>16</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>address=app.SICLAddress</code> <em>(Read)</em></td>
</tr>
<tr>
<td></td>
<td><code>app.SICLAddress=16</code> <em>(Write)</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_SICLAddress(short busIndex, short* address);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_SICLAddress(short busIndex,short address);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IApplication8</code></td>
</tr>
</tbody>
</table>
# Simultaneous2PortAcquisition Property

## Description
Specifies whether a 2-port calibration will be done with a single set of standards (one port at a time) or with two sets of standards (simultaneously).

The `AcquireCalStandard2` command uses the same standard index for each calibration class. To specify the calibration standard gender for each port, you must first ensure that the order of calibration class accurately reflects the configuration of your DUT. For example, for a DUT with a male connector on port 1 and a female connector on port 2, order the devices within the S11 classes (A, B, and C) such that the MALE standards are first in the list. Then order the S22 classes specifying the FEMALE standards as the first in the list.

## VB Syntax
```vbnet
cal.Simultaneous2PortAcquisition = state
```

### Variable (Type) - Description
- **cal**: A Calibrator (object)
- **state**: (boolean) - Choose from:
  - **True**: measures 2 ports simultaneously
  - **False**: measures 1 port at a time

### Return Type
Boolean

### Default
True

### Examples
```vbnet
cal.Simultaneous2PortAcquisition = True
```

## C++ Syntax
```c++
HRESULT put_Simultaneous2PortAcquisition( VARIANT_BOOL bTwoSetsOfStandards)

HRESULT Simultaneous2PortAcquisition( VARIANT_BOOL *bTwoSetsOfStandards)
```

## Interface
ICalibrator
**SingleMarkerSearch Property**

**Description**
Set and return whether to use one marker for marker search.

**Enabled behavior:**

- Only one marker is used for bandwidth, notch, PNOP, and PSAT marker searches. The points of interest are marked with a notational UI element, i.e. a small triangle.

- Bandwidth, notch, PNOP, and PSAT marker searches are always tracking. Tracking cannot be disabled.

- One basic search and one advanced search may be set per marker.

- The advanced search is enabled until the user disables the search or a multi-peak or multi-target search is executed.

**Disabled behavior:**

- Bandwidth, notch, PSAT, and PNOP marker searches use multiple markers.

- One advanced marker search is allowed per trace.

- A marker may only perform a basic search or be part of an advanced search. Not both.

- If an advanced marker search is enabled on a trace and then the user performs a basic search, the advanced search is automatically disabled.

- Advanced searches may enable or disable tracking. Only one search may be tracked.

**VB Syntax**

```
pref.SingleMarkerSearch = bool
```

**Variable** *(Type)* - Description

`pref` A *Preferences* *(object)*
bool (boolean)
False - Disable single marker search.
True - Enable single marker search.

Return Type: Boolean
Default: False

Examples:
```c
pref.SingleMarkerSearch = True 'Write
prefer = pref.SingleMarkerSearch 'Read
```

C++ Syntax:
```cpp
HRESULT get_SingleMarkerSearch(VARIANT_BOOL *mkrState)
HRESULT put_SingleMarkerSearch(VARIANT_BOOL mkrState)
```

Interface: IPreferences18
**SmartSweepMaximumIterations Property**

**Description**
Set and read the maximum permitted number of iterations which SMART Sweep may utilize to find the desired compression level, to within the specified tolerance.

**VB Syntax**
```
gca.SmartSweepMaximumIterations = value
```

**Variable**
- **(Type)** - Description
  - `gca` A *GainCompression* (object)
  - `value` (integer) - Maximum number of iterations. Choose a value between 1 and 50.

**Return Type**
Integer

**Default**
20

**Examples**
```
gca.SmartSweepMaximumIterations = 10 'Write

iters = gca.SmartSweepMaximumIterations 'Read
```

**C++ Syntax**
- `HRESULT get_SmartSweepMaximumIterations(int* pVal)`
- `HRESULT put_SmartSweepMaximumIterations(int newVal)`

**Interface**
*IGainCompression*
About SMART Sweep

SmartSweepSettlingTime Property

Description
Set and read the amount of time SMART Sweep will dwell at the first point where the input power changes by the Backoff or X level.

Learn more.

VB Syntax
\texttt{\textit{gca}.SmartSweepSettlingTime = value}

Variable (Type) - Description
\textit{gca} - A GainCompression (object)
\textit{value} - (double) - Settling time in seconds. Choose any positive value.

Return Type
Double

Default
0

Examples
\texttt{\textit{gca}.SmartSweepSettlingTime = .01 \textasciitilde\textit{Write}}

\texttt{sTime = gca.SmartSweepSettlingTime \textasciitilde\textit{Read}}

C++ Syntax
HRESULT get_SmartSweepSettlingTime(double* pVal)

HRESULT put_SmartSweepSettlingTime(double newVal)

Interface
IGainCompression
## SmartSweepShowIterations Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read whether to show intermediate results for each iteration in SMART sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>gca.SmartSweepShowIterations = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>gca</td>
<td>A GainCompression (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Boolean) Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong>  Compression traces are updated after each iteration.</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> Compression traces are updated after ALL iterations are complete.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
<tr>
<td>Examples</td>
<td><code>gca.SmartSweepShowIterations = True</code>  <code>Write</code></td>
</tr>
<tr>
<td></td>
<td><code>SShow = gca.SmartSweepShowIterations</code>  <code>Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_SmartSweepShowIterations(VARIANT_BOOL *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_SmartSweepShowIterations(VARIANT_BOOL newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IGainCompression</td>
</tr>
</tbody>
</table>
## SmartSweepTolerance Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the acceptable range SMART Sweep will allow for the measured compression level.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>$gca$.SmartSweepTolerance = value</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td></td>
<td>$gca$ - A GainCompression (object)</td>
</tr>
<tr>
<td></td>
<td>$value$ - (double) - Tolerance level in dB. Choose a value between .01 and 10</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>.05</td>
</tr>
<tr>
<td>Examples</td>
<td>$gca$.SmartSweepTolerance = .01 'Write</td>
</tr>
<tr>
<td></td>
<td>tol = gca.SmartSweepTolerance 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_SmartSweepTolerance(double* pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_SmartSweepTolerance(double newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IGainCompression</td>
</tr>
</tbody>
</table>


### SmoothingAperture Property

**Description**: Specifies or returns the amount of smoothing as a ratio of the number of data points in the measurement trace.

There is no COM command for specifying smoothing by number of aperture points.

**VB Syntax**: `meas.SmoothingAperture = value`

**Variable**

- **meas**: A Measurement (object)
- **value**: (double) - Smoothing Aperture. A ratio of (aperture points / trace points). Choose any number between .01 and .25.

**Return Type**: Double

**Default**: .25

**Examples**:  
- `meas.SmoothingAperture = .10`  
- `saperture = meas.SmoothingAperture`

**C++ Syntax**:  
- `HRESULT get_SmoothingAperture(double *pVal)`  
- `HRESULT put_SmoothingAperture(double newVal)`

**Interface**: IMeasurement
## Smoothing Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns ON and OFF data smoothing.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.Smoothing = state</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement <strong>(object)</strong></td>
</tr>
<tr>
<td>state</td>
<td><strong>(boolean)</strong></td>
</tr>
<tr>
<td>True</td>
<td><strong>-</strong> Turns smoothing ON</td>
</tr>
<tr>
<td>False</td>
<td><strong>-</strong> Turns smoothing OFF</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td><strong>False</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meas.Smoothing = False</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>smooth = meas.Smoothing</code> 'Read'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Smoothing(VARIANT_BOOL *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Smoothing(VARIANT_BOOL newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
SnPFormat Property

Description  Specifies the format of .SnP files.

Use either app.Save (saves data to file) or meas.Get SnpDataWithSpecifiedPorts (reads
data into variant array).

VB Syntax  

```
pref.SnPFormat = value
```  

Variable (Type) - Description

- **pref** (A Preferences **object**)  
- **value** (Format of the .S1P, .S2P, .S3P, .S4P data. Choose from:
  - "MA" - Linear Magnitude / degrees
  - "DB" - Log Mag / degrees
  - "RI" - Real / Imaginary
  - "Auto" - Format in which the trace is already displayed. If other than Log Mag, Linear Magnitude, or Real/Imag, then the format will be in Real/Imag.

Return Type  String

Default  "Auto"

Examples

```  
prefer.SnPFormat = "MA"  'Write

format = prefer.SnPFormat  'Read
```  

C++ Syntax  

```
HRESULT get_SnPFormat(BSTR *Format)

HRESULT put_SnPFormat(BSTR Format)
```  

Interface  IPreferences
SoftwareGateState Property

Description
When set to OFF, the improved software gating sensitivity is turned OFF and all data outside the measurement band is zeroed. This setting is used for troubleshooting purposes. There is NO user-interface control for this setting.

VB Syntax
pulseMeas.SoftwareGateState = bool

Variable
pulseMeas A PulseMeasurementControl (object)
bool True - Turn ON software gating.
False - Turn OFF software gating.

Return Type
Boolean

Default
True

Examples
pulse.SoftwareGateState = True 'Write
value = pulse.SoftwareGateState 'Read

C++ Syntax
HRESULT get_SoftwareGateState(VARIANT_BOOL *pVal);
HRESULT put_SoftwareGateState(VARIANT_BOOL newVal);

Interface
IPulseMeasurementControl2
### SoundOnFail Property

**Description**
Turns ON or OFF the audio indicator for limit failures.

**VB Syntax**
```vbnet
limitst.SoundOnFail = state
```

**Variable**
- **Type**: Description
  - `limitst` (object)
  - `state` (boolean)

  - **False**: Turns the sound OFF
  - **True**: Turns the sound ON

**Return Type**
Long Integer

**Default**
`True`

**Examples**
- `Limttest.SoundOnFail = False` 'Write
- `sound = Limttest.SoundOnFail` 'Read

**C++ Syntax**
```c++
HRESULT get_SoundOnFail(VARIANT_BOOL *pVal)
HRESULT put_SoundOnFail(VARIANT_BOOL newVal)
```

**Interface**
ILimitTest
About Trigger Source

Source Property

Sets or returns the source of triggering on the VNA and PNA-X.

**VB Syntax**

```
trigSetup.Source = value
```

**Variable**

* (Type) - Description

- **trigSetup** A TriggerSetup (object)
- **value** (enum NATriggerSource) - Choose from:
  
  0 - **naTriggerSourceInternal** - free run
  
  
  2 - **naTriggerSourceExternal** - a trigger signal is generated when a signal is sensed on the appropriate external trigger input connector. Use ExternalTriggerConnectionBehavior to configure the characteristics of the external trigger signal.

This setting has implications on Calibration. Learn more.

**Return Type**

Long Integer

**Default**

naTriggerSourceInternal

**Examples**

```
trigSetup.Source = naTriggerSourceInternal 'Write
trigsource = trigSetup.Source 'Read
```

**C++ Syntax**

```
HRESULT get_Source(tagNATriggerSource *pTrigger);
HRESULT put_Source(tagNATriggerSource trigger);
```

**Interface**

ITriggerSetup
### SourceCWFrequency Property

**Description**  
Set and read the source CW frequency.

**VB Syntax**  
\[
\text{sa.SourceCWFrequency (source)} = \text{value}
\]

**Variable**  
**Type**  
- \text{sa}  
  A SpectrumAnalyzer (object)

- \text{source}  
  (String) Source name enclosed in quotes. Use \text{SourcePortNames} to read a list of available source port names. See also \text{Remotely Specifying a Source Port}.

- \text{value}  
  (Double) CW frequency in Hz. Choose a value within the frequency range of the analyzer.

**Return Type**  
Double

**Default**  
Center frequency of the analyzer.

**Examples**  
\[
\begin{align*}
\text{sa.SourceCWFrequency("Port 1")} & = 1e9 \quad \text{Write} \\
\text{value} & = \text{sa.SourceCWFrequency("Port 1")} \quad \text{Read}
\end{align*}
\]

See an example program.

**C++ Syntax**  
```c++
HRESULT put_SourceCWFrequency(BSTR sourcename, double frequency);

HRESULT get_SourceCWFrequency(BSTR sourcename, double* frequency);
```

**Interface**  
\text{ISpectrumAnalyzer}
### Sources Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the names of the configured DC sources for the specified channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>$names = dc.Sources$</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td>$names$</td>
<td>(Variant) Variable to store the returned DC source names.</td>
</tr>
<tr>
<td>$dc$</td>
<td>An DCStimulus (object)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>$names = dc.Sources$ 'Read$</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Sources(VARIANT * pValue);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IDCStimulus</td>
</tr>
</tbody>
</table>
### SourceAttenuator (Cal All) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the Source Attenuator setting for a Cal All calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>calAll.SourceAttenuator (port) = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>calAll</code></td>
<td>A <code>CalibrateAllSourceAttenuator (object)</code></td>
</tr>
<tr>
<td><code>port</code></td>
<td>(Long) Source port number.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Attenuation value in dB for a Cal All calibration. Choose a valid value for the VNA model. See valid settings.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>0 (zero dB)</td>
</tr>
<tr>
<td>Examples</td>
<td><code>calAll.SourceAttenuator = 0 'Set value</code></td>
</tr>
<tr>
<td></td>
<td><code>value = calAll.SourceAttenuator 'Return value</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_SourceAttenuator (long port, double val);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_SourceAttenuator long port, double* newVal);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ICalibrateAll</td>
</tr>
</tbody>
</table>
**SourceAttenuator Property**

**Description**
Sets and returns the Source Attenuator setting for the Phase Reference calibration.

**Note:** This setting MUST match the source attenuator setting at the mixer input port for subsequent SMC+Phase measurements.

**VB Syntax**
```
phasRef.SourceAttenuator = value
```

**Variable**
- **phasRef**: A PhaseReferenceCalibration Object
- **value**: Attenuation value in dB. Choose a valid value for the VNA model. See valid settings.

**Return Type**
Double

**Default**
10 dB

**Examples**
```
phase.SourceAttenuator = 0
```

**See example program**

**C++ Syntax**
```
HRESULT get_SourceAttenuator(Double* pVals);

HRESULT put_SourceAttenuator(Double pVals);
```

**Interface**
IPhaseReferenceCalibration
## SourceCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of sources in the remote VNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = cap.SourceCount</code></td>
</tr>
<tr>
<td>Variable</td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) - Variable to store the returned number of sources.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = cap.SourceCount</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_SourceCount(long * sourceCount );</td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities</td>
</tr>
</tbody>
</table>
### SourceImpedance Property

**Description**: Sets and returns the source impedance of the pulse generator.

**VB Syntax**

```vbnet
extPulseGen.SourceImpedance = value
```

**Variable**

- `(Type)` - Description
- `extPulseGen` An `ExternalPulseGenerator` *(object)*
- `value` *(Double)* Pulse generator source impedance.

**Return Type**

- Double

**Default**

- 50

**Examples**

- `extPulseGen.SourceImpedance = 50` *'Write*
- `srcImp = extPulseGen.SourceImpedance` *'Read*

**C++ Syntax**

- `HRESULT get_SourceImpedance (double *pValue)`
- `HRESULT put_SourceImpedance (double newVal)`

**Interface**

- `IExternalPulseGenerator`
### Read-only

**SourcePort Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the source port of measurement. To understand how this property is useful, see IMeasurement2 Interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = meas.SourcePort</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long)</em> - Variable to store the returned value</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>sp = meas.SourcePort</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SourcePort( [out, retval] Long* srcPort);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
### SourcePortCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of ports that can output a signal. To learn more, see <a href="#">Remotely Specifying a Source Port</a>.</th>
</tr>
</thead>
</table>

#### VB Syntax

```vbnet
value = object.SourcePortCount
```

#### Variable (Type) - Description

- **value** (Long) - Variable to store the returned integer value of the number of source ports.
- **object** A Channel (object) - always more complete than capabilities object.

A Capabilities (object) - use when a channel is not available, or to find the common ports across all channels.

#### Return Type

Long

#### Default

Not Applicable

#### Examples

```vbnet
value = chan.SourcePortCount 'Read
```

#### C++ Syntax

```cpp
HRESULT get_SourcePortCount(long * count);
```

#### Interface

IChannel13

ICapabilities4
# SourcePointCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the number of steps the source will make across the specified source frequency range.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>sa.SourcePointCount = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>sa</code></td>
<td>A <code>SpectrumAnalyzer</code> <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long)</em> Point count. Choose a value between 1 and 2e9.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td></td>
</tr>
</tbody>
</table>

```
sa.SourcePointCount = 10  'Write
value = sa.SourcePointCount  'Read
```

See an example program.

<table>
<thead>
<tr>
<th><strong>C++ Syntax</strong></th>
<th>HRESULT put_SourcePointCount(long points);</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HRESULT get_SourcePointCount(long* points);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISpectrumAnalyzer</td>
</tr>
</tbody>
</table>
## SourcePortFixedFrequency Property

**Description**
Set and read the fixed frequency value for a specific port.

**VB Syntax**

```vbnet
chan.SourcePortFixedFrequency(port) = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chan</td>
<td>A SA Channel (object)</td>
</tr>
<tr>
<td>port</td>
<td>Source port number for which to set the fixed frequency value.</td>
</tr>
<tr>
<td>value</td>
<td>Fixed frequency value.</td>
</tr>
</tbody>
</table>

**Return Type**
double

**Default**
1

**Examples**

```vbnet
chan.SourcePortFixedFrequency(1) = 1e9  'Write
data = chan.SourcePortFixedFrequency(1)  'Read
```

**C++ Syntax**

```c++
HRESULT put_SourcePortFixedFrequency(long port, double value);
HRESULT get_SourcePortFixedFrequency(long port, double* value);
```

**Interface**

IChannel27
## SourcePortFrequencyOrder Property

**Description**
Set and read the source frequency domain’s order in the multi-dimensional sweep.

**VB Syntax**
```vbnet
md.SourcePortFrequencyOrder(port) = value
```

**Variable**
- **(Type)**: Description
  - `md` (MultiDimensionalSweep (object)) which belongs to a SA channel.
  - `port` (long) Source port number for which to set the source frequency order value.
  - `value` (long) Dimension order. Choose an integer value of 1 or higher.

**Return Type**
long

**Default**
1

**Examples**
```vbnet
md.SourcePortFrequencyOrder(0) = 2 'Write
value = md.SourcePortFrequencyOrder(1) 'Read
```

**C++ Syntax**
```cpp
HRESULT put_SourcePortFrequencyOrder(long port, long value);
HRESULT get_SourcePortFrequencyOrder(long port, long* value);
```

**Interface**
IMultiDimensionalSweep
### SourcePortFrequencyState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the source frequency domain’s ON/OFF state in the multi-dimensional sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( \text{md.SourcePortFrequencyState(port)} = \text{value} )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>( \text{md} )</td>
<td>A <code>MultiDimensionalSweep (object)</code> which belongs to a SA channel.</td>
</tr>
<tr>
<td>( \text{port} )</td>
<td>(long) Source port number for which to set the source frequency state.</td>
</tr>
<tr>
<td>( \text{value} )</td>
<td>Boolean) Choose from:</td>
</tr>
<tr>
<td>0</td>
<td><strong>OFF</strong> - Disable the specified source frequency domain in multi-dimensional sweep.</td>
</tr>
<tr>
<td>1</td>
<td><strong>ON</strong> - Enable the specified source frequency domain in multi-dimensional sweep.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>md.SourcePortFrequencyStateOrder(0) = OFF</code> ( \text{Write} ) <code>value = md.SourcePortFrequencyStateOrder(1)</code> ( \text{Read} )</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_SourcePortFrequencyState(long port, VARIANT_BOOL value);</code> <code>HRESULT get_SourcePortFrequencyState(long port, VARIANT_BOOL* value);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMultiDimensionalSweep</td>
</tr>
</tbody>
</table>
SourcePortMode Property

Description  Sets the state of the VNA source for the specified port.

VB Syntax  

```vbnet
chan.SourcePortMode(sourcePort) = value
```

Variable  

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chan</td>
<td>object</td>
<td>A Channel object</td>
</tr>
<tr>
<td>sourcePort</td>
<td>long integer</td>
<td>The source port for which to make this setting.</td>
</tr>
</tbody>
</table>

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source VNA-x model, then you must use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port.

value  (enum) - State of the source. Choose from:

0 - naSourcePortAuto  Port power is turned on when required for a measurement.

1 - naSourcePortOn  Port power is always ON, regardless of the measurement.

2 - naSourcePortOff  Port power is always OFF, regardless of the measurement.

3 - naSourcePortNOTUSE  Do not send OFF commands to the external sources. If an external source is in the OFF state, this option is used to stop sending OFF commands to the external source to increase sweep speed.

Return Type  Enum

Default  0 - naSourcePortAuto

Examples  

```vbnet
chan.SourcePortMode(1) = naSourcePortOn  'Write
state = chan.SourcePortMode(4)  'Read
```

C++ Syntax  

```c
HRESULT get_SourcePortMode(long sourcePort, enum NASourcePortMode*);
```

```c
HRESULT put_SourcePortMode(long sourcePort, enum NASourcePortMode);
```

Interface  IChannel9
SourcePortNames Property

Description

Returns the string names of ports that can output a signal.

The following is a list of string names for the PNA-X. Your VNA will NOT have all of these ports. Use GetPortNumber Method to return the correct port number for the specified port name.

- “Port 1”
- “Port 2”
- “Port 3”
- “Port 4”
- “Src2 Out1”
- “Src2 Out2”
- “Port 1 Src2”

For iTMSA (Opt S93460A/B)

- "Bal Port 1"
- "Bal Port 2"
- "SE Port1"
- "SE Port 2"

This command also lists the External Sources that are currently configured and selected.

To learn more, see Remotely Specifying a Source Port.

VB Syntax

value = object.SourcePortNames

Variable

value (Variant array) - Variable to store the returned list of source port names.

object A Channel (object) - always more complete than capabilities object.

A Capabilities (object) - use when a channel is not available, or to find the common ports across all channels.
<table>
<thead>
<tr>
<th>Return Type</th>
<th>Variant array of string names.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = chan.SourcePortNames</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_SourcePortNames(VARIANT *names);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IChannel13</td>
</tr>
<tr>
<td></td>
<td>ICapabilities4</td>
</tr>
</tbody>
</table>
### SourcePortPhaseOrder Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the source phase domain’s order in the multi-dimensional sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>md.SourcePortPhaseOrder(port) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>md</code></td>
<td>A <code>MultiDimensionalSweep</code> <em>(object)</em> which belongs to a SA channel.</td>
</tr>
<tr>
<td><code>port</code></td>
<td><em>(long)</em> Source port number for which to set the source phase order value.</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(long)</em> Dimension order. Choose an integer value of 1 or higher.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td><code>long</code></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>md.SourcePortPhaseOrder(0) = 2</code> <em>(Write)</em></td>
</tr>
<tr>
<td><code>value = md.SourcePortPhaseOrder(1)</code> <em>(Read)</em></td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_SourcePortPhaseOrder(long port, long value);</code></td>
</tr>
<tr>
<td><code>HRESULT get_SourcePortPhaseOrder(long port, long* value);</code></td>
<td></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IMultiDimensionalSweep</code></td>
</tr>
</tbody>
</table>
**SourcePortPhaseState Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the source phase domain’s ON/OFF state in the multi-dimensional sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>md.SourcePortPhaseState(port) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>md</code></td>
<td>A MultiDimensionalSweep (object) which belongs to a SA channel.</td>
</tr>
<tr>
<td><code>port</code></td>
<td>(long) Source port number for which to set the source phase state.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>Boolean) Choose from:</td>
</tr>
<tr>
<td>0</td>
<td><strong>OFF</strong> - Disable the specified source phase domain in multi-dimensional sweep.</td>
</tr>
<tr>
<td>1</td>
<td><strong>ON</strong> - Enable the specified source phase domain in multi-dimensional sweep.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>md.SourcePortPhaseState(0) = OFF</code> 'Write value = md.SourcePortPhaseState(1) 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_SourcePortPhaseState(long port, VARIANT_BOOL value);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_SourcePortPhaseState(long port, VARIANT_BOOL* value);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMultiDimensionalSweep</td>
</tr>
</tbody>
</table>
SourcePortPowerOrder Property

**Description**
Set and read the source power domain’s order in the multi-dimensional sweep.

**VB Syntax**

```vbnet
md.SourcePortPowerOrder(port) = value
```

**Variable (Type) - Description**

- **md** (MultiDimensionalSweep (object)) which belongs to a SA channel.
- **port** (long) Source port number for which to set the source power order value.
- **value** (long) Dimension order. Choose an integer value of 1 or higher.

**Return Type**
long

**Default**
1

**Examples**

```vbnet
md.SourcePortPowerOrder(0) = 2 'Write
value = md.SourcePortPowerOrder(1) 'Read
```

**C++ Syntax**

```cpp
HRESULT put_SourcePortPowerOrder(long port, long value);
HRESULT get_SourcePortPowerOrder(long port, long* value);
```

**Interface**
IMultiDimensionalSweep
# SourcePortPowerState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the source power domain’s ON/OFF state in the multi-dimensional sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>md.SourcePortPowerState(port) = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td></td>
<td><code>md</code> A MultiDimensionalSweep <em>(object)</em> which belongs to a SA channel.</td>
</tr>
<tr>
<td></td>
<td><code>port</code> <em>(long)</em> Source port number for which to set the source power state.</td>
</tr>
<tr>
<td></td>
<td><code>value</code> <em>(Boolean)</em> Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <strong>OFF</strong> - Disable the specified source power domain in multi-dimensional sweep.</td>
</tr>
<tr>
<td></td>
<td>1 - <strong>ON</strong> - Enable the specified source power domain in multi-dimensional sweep.</td>
</tr>
<tr>
<td>Return Type</td>
<td><em>(Boolean)</em></td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
<tr>
<td>Examples</td>
<td><code>md.SourcePortPowerState(0) = OFF</code> <em>Write</em></td>
</tr>
<tr>
<td></td>
<td><code>value = md.SourcePortPowerState(1)</code> <em>Read</em></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT put_SourcePortPowerState(long port, VARIANT_BOOL value);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_SourcePortPowerState(long port, VARIANT_BOOL* value);</td>
</tr>
<tr>
<td>Interface</td>
<td>IMultiDimensionalSweep</td>
</tr>
</tbody>
</table>
# SourcePortStartFrequency Property

Set and read the start frequency value for a specific port.

**VB Syntax**

```vbnet
chan.SourcePortStartFrequency(port) = value
```

**Variable (Type) - Description**

- `chan`: A SA Channel (object)
- `port`: Source port number for which to set the start frequency value.
- `value`: Start frequency value.

**Return Type**

double

**Default**

1

**Examples**

```vbnet
chan.SourcePortStartFrequency(1) = 1e9  'Write
value = chan.SourcePortStartFrequency(1)  'Read
```

**C++ Syntax**

```cpp
HRESULT put_SourcePortStartFrequency(long port, double value);
HRESULT get_SourcePortStartFrequency(long port, double* value);
```

**Interface**

IChannel27
### SourcePortStopFrequency Property

**Description**  
Set and read the stop frequency value for a specific port.

**VB Syntax**  
\[ \text{chan.SourcePortStopFrequency(port)} = \text{value} \]

**Variable (Type) - Description**
- **chan**: A SA Channel (object)
- **port**: (long) Source port number for which to set the stop frequency value.
- **value**: (double) Stop frequency value.

**Return Type**  
double

**Default**  
1

**Examples**

```vbnet
chan.SourcePortStopFrequency(1) = 1e9 'Write
value = chan.SourcePortStopFrequency(1) 'Read
```

**C++ Syntax**

```cpp
HRESULT put_SourcePortStopFrequency(long port, double value);
HRESULT get_SourcePortStopFrequency(long port, double* value);
```

**Interface**

IChannel27
SourcePower Property

Description: Set and read the source power level. This command applies to CW or linear sweep types.

VB Syntax: `sa.SourcePower(source) = value`

Variable (Type) - Description
- `sa` A SpectrumAnalyzer object
- `source` (String) Source name enclosed in quotes. Use `SourcePortNames` to read a list of available source port names. See also Remotely Specifying a Source Port.
- `value` (Double) Source power level in dBm. Choose a value within the power range of the source.

Return Type: Double

Default: Default of the source.

Examples:
- `sa.SourcePower("Port 1") = -5 'Write
- `value = sa.SourcePower("Port 1") 'Read

C++ Syntax:
- `HRESULT put_SourcePower(BSTR sourcename, double power);`
- `HRESULT get_SourcePower(BSTR sourcename, double* power);`

Interface: ISpectrumAnalyzer
## SourcePowerCalPowerOffset Property

**Description**
Sets or returns a power level offset from the VNA test port power. This can be a gain or loss value (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier at the input of your DUT.

Cal power is the sum of the test port power setting and this offset value. Following the calibration, the VNA power readouts are adjusted to the cal power.

This property performs the same function as the power offset argument on the `SetCalInfoEx` Method, except that this property can read the offset value.

**VB Syntax**

```
chan.SourcePowerCalPowerOffset(sourcePort) = value
```

**Variable**

- **chan** *(object)* - A Channel object
- **sourcePort** *(long integer)* - The source port for which to set this power offset value.
- **value** *(double)* - Gain or loss value in dB. Choose a value between -200 and 200.

**Return Type**
Double

**Default**
0 dB

**Examples**

```
chan.SourcePowerCalPowerOffset(1) = 10 'Write
offset = chan.SourcePowerCalPowerOffset(2) 'Read
```

**C++ Syntax**

```
HRESULT get_SourcePowerCalPowerOffset(long sourcePort, double *pVal);

HRESULT put_SourcePowerCalPowerOffset(long sourcePort, double newVal);
```

**Interface**
IChannel4
SourcePowerCorrection Property

Description
Sets source power correction ON or OFF for a specific source port on this channel, or returns the current ON or OFF state of correction for that source port.

VB Syntax
chan.SourcePowerCorrection(srcPort) = value

Variable (Type) - Description
chan (object) – A Channel object
srcPort (long integer) – Source port for which to set or return the ON or OFF state of source power correction.

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source VNA-x model, then you must use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port.

value (boolean)

False – Turns source power correction OFF for the source port.

True – Turns source power correction ON for the source port.

Return Type
Boolean

Default
False - Source power correction will turn correction ON

Examples
chan.SourcePowerCorrection(1) = False 'Write
calOnPort2 = chan.SourcePowerCorrection(2) 'Read

C++ Syntax
HRESULT put_SourcePowerCorrection(VARIANT_BOOL bState);
HRESULT get_SourcePowerCorrection(VARIANT_BOOL *bState);

Interface
IChannel
### SourcePowerPointCount Property

**Description**  Set and read the number of steps the source will make across the specified source power range. This setting is common to all sources.

**VB Syntax**  

```vbnet
sa.SourcePowerPointCount = value
```

**Variable**  

- **sa** (Type) - A SpectrumAnalyzer (object)
- **value** (Long) - Point count. Choose an integer value of 1 or higher.

**Return Type**  Long

**Default**  1

**Examples**  

```vbnet
sa.SourcePowerPointCount = 10 'Write
value = sa.SourcePowerPointCount 'Read
```

**C++ Syntax**  

```cpp
HRESULT put_SourcePowerPointCount(long points);
HRESULT get_SourcePowerPointCount(long* points);
```

**Interface**  ISpectrumAnalyzer
## SourcePowerRepeatCount Property

**Description**  
Set and read the number of SA (receiver) sweeps for each Source Step. This setting is common to all sources.

**VB Syntax**  
`sa.SourcePowerRepeatCount = value`

**Variable (Type) - Description**

- `sa`  
  A SpectrumAnalyzer (object)

- `value`  
  (Long) Repeat count. Choose an integer value of 1 or higher.

**Return Type**  
Long

**Default**  
1

**Examples**

```vbnet
sa.SourcePowerRepeatCount = 10 'Write
value = sa.SourcePowerRepeatCount 'Read
```

**C++ Syntax**

```c++
HRESULT put_SourcePowerRepeatCount(long count);
HRESULT get_SourcePowerRepeatCount(long* count);
```

**Interface**  
ISpectrumAnalyzer
SourcePowerOption Property

Description
Enables the source power to be set on individual sweep segments. This property must be set True before seg.TestPortPower = value is sent. Otherwise, the test port power command will be ignored.

VB Syntax
segs.SourcePowerOption = state

Variable (Type) - Description

segs A Segments collection (object)
state (boolean)

True - Enables variable TestPortPower to be set segment sweep
False - Disables variable TestPortPower to be set segment sweep

Return Type
Boolean

Default
False

Examples
segs.SourcePowerOption = True 'Write
powerOption = SourcePowerOption 'Read

C++ Syntax
HRESULT get_SourcePowerOption(VARIANT_BOOL *pVal)
HRESULT put_SourcePowerOption(VARIANT_BOOL newVal)

Interface ISegments
SourcePowerState Property

Description  Turns Source Power ON and OFF.

See note about source power state with instrument state save and recall.

VB Syntax  
```
app.SourcePowerState = state
```

Variable  
```
app
An Application (object)
```
```
state (boolean)
```

**False** - Turns Source Power OFF

**True** - Turns Source Power ON

Return Type  Boolean

Default  True

Examples  
```
app.SourcePowerState = True 'Write
```
```
pwr = app.SourcePowerState 'Read
```

C++ Syntax  
```
HRESULT get_SourcePowerState(VARIANT_BOOL *pVal)
HRESULT put_SourcePowerState(VARIANT_BOOL newVal)
```

Interface  IApplication
<table>
<thead>
<tr>
<th><strong>SourcePullForSParameters Property</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
</tr>
<tr>
<td><code>nfx</code></td>
</tr>
<tr>
<td><code>value</code></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
</tr>
<tr>
<td><strong>Default</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
</tr>
</tbody>
</table>
### SourceRepeatCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the number of SA (receiver) sweeps for each Source Step. This setting is common to all sources.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>sa.SourceRepeatCount = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>sa</code></td>
<td>A <code>SpectrumAnalyzer</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) Repeat count. Choose a value between 1 and 2e9.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>sa.SourceRepeatCount = 10 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = sa.SourceRepeatCount 'Read</code></td>
</tr>
</tbody>
</table>

See an example program.

<table>
<thead>
<tr>
<th><strong>C++ Syntax</strong></th>
<th>HRESULT put_SourceRepeatCount(long count);</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HRESULT get_SourceRepeatCount(long* count);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>ISpectrumAnalyzer</code></td>
</tr>
</tbody>
</table>
## SourceStartFrequency Property

**Description**  
Set and read the source start frequency.

**VB Syntax**  
\[ sa\.SourceStartFrequency\( source \) = value \]

**Variable (Type) - Description**

- \( sa \)  
  A SpectrumAnalyzer (object)

- \( source \)  
  (String) Source name enclosed in quotes. Use SourcePortNames to read a list of available source port names. See also Remotely Specifying a Source Port.

- \( value \)  
  (Double) Start frequency in Hz. Choose a value within the frequency range of the analyzer.

**Return Type**  
Double

**Default**  
Start frequency of the analyzer.

**Examples**

\[ sa\.SourceStartFrequency\("Port 1"\) = 1e9 \]  
'Write

\[ value = sa\.SourceStartFrequency\("Port 1"\) \]  
'Read

See an example program.

**C++ Syntax**

```
HRESULT put_SourceStartFrequency(BSTR sourcename, double frequency);
HRESULT get_SourceStartFrequency(BSTR sourcename, double* frequency);
```

**Interface**  
ISpectrumAnalyzer
SourceStartPower Property

Description: Set and read the source start power level. This command applies to Power or LF Power sweep types.

VB Syntax: `sa.SourceStartPower(source) = value`

Variable (Type) - Description

- `sa` A SpectrumAnalyzer (object)
- `source` (String) Source name enclosed in quotes. Use SourcePortNames to read a list of available source port names. See also Remotely Specifying a Source Port.
- `value` (Double) Start power level in dBm. Choose a value within the power range of the source.

Return Type: Double

Default: Default of the source.

Examples:

```vbnet
sa.SourceStartPower("Port 1") = -5 'Write
value = sa.SourceStartPower("Port 1") 'Read
```

C++ Syntax:

```c
HRESULT put_SourceStartPower(BSTR sourcename, double power);
HRESULT get_SourceStartPower(BSTR sourcename, double* power);
```

Interface: ISpectrumAnalyzer
SourceStepAttenuatorStepSize Property

Description
Returns a value indicating the step size of the source attenuator.

VB Syntax
value = cap.SourceStepAttenuatorStepSize (n)

Variable
(Type) - Description
- value (Double) - Variable to store the returned value of the attenuator step size.
- cap A Capabilities (object)
- n (Long) - port number to query for the value of the attenuator step size.

Return Type
Double

Default
Not Applicable

Examples
value = cap.SourceStepAttenuatorStepSize(1)

C++ Syntax
HRESULT get_SourceStepAttenuatorStepSize(long portNumber, double * stepSize);

Interface
ICapabilities5
### SourceStopFrequency Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the source stop frequency.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>sa.SourceStopFrequency(source) = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>sa</strong> A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td>source</td>
<td>(String) Source name enclosed in quotes. Use SourcePortNames to read a list of available source port names. See also Remotely Specifying a Source Port.</td>
</tr>
<tr>
<td>value</td>
<td>(Double) Stop frequency in Hz. Choose a value within the frequency range of the analyzer.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Stop frequency of the analyzer.</td>
</tr>
<tr>
<td>Examples</td>
<td><code>sa.SourceStopFrequency(&quot;Port 1&quot;) = 1e9   'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = sa.SourceStopFrequency(&quot;Port 1&quot;) 'Read</code></td>
</tr>
</tbody>
</table>

See an example program.

| C++ Syntax | HRESULT put_SourceStopFrequency(BSTR sourcename, double frequency); |
|            | HRESULT get_SourceStopFrequency(BSTR sourcename, double* frequency); |

Interface ISpectrumAnalyzer
**SourceStopPower Property**

**Description**  
Set and read the source stop power level. This command applies to Power or LFPower sweep types.

**VB Syntax**  
`sa.SourceStopPower(source) = value`

**Variable**  
`sa`  
A `SpectrumAnalyzer` object

`source`  
(String) Source name enclosed in quotes. Use `SourcePortNames` to read a list of available source port names. See also Remotely Specifying a Source Port.

`value`  
(Double) Stop power level in dBm. Choose a value within the power range of the source.

**Return Type**  
Double

**Default**  
Default of the source.

**Examples**  
`sa.SourceStopPower("Port 1") = 10`  
'Write

`value = sa.SourceStopPower("Port 1")`  
'Read

**C++ Syntax**  
`HRESULT put_SourceStopPower(BSTR sourcename, double power);`

`HRESULT get_SourceStopPower(BSTR sourcename, double* power);`

**Interface**  
ISpectrumAnalyzer
**SourceSweepFirstDimension Property**

**Description**
Set and read the sweep order. This command applies whenever frequency and power are being swept (sweep type set using the `SourceSweepType2` command). Otherwise, this setting is ignored. For example, if all the active sources are set to CW and/or linear sweep type, or if all the active sources are set to CW and/or power sweep type, the sweep order is ignored. If any active source is set to linear and power sweep type, or if an active source is set to linear sweep type and another active source is set to power sweep type, then the sweep order setting will be used.

**VB Syntax**

```vbnet
sa.SourceSweepFirstDimension (source) = value
```

**Variable (Type) - Description**

- `sa` A `SpectrumAnalyzer` (object)
- `value` (Enum as `NASASweepFirstTypes`) Choose from:
  - `0 - naSAFreqFirst` - Sweep from Start to Stop frequency first followed by a power sweep.
  - `1 - naSAPowerFirst` - Sweep power first then sweep from Start to Stop frequency.

**Return Type**
Enum

**Default**
`naSAFreqFirst`

**Examples**

```vbnet
sa.SourceSweepFirstDimension("Port 1") = naSAPowerFirst 'Write

value = sa.SourceSweepFirstDimension("Port 1") 'Read
```

**C++ Syntax**

```cpp
HRESULT put_SourceSweepFirstDimension(tagNASASweepFirstTypes first);

HRESULT get_SourceSweepFirstDimension(tagNASASweepFirstTypes* first);
```

**Interface**
`ISpectrumAnalyzer`
**SourceSweepType Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the source sweep type.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( sa.SourceSweepType \left( \text{source} \right) = \text{value} )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>( sa )</td>
<td>A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td>( \text{source} )</td>
<td>Source name enclosed in quotes. Use SourcePortNames to read a list of available source port names. See also Remotely Specifying a Source Port.</td>
</tr>
<tr>
<td>( \text{value} )</td>
<td>(Enum as NASweepType) Choose from:</td>
</tr>
<tr>
<td>0</td>
<td>naLinearSweep - SA source sweeps from Start to Stop in linear steps.</td>
</tr>
<tr>
<td>3</td>
<td>naCWTimeSweep - SA source is at a single frequency, set with SourceCWFrequency Property.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>3 - naCWTimeSweep</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>See an example program.</td>
</tr>
<tr>
<td>( sa.SourceSweepType\left(&quot;Port 1&quot;\right) = \text{naCWTimeSweep} \quad \text{'Write} )</td>
<td></td>
</tr>
<tr>
<td>( \text{value} = sa.SourceSweepType\left(&quot;Port 1&quot;\right) \quad \text{'Read} )</td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT put_SourceSweepType(BSTR sourcename, tagNASweepTypes sweeptype);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_SourceSweepType(BSTR sourcename, tagNASweepTypes* sweeptype);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISpectrumAnalyzer</td>
</tr>
</tbody>
</table>
# SourceSweepType2 Property

**Description**  
Set and read the source sweep type.

**VB Syntax**  
\( sa.\text{SourceSweepType2}\ (source) = value \)

**Variable (Type) - Description**

- \( sa\) - A SpectrumAnalyzer (object)
- \( source\) - (String) Source name enclosed in quotes. Use SourcePortNames to read a list of available source port names. See also Remotely Specifying a Source Port.
- \( value\) - (Enum as NASASourceSweepTypes) Choose from:
  
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>naSASourceFreqSweep - SA source sweeps from Start to Stop in linear steps.</td>
</tr>
<tr>
<td>1</td>
<td>naSASourceFreqAndPowerSweep - The source is set to sweep from the Start to Stop frequency and power sweep. The order is determined by the SourceSweepFirstDimension command.</td>
</tr>
<tr>
<td>2</td>
<td>naSASourcePowerSweep - SA source is set to a power sweep.</td>
</tr>
<tr>
<td>3</td>
<td>naSASourceCWFreqSweep - SA source is at a single frequency, set with SourceCWFrequency Property.</td>
</tr>
</tbody>
</table>

**Return Type**  
Enum

**Default**  
3 - naSASourceCWFreqSweep

**Examples**  

```vbnet
sa.SourceSweepType2("Port 1") = naSASourceCWFreqSweep 'Write

value = sa.SourceSweepType2("Port 1") 'Read
```

**C++ Syntax**  

```cpp
HRESULT put_SourceSweepType2(BSTR sourcename, tagNASASourceSweepTypes sweeptype);

HRESULT get_SourceSweepType2(BSTR sourcename, tagNASASourceSweepTypes* sweeptype);
```

**Interface**  
ISpectrumAnalyzer
## Span Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the Span time of either Gating or Time Domain transform windows</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>object.Span = value</code></td>
</tr>
</tbody>
</table>

### Variable

<table>
<thead>
<tr>
<th>(Type) - Description</th>
<th>(object) As Gating or (object) As Transform</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>value</strong></td>
<td>(double) - Span time in seconds. Choose any number between: <code>2*[(number of points-1) / frequency span]</code> and 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Double</th>
</tr>
</thead>
</table>

| Default | 20ns |

### Examples

- `Trans.Span = 4.5e-9` 'sets the time span of a transform window - Write
- `Gate.Span = 4.5e-9` 'sets the Span time of a gating window - Write
- `span = Trans.Span` 'Read

### C++ Syntax

- `HRESULT get_Span(double *pVal)`
- `HRESULT put_Span(double newVal)`

### Interface

- ITransform
- IGating
### Span Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the stimulus span of the measurement (stop-start data points). To understand how this property is useful, see IMeasurement2 Interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = meas.Span</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td>value</td>
<td><strong>(Double) - Variable to store the returned value.</strong></td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement <strong>(object)</strong></td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><strong>Print meas.Span</strong> 'prints the span of the measurement'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_Span(double * Val);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement2</td>
</tr>
<tr>
<td>Description</td>
<td>Read the current span DFT bin count, the number of DFT points processed across the total RF span. When the Detector is bypassed, this is the number of points that are sent to the display.</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>VB Syntax</td>
<td><code>value = sa.SpanBinsCount</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>sa</code></td>
<td>A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) Variable to store the returned value.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Example</td>
<td><code>value = sa.SpanBinsCount</code> Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_SpanBinsCount(long* val);</td>
</tr>
<tr>
<td>Interface</td>
<td>ISpectrumAnalyzer4</td>
</tr>
</tbody>
</table>
### SpanResolutionBWRatio Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the Frequency Span / RBW ratio.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>sa.SpanResolutionBWRatio = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>sa</code></td>
<td>A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Frequency Span / RBW ratio. Choose a value between 1 and 200e9.</td>
</tr>
</tbody>
</table>

*Learn about these settings.*

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Double</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>106</td>
</tr>
</tbody>
</table>

**Examples**

```vbnet
sa.SpanResolutionBWRatio = 100  'Write
value = sa.SpanResolutionBWRatio  'Read
```

*See an example program.*

<table>
<thead>
<tr>
<th><strong>C++ Syntax</strong></th>
<th>HRESULT put_SpanResolutionBWRatio(double ratio);</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HRESULT get_SpanResolutionBWRatio(double* ratio);</td>
</tr>
</tbody>
</table>

**Interface**

ISpectrumAnalyzer
SParameterCalPorts Property

Description
Read the final list of ports that will be fully calibrated by Calibrate All Channels. For the returned list of ports, specify connectors and cal kits for the calibration using the GuidedCal commands.

Ports to received ONLY a power cal (such as mixer LO ports) will NOT be on the returned list.

For each channel, specify the ports to be calibrated using CalibrationPorts Property

VB Syntax
ports = calAll.SParameterCalPorts

Variable (Type) - Description
ports (Variant Array) Ports to be calibrated.
calAll A CalibrateAllChannels (object)

Return Type Variant Array
Default Not Applicable

Examples
ports = calAll.SParameterCalPorts  'returns the ports to be cal'd

C++ Syntax
HRESULT SParameterCalPorts([out,retval] VARIANT* calports);

Interface ICalibrateAllChannels
## SpectrumCenterFrequency Property

Sets and returns the receiver Center frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when `Sweep Type` = Linear. Otherwise, this setting is ignored.

### Description

Sets and returns the receiver Center frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when `Sweep Type` = Linear. Otherwise, this setting is ignored.

### VB Syntax

```vbnet
ims.SpectrumCenterFrequency = value
```

### Variable

- **ims**: An IMSpectrum Object
- **value**: (Double) Center frequency in Hz. Choose a frequency within the range of the VNA.

### Return Type

Double

### Default

1.0 GHz

### Examples

```vbnet
ims.SpectrumCenterFrequency = 10e9 'Write
value = ims.SpectrumCenterFrequency 'Read
```

### C++ Syntax

```cpp
HRESULT get_SpectrumCenterFrequency(double *pVal)
HRESULT put_SpectrumCenterFrequency(double newVal)
```

### Interface

IIMSpectrum
### SpectrumSpanFrequency Property

**Description**
Sets and returns the Span of receiver frequencies for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when *Sweep Type* = Linear. Otherwise, this setting is ignored.

**VB Syntax**
```
ims.SpectrumSpanFrequency = value
```

**Variable**

- **ims**
  An IMSpectrum Object
- **value**
  (Double) Frequency span in Hz. Choose a frequency within the range of the VNA.

**Return Type**
Double

**Default**
100 MHz

**Examples**
```
ims.SpectrumSpanFrequency = 10e9 'Write
value = ims.SpectrumSpanFrequency 'Read
```

**C++ Syntax**
```
HRESULT get_SpectrumSpanFrequency(double *pVal)
HRESULT put_SpectrumSpanFrequency(double newVal)
```

**Interface**
IIMSpectrum
### SpectrumStartFrequency Property

**Description**
Sets and returns the receiver Start frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when Sweep Type = Linear. Otherwise, this setting is ignored.

**VB Syntax**
```vbnet
ims.SpectrumStartFrequency = value
```

**Variable (Type) - Description**
- `ims` An IMSpectrum Object
- `value` (Double) Start frequency in Hz. Choose a frequency within the range of the VNA.

**Return Type**
Double

**Default**
950 MHz

**Examples**
```vbnet
ims.SpectrumStartFrequency = 10e9 'Write
value = ims.SpectrumStartFrequency 'Read
```

**C++ Syntax**
```cpp
HRESULT get_SpectrumStartFrequency(double *pVal)
HRESULT put_SpectrumStartFrequency(double newVal)
```

**Interface**
IIMSpectrum
**SpectrumStopFrequency Property**

**Description**
Sets and returns the receiver Stop frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when Sweep Type = Linear. Otherwise, this setting is ignored.

**VB Syntax**

```vbnet
ims.SpectrumStopFrequency = value
```

**Variable (Type) - Description**

- **ims** An IMSpectrum Object
- **value** (Double) Stop frequency in Hz. Choose a frequency within the range of the VNA.

**Return Type**
Double

**Default**
950 MHz

**Examples**

```vbnet
ims.SpectrumStopFrequency = 10e9 'Write
value = ims.SpectrumStopFrequency 'Read
```

**C++ Syntax**

```cpp
HRESULT get_SpectrumStopFrequency(double *pVal)
HRESULT put_SpectrumStopFrequency(double newVal)
```

**Interface**
IIMSpectrum
SSB_BalPortNegative Property

<table>
<thead>
<tr>
<th>Description</th>
<th>With a Single-ended - Single-ended - Balanced topology, returns the VNA port number that is connected to the Negative side of the DUT's Balanced Port. Use SetSSBPorts Method to set the port mapping for a Single-Ended - Single-Ended - Balanced topology.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>var = balTopology.SSB_BalPortNegative</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>var</code></td>
<td>(Long Integer) Variable to store the returned value.</td>
</tr>
<tr>
<td><code>balTopology</code></td>
<td>A BalancedTopology (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long Integer</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>variable = balTopology.SSB_BalPortNegative</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_SSB_BalPortNegative(long *bVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IBalancedTopology</td>
</tr>
</tbody>
</table>
### SSB_BalPortPositive Property

**Description**  
With a Single-ended - Single-ended - Balanced topology, returns the VNA port number that is connected to the Positive side of the DUT's Balanced Port.

Use `SetSSBPorts Method` to set the port mapping for a Single-Ended - Single-Ended - Balanced topology.

**VB Syntax**  
```vbnet
var = balTopology.SSB_BalPortPositive
```

**Variable**  
`var` (Long Integer) Variable to store the returned value.

**balTopology**  
A `BalancedTopology (object)`

**Return Type**  
Long Integer

**Default**  
Not Applicable

**Examples**  
```vbnet
variable = balTopology.SSB_BalPortPositive 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_SSB_BalPortPositive(long *bVal)
```

**Interface**  
`IBalancedTopology`
### SSB_SEPort1 Property

**Description**  
With a Single-ended - Single-ended - Balanced topology, returns the VNA port number that is connected to the DUT's Logical Port 1.

Use SetSSBPorts Method to set the port mapping for a Single-Ended - Single-Ended - Balanced topology.

**VB Syntax**  
```vbnet
var = balTopology.SSB_SEPort1
```

**Variable**  
*var* (Long Integer) Variable to store the returned value.

**balTopology**  
A `BalancedTopology` object

**Return Type**  
Long Integer

**Default**  
Not Applicable

**Examples**  
```vbnet
variable = balTopology.SSB_SEPort1 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_SSB_SEPort1(long *bVal)
```

**Interface**  
IBalancedTopology
## SSB_SEPort2 Property

<table>
<thead>
<tr>
<th>Description</th>
<th>With a Single-ended - Single-ended - Balanced topology, returns the VNA port number that is connected to the DUT's Logical Port 2.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use</td>
<td>SetSSBPorts Method to set the port mapping for a Single-Ended - Single-Ended - Balanced topology.</td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>var = balTopology.SSB_SEPort2</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>var</code></td>
<td>(Long Integer) Variable to store the returned value.</td>
</tr>
<tr>
<td><strong>balTopology</strong></td>
<td>A BalancedTopology (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>variable = balTopology.SSB_SEPort2</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_SSB_SEPort2(long *bVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IBalancedTopology</td>
</tr>
</tbody>
</table>
SSBMeasurement Property

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>balMeas.SSBMeasurement = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td>balMeas</td>
<td>A <code>BalancedMeasurement</code> <em>(object)</em></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Sss11</th>
<th>Sss12</th>
<th>Ssd13</th>
<th>Ssc13</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sss21</td>
<td>Sss22</td>
<td>Ssd23</td>
<td>Ssc23</td>
</tr>
<tr>
<td></td>
<td>Sds31</td>
<td>Sds32</td>
<td>Sdd33</td>
<td>Sdc33</td>
</tr>
<tr>
<td></td>
<td>Scs31</td>
<td>Scs32</td>
<td>Scd33</td>
<td>Scc33</td>
</tr>
<tr>
<td>Imb1</td>
<td>Imb2</td>
<td>CMRR1</td>
<td>CMRR2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(Sds31/Scs31)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Sss11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
<th><code>balMeas.SSBMeasurement = &quot;Sss11&quot;</code> <em>(Write)</em></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>variable = balMeas.SSBMeasurement</code> <em>(Read)</em></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C++ Syntax</th>
<th>HRESULT get_SSBMeasurement(BSTR *pVal)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HRESULT put_SSBMeasurement(BSTR p newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IBalancedMeasurement</td>
</tr>
</tbody>
</table>
**Stage1Coefficients Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the digital filter coefficients of stage1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>spm4.Stage1Coefficients = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>spm4</code></td>
<td>A <code>SignalProcessingModuleFour</code> <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Variant Array)</em> Coefficients. An array of real values.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Variant</td>
</tr>
<tr>
<td>Default</td>
<td>Stage dependent.</td>
</tr>
<tr>
<td>Examples</td>
<td><code>spm4.Stage1Coefficients = 0,0.1,0.7,0.7,0.1</code> <em>(Write)</em></td>
</tr>
<tr>
<td></td>
<td><code>mode = spm4.Stage1Coefficients</code> <em>(Read)</em></td>
</tr>
</tbody>
</table>

**C++ Syntax**

```c++
HRESULT get_Stage1Coefficients(VARIANT* pCoefs);
HRESULT put_Stage1Coefficients(VARIANT pCoefs);
```

**Interface**

`ISignalProcessingModuleFour`
Write/Read

About PNA-X Pulsed Capabilities

Stage1Frequency Property

Description
Sets and returns the Numerically Controlled Oscillator (NCO) frequency of the Stage 1 filter. This command is only used when FilterMode Property is set to Manual.

VB Syntax

\[ spm4.Stage1Frequency = \text{value} \]

Variable

- **Type** - Description
  - \( spm4 \) A SignalProcessingModuleFour (object)
  - \( \text{value} \) (Double) Stage 1 Frequency. Min value = 0 Hz

Stage 1 Frequency. Min value = 0 Hz

With DSP 4 versions, Max value = 15 MHz.

With DSP 5 versions, Max value = 38 MHz.

Learn more about DSP versions.

Or programmatically use MinimumIFFrequency Property and MaximumIFFrequency Property to determine the range of settable values.

Return Type
Double

Default
Nominal IF Frequency. Learn more

Examples

\[ spm4.Stage1Frequency = 9E6 \quad \text{Write} \]
\[ \text{mode} = spm4.Stage1Frequency \quad \text{Read} \]

C++ Syntax

HRESULT get_Stage1Frequency(double *val);

HRESULT put_Stage1Frequency(double val);

Interface
ISignalProcessingModuleFour
## Stage1MaximumCoefficient Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the maximum value of stage 1 coefficients.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = spm4.Stage1MaximumCoefficient</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>value</code> <em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long)</em> Variable to store the returned Max coefficient.</td>
</tr>
<tr>
<td><code>spm4</code></td>
<td>A SignalProcessingModuleFour <em>(object)</em></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mode = spm4.Stage1MaximumCoefficient 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get Stage1MaximumCoefficient(long* val);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISignalProcessingModuleFour</td>
</tr>
</tbody>
</table>
### Stage1MaximumCoefficientCount Property

**Description**  
Returns the maximum number of coefficients for Stage1.

**VB Syntax**  
`value = spm4.Stage1MaximumCoefficientCount`

**Variable**  
(Type) - Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td>(Long) Variable to store the returned Max coefficient count.</td>
</tr>
</tbody>
</table>

**spm4**  
A `SignalProcessingModuleFour` (object)

**Default**  
Not Applicable

**Examples**  
`mode = spm4.Stage1MaximumCoefficientCount`  
`'Read`

**C++ Syntax**  
`HRESULT get_Stage1MaximumCoefficientCount(long* val);`

**Interface**  
`ISignalProcessingModuleFour`
### Stage1MaximumCoefficientSum Property

**Description**
Returns the maximum sum of all Stage1 coefficients.

**VB Syntax**
```
value = spm4.Stage1MaximumCoefficientSum
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(__int64* val) Variable to store the returned Max sum of all coefficients.</td>
</tr>
</tbody>
</table>

**spm4**
A SignalProcessingModuleFour (object)

**Default**
Not Applicable

**Examples**
```
mode = spm4.Stage1MaximumCoefficientSum 'Read
```

**C++ Syntax**
```
HRESULT get_Stage1MaximumCoefficientSum(__int64* val);
```

**Interface**
ISignalProcessingModuleFour
# Stage1MinimumCoefficientCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the minimum number of coefficients for Stage1</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = spm4.Stage1MinimumCoefficientCount</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td>value</td>
<td><em>(Long)</em> Variable to store the returned Min coefficient count.</td>
</tr>
<tr>
<td>spm4</td>
<td>A SignalProcessingModuleFour <em>(object)</em></td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mode = spm4.Stage1MinimumCoefficientCount 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_Stage1MinimumCoefficientCount(long* val);</td>
</tr>
<tr>
<td>Interface</td>
<td>ISignalProcessingModuleFour</td>
</tr>
</tbody>
</table>
## Stage2Coefficients Property

**Description**
Sets and returns Stage2Coefficients.

*Note:* Stage2 settings are ignored when using DSP Version 5. [Learn more.](#)

**VB Syntax**

```vbnet
spm4.Stage2Coefficients = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>spm4</code></td>
<td>A <code>SignalProcessingModuleFour</code> <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Variant)</em> An array of real numbers. Filter coefficients</td>
</tr>
</tbody>
</table>

**Return Type**

Variant

**Default**

Not Applicable

**Examples**

```vbnet
spm4.Stage2Coefficients = 'Write
mode = spm4.Stage2Coefficients 'Read
```

**C++ Syntax**

```cpp
HRESULT get_Stage2Coefficients(VARIANT* pCoefs);

HRESULT put_Stage2Coefficients(VARIANT pCoefs);
```

**Interface**

`ISignalProcessingModuleFour`
## Stage2MaximumCoefficient Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the maximum value of stage 2 coefficients.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = spm4.Stage2MaximumCoefficient</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(Long) Variable to store the returned Max coefficient.</td>
</tr>
<tr>
<td>spm4</td>
<td>A SignalProcessingModuleFour (object)</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mode = spm4.Stage2MaximumCoefficient 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_Stage2MaximumCoefficient(long* val);</td>
</tr>
<tr>
<td>Interface</td>
<td>ISignalProcessingModuleFour</td>
</tr>
</tbody>
</table>
**Stage2MaximumCoefficientCount Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the maximum number of coefficients for Stage2.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note:</td>
<td>Stage2 settings are ignored when using DSP Version 5. Learn more.</td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
type = spm4.Stage2MaximumCoefficientCount
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(Long) Variable to store the returned Max coefficient count.</td>
</tr>
<tr>
<td>spm4</td>
<td>A SignalProcessingModuleFour (object)</td>
</tr>
</tbody>
</table>

**Default**

Not Applicable

**Examples**

```vbnet
mode = spm4.Stage2MaximumCoefficientCount 'Read
```

**C++ Syntax**

```cpp
HRESULT get_Stage2MaximumCoefficientCount(long* val);
```

**Interface**

ISignalProcessingModuleFour
### Stage2MaximumCoefficientSum Property

**Description**

Returns the maximum sum of all Stage2 coefficients.

**Note:** Stage2 settings are ignored when using DSP Version 5. Learn more.

**VB Syntax**

```vbnet
value = spm4.Stage2MaximumCoefficientSum
```

**Variable**

- **value** (`__int64* val`) Variable to store the returned Max sum of all coefficients.

- **spm4** A `SignalProcessingModuleFour` (object)

**Default**

Not Applicable

**Examples**

```vbnet
mode = spm4.Stage2MaximumCoefficientSum 'Read
```

**C++ Syntax**

```c++
HRESULT get_Stage2MaximumCoefficientSum(__int64* val);
```

**Interface**

`ISignalProcessingModuleFour`
<table>
<thead>
<tr>
<th><strong>Stage2MinimumCoefficientCount Property</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
</tr>
<tr>
<td><strong>Note:</strong></td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
</tr>
<tr>
<td><code>value</code></td>
</tr>
<tr>
<td><code>spm4</code></td>
</tr>
<tr>
<td><strong>Default</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
</tr>
</tbody>
</table>
**About PNA-X Pulsed Capabilities**

---

### Stage3FilterType Property

**Description**
Sets and returns the Stage 3 filter type. This command is only used when **FilterMode** is set to Manual.

**VB Syntax**
```
spm4.Stage3FilterType = value
```

**Variable (Type) - Description**
- `spm4` [A SignalProcessingModuleFour](object)
- `value` (String) Filter type. Chose from:
  - "RECT" Rectangular Window Filter
  - "TUKEY" Tukey Filter
  - "PWIN" Pulse window filter

**Default**
TUKEY

**Examples**
```
spm4.Stage3FilterType = "PWIN"
mode = spm4.Stage3FilterType 'Read
```

**C++ Syntax**
```
HRESULT get_Stage3FilterType(BSTR* pFType);
HRESULT put_Stage3FilterType(BSTR GType);
```

**Interface**
ISignalProcessingModuleFour
## Stage3FilterTypes Property

**Description**
Returns a list of strings for the currently supported filter types that can be used for the stage 3 filter. This command is only used when FilterMode is set to False (Manual). See `Stage3FilterType` for a list of currently supported filter types.

**VB Syntax**
```
values = spm4.Stage3FilterTypes
```

**Variable**
- **Type**: Description
- **value** *(Variant)* Variable to store the returned filter types.
- **spm4** A `SignalProcessingModuleFour` *(object)*

**Return Type**
Variant Array

**Default**
Not Applicable

**Examples**
```
mode = spm4.Stage3FilterTypes 'Read
```

**C++ Syntax**
```
HRESULT get_Stage3FilterTypes(VARIANT* pTypes);
```

**Interface**
`ISignalProcessingModuleFour`
# Stage3Parameter Property

## Description
Sets and returns the Stage 3 filter parameters.

Must first select the filter type using `Stage3FilterType` before setting these parameters.

Use `Stage3Parameters` to return a list of the available parameters for the currently selected filter type.

## VB Syntax
```
spm4.Stage3Parameter(param) = value
```

## Variable *(Type) - Description*

- `spm4` (A SignalProcessingModuleFour (object))
- `param` (String) Filter parameter. Choose from:
  - "C" - Tap count (Tukey, RECT, PWIN)
  - "P" - Period (PWIN ONLY)
  - "D" - Delay (PWIN ONLY)
  - "W" - Width (PWIN ONLY)
  - "R" - Ramp Count (PWIN ONLY)
  - "M" - Number of times to repeat the user-supplied array for each data point (COEF ONLY)
- `value` (String) Parameter Value for the specified stage 3 parameter. Use `Stage3ParameterMaximum` and `Stage3ParameterMinimum` to return a range of values for the specified parameter.

## Default
- RECT: C = 1
- PWIN: C=1E6, P=10ms, D=50us, W=50us, R=7
- TUKEY: C=1

## Examples
```
spm4.Stage3Parameter("C") = 2
mode = spm4.Stage3Parameter("pwin") 'Read
```

## C++ Syntax
```
HRESULT get_Stage3Parameter(BSTR pName, double* pVal);
HRESULT put_Stage3Parameter(BSTR pName, double pVal);
```

## Interface
ISignalProcessingModuleFour
### Stage3ParameterMaximum Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns maximum parameter value for the current filter type.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = spm4.Stage3ParameterMaximum (parameter)</code></td>
</tr>
<tr>
<td>Variable (Type) - Description</td>
<td></td>
</tr>
<tr>
<td>value (Variant)</td>
<td>Variable to store the maximum parameter value.</td>
</tr>
<tr>
<td>spm4</td>
<td>A <code>SignalProcessingModuleFour</code> (object)</td>
</tr>
<tr>
<td>parameter (String)</td>
<td>Parameter name. See <code>Stage3Parameter Property</code> for a list of parameters.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mode = spm4.Stage3ParameterMaximum (&quot;c&quot;)</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_Stage3ParameterMaximum(BSTR pName, double* pVal);</code></td>
</tr>
<tr>
<td>Interface</td>
<td><code>ISignalProcessingModuleFour</code></td>
</tr>
</tbody>
</table>
Stage3ParameterMinimum Property

Description
Returns minimum parameter value for the current filter type.

VB Syntax
value = spm4.Stage3ParameterMinimum (parameter)

Variable (Type) - Description
value (Variant) Variable to store the minimum parameter value.

spm4 A SignalProcessingModuleFour (object)

parameter (String) Parameter name. See Stage3Parameter Property for a list of parameters.

Return Type Double

Default Not applicable

Examples
mode = spm4.Stage3ParameterMinimum ("c") 'Read

C++ Syntax
HRESULT get_Stage3ParameterMinimum(BSTR pName, double* pVal);

Interface ISignalProcessingModuleFour
### Stage3Parameters Property

**Description**
Returns the names of parameters for the current filter type. Use
Stage3FilterType Property to set the filter type.

**VB Syntax**

```vbnet
values = spm4.Stage3Parameters
```

**Variable**

- **value** *(Variant)* Variable to store the returned parameter names.
- **spm4** A SignalProcessingModuleFour *(object)*

**Return Type**
Variant

**Default**
Not Applicable

**Examples**

```vbnet
mode = spm4.Stage3Parameters
```

**C++ Syntax**

```c++
HRESULT get_Stage3Parameters(VARIANT* pNames);
```

**Interface**
ISignalProcessingModuleFour
## StandardDeviation Property

**Description**
Returns the standard deviation of the measurement.
To retrieve all 3 statistics value at the same time, use `meas.GetTraceStatistics`.

**VB Syntax**

```
stdev = meas.StandardDeviation
```

**Variable**

- `stdev` (single) - Variable to store standard deviation value
- `meas` - A Measurement (object)

**Return Type**
Single

**Default**
Not applicable

**Examples**

```
stdev = meas.StandardDeviation 'Read
```

**C++ Syntax**

```
HRESULT get_StandardDeviation(float* stdDeviation)
```

**Interface**
IMeasurement
StandardForClass Property - **Superseded**

**Description**  
Superseded  This command sets a single standard to a calibration class. Does NOT set or dictate the order for measuring the standards.

Use GetStandardForClass and SetStandardForClass. These commands allow up to seven standards to be assigned to a cal class.

**VB Syntax**  
`calKit.StandardForClass(class, portNum) = value`

**Variable**

<table>
<thead>
<tr>
<th>(Type) - Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>calKit</code></td>
<td>A CalKit (object). Use <code>calKit.GetCalStandard</code> to get a handle to the standard.</td>
</tr>
<tr>
<td><code>class</code></td>
<td>(enum NACalClass) Standard. Choose from:</td>
</tr>
<tr>
<td>1 - naClassA</td>
<td></td>
</tr>
<tr>
<td>2 - naClassB</td>
<td></td>
</tr>
<tr>
<td>3 - naClassC</td>
<td></td>
</tr>
<tr>
<td>4 - naClassD</td>
<td></td>
</tr>
<tr>
<td>5 - naClassE</td>
<td></td>
</tr>
<tr>
<td>6 - naReferenceRatioLine</td>
<td></td>
</tr>
<tr>
<td>7 - naReferenceRatioThru</td>
<td></td>
</tr>
</tbody>
</table>

**SOLT Standards**
1 - naSOLT_Open  
2 - naSOLT_Short  
3 - naSOLT_Load  
4 - naSOLT_Thru  
5 - naSOLT_Isolation

**TRL Standards**
1 - naTRL_Reflection  
2 - naTRL_Line_Reflection  
3 - naTRL_Line_Tracking  
4 - naTRL_Thru  
5 - naTRL_Isolation

**portNum** (long) - The port number the standard will be connected to. For example, you may have a 3.5mm connector designated for port 1, and Type N designated for port 2.
Parameter: **value** (long) - Calibration class number. Choose a number between 1 and 8. The <value> numbers are associated with the following calibration classes:

<table>
<thead>
<tr>
<th>&lt;value&gt;</th>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S11A</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>2</td>
<td>S11B</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>3</td>
<td>S11C</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>4</td>
<td>S21T</td>
<td>Thru standard</td>
</tr>
<tr>
<td>5</td>
<td>S22A</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>6</td>
<td>S22B</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>7</td>
<td>S22C</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>8</td>
<td>S21T</td>
<td>Thru standard</td>
</tr>
</tbody>
</table>

**Return Type**: Long Integer

**Default**: Not Applicable

**Examples**

```cpp
calKit.StandardForClass(naSOLT_Short, 1) = 1
Kclass = calKit.StandardForClass(naSOLT_Short, 1)
```

**C++ Syntax**

```cpp
HRESULT put_StandardForClass (NACalClass item, long pNum);

HRESULT get_StandardForClass (NACalClass* item, long *pNum);
```

**Interface**: ICalKit
### StandardReceiverNoiseBWList Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the list of supported Noise Bandwidths values when using the NA receiver for noise measurements (option 028). <a href="#">Learn more about Opt. 028.</a></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.StandardReceiverNoiseBWList</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>Type</strong> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>Variant containing one-dimensional array of long integers.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.StandardReceiverNoiseBWList</code> 'Read'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_StandardReceiverNoiseBWList (Variant *list);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ICapabilities12</td>
</tr>
</tbody>
</table>
# StartFrequency (Cal Set) Property

**Description**
Returns the start frequency that is stored in the Cal Set.

**VB Syntax**

```vbnet
value = CalSet.StartFrequency(range)
```

**Variable**
- **value** *(double)* - returned Start frequency in Hertz.
- **CalSet** *(object)*
- **range** *(Long)* Choose: 0 (Source and receiver frequency)

**Return Type**
Double

**Default**
Not Applicable

**Examples**

```vbnet
start = calset.StartFrequency(0) 'Reads the start frequency stored in the cal set.
```

**C++ Syntax**

```cpp
HRESULT get_StartFrequency(long range, *pVal)
```

**Interface**
|CalSet3|
StartFrequency Property

Description
Sets or returns the start frequency of the channel. (Channel Object)

Sets or returns the start frequency of the segment. (Segment Object)

Sets or returns the start frequency of the FOM Range. (FOMRange Object)

Sets or returns the start frequency of the Power Sensor coverage (GuidedCalibrationPowerSensor Object)

See also Measurement2 interface

VB Syntax
object.StartFrequency = value

Variable (Type) - Description
object
Any of the following:

Channel (object)

Segment (object)

FOMRange (object)

GuidedCalibrationPowerSensor (object)

value (double) - Start frequency in Hertz. Choose any number between the minimum and maximum frequencies of the analyzer. If start frequency is set greater than the stop frequency, then the stop frequency is set to the start frequency + frequency span.

Return Type
Double

Default
Channel - Minimum frequency of the analyzer

Segment - 0

FOMRange - Minimum frequency of the analyzer

PowerSensor - Minimum frequency of the analyzer

Examples
chan.StartFrequency = 4.5e9 'sets the start frequency of a linear sweep for the channel object
startfreq = Chan.StartFrequency 'Read
C++ Syntax

HRESULT get_StartFrequency(double *pVal)

HRESULT put_StartFrequency(double newVal)

Interface

IChannel

ISegment

IFOMRange

IGuidedCalibrationPowerSensor
Write/Read

About Power Sweep

StartPower Property

Description
Sets the start power of the analyzer when sweep type is set to Power Sweep. Frequency of the measurement is set with chan.CWFrequency.

VB Syntax
object.StartPower = value

Variable (Type) - Description
object One of the following:

- Channel (object)
- CalSet (object) - Read-only property

value (double) - Start Power in dBm.

Note: The range of settable power values depends on the VNA model and if source attenuators are installed. To determine the range of values, use cap.MaximumSourceALCPower and cap.MinimumSourceALCPower

Auto attenuation is not allowed in Power Sweep.

Return Type
Double

Default
0

Examples
Chan.StartPower = -10 'Write
strtpwr = Chan.StartPower 'Read

C++ Syntax
HRESULT get_StartPower(double *pVal)
HRESULT put_StartPower(double newVal)

Interface
IChannel

|CalSet3
### Start Property

**Description**  
Sets or returns the start time of either Gating or Time Domain transform windows

**VB Syntax**  
`object.Start = value`

**Variable**  
- `(Type) - Description`
  - `object`  
    - `(object) As Gating`
    - `or`
    - `(object) As Transform`
  - `value`  
    - `(double) - Start time in seconds. Choose any number between: ± (number of points-1) / frequency span`

**Return Type**  
Double

**Default**  
-10ns

**Examples**

- `Trans.Start = 4.5e-9 'sets the start time of a transform window -Write`
- `Gate.Start = 4.5e-9 'sets the start time of a gating window -Write`

- `strt = Trans.Start 'Read`

**C++ Syntax**

- `HRESULT get_Start(double *pVal)`
- `HRESULT put_Start(double newVal)`

**Interface**  
ITransform
IGating
Start Property

Description: Returns the stimulus value of the first data point for the measurement. To understand how this property is useful, see IMeasurement2 Interface.

VB Syntax: `value = meas.Start`

Variable (Type) - Description

value (Double) - Variable to store the returned value

meas A Measurement (object)

Return Type: Double

Default: Not Applicable

Examples: `Print meas.Start 'prints the stimulus value of the first data point`

C++ Syntax: `HRESULT get_Start (double * Val);`

Interface: IMeasurement2
StartFrequency (PowerCalRange) Property

**Description**
Sets and gets the start frequency for range <m> for source port<n>.

**VB Syntax**
```
PwrrCalRange .StartFrequency = freq
```

**Variable**
(Type) - Description
```
freq (Long) - Variable to store the returned start frequency.
```

**PwrrCalRange**
A PowerCalRange (object)

**Return Type**
Double

**Default**
Not Applicable

**Example**
```
CalibrateAllChannels.IndependentPowerCalibration(3).PowerCalRange(1).StartFrequency = 20e9 'Write

freq = CalibrateAllChannels.IndependentPowerCalibration(3).PowerCalRange(1).StartFrequency 'Read
```

**C++ Syntax**
```
HRESULT get_StartFrequency(double* freq);
HRESULT put_StartFrequency(double freq);
```

**Interface**
IPowerCalRange
## Start Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns start DC value for the specified DC source.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>$dc.Start(name, port) = value$</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>$dc$</td>
<td>An DCStimulus (object)</td>
</tr>
<tr>
<td>$name, port$</td>
<td>(String) Name of the &quot;DC source, port&quot;</td>
</tr>
</tbody>
</table>

Use **Source Property** to read a list of configured DC source names.

To set the DC source to be always ON, do NOT specify a port.

<table>
<thead>
<tr>
<th><strong>value</strong></th>
<th>(Double) DC Start value. Choose a value within the range of the DC source.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>.5</td>
</tr>
</tbody>
</table>

**Examples**

'**Set AO1 to always ON**

dc.Start "AO1", 3

'**Read Start for MyDCSource, Port 1**

dc.Start? "MyDCSource,Port 1"

**C++ Syntax**

HRESULT get_Start(BSTR name, VARIANT_BOOL * pValue);

HRESULT put_Start(BSTR name, VARIANT_BOOL newValue);

**Interface**

IDCStimulus
StartFrequency Property

Description
Sets and returns the phase reference cal start frequency.

VB Syntax
phasRef.StartFrequency = value

Variable (Type) - Description
phasRef A PhaseReferenceCalibration Object
value Start frequency in Hz. Choose any frequency from 17.5 MHz to the stop frequency of the VNA.

Return Type
Double

Default
17.5e6

Examples
phase.StartFrequency = 20e6

See example program

C++ Syntax
HRESULT get_StartFrequency(Double* pVals);
HRESULT put_StartFrequency(Double pVals);

Interface
IPhaseReferenceCalibration
### StartPhase Property

**Description**: Write and read the start value of phase sweep. Must also send Sweep Type Property to put the analyzer into phase sweep mode.

**VB Syntax**: `phase.StartPhase(srcPort) = value`

**Variable (Type) - Description**

- **phase**: A PhaseControl Object
- **srcPort**: (Long Integer) Source port for which to make phase control settings.

**Note**: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port

- **value**: (Double) Start phase value in degrees. Choose a value between -360 and 360.

**Return Type**: Double

**Default**: 0 degrees

**Examples**

```
phase.StartPhase 1 = 60 ' Write
value = phase.StartPhase 2' Read
```

**C++ Syntax**

```c++
HRESULT get_StartPhase(long port, double* pVal);

HRESULT put_StartPhase(long port, double newVal);
```

**Interface**: IPhaseControl
StartPowerEx Property

Description
Sets and reads the power sweep start power value for a specific port. This allows uncoupled forward and reverse power sweep ranges. Must also set SweepType = naPowerSweep, Coupled = False (Off), and StopPowerEx.

VB Syntax

`chan.StartPowerEx (srcPort) = value`

Variable (Type) - Description
- `chan` A Channel (object)
- `srcPort` (long integer) – Source port for which to set the Start power value.
- `value` (double) - Start Power in dBm.

**Note:** If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source VNA-x model, then you must use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port.

**Note:** The range of settable power values depends on the VNA model and if source attenuators are installed. To determine the range of values, use `cap.MaximumSourceALCPower` and `cap.MinimumSourceALCPower`.

Auto attenuation is not allowed in Power Sweep.

Return Type
Double

Default
-10 dBm

Examples

```
Chan.StartPowerEx 1 = -10 'Write
strtpwr = Chan.StartPowerEx 2 'Read
```

C++ Syntax

`HRESULT get_StartPowerEx(long port, double *pVal)`

`HRESULT put_StartPowerEx(long port, double newVal)`

Interface
IChannel13
### StartPowerIn3DSweep Property

**Description**  
Set and read the start power level for a 3D sweep.

**VB Syntax**  
```
HotS22.StartPowerIn3DSweep = level
```

**Variable**  
- **Type**: Description
- **HotS22** - A `ActiveParametersApp` object
- **level** - (Double) - Start power level in dBm. Choose a value from min power to max power of the hardware.

**Return Type**  
Double

**Default**  
-10 dBm

**Examples**  
```
HotS22.StartPowerIn3DSweep = 0 'Write
level = HotS22.StartPowerIn3DSweep 'Read
```

**C++ Syntax**  
```
HRESULT get_StartPowerIn3DSweep(double* level)
HRESULT put_StartPowerIn3DSweep(double level)
```

**Interface**  
IActiveChannelSettings
### StartRatioedPower Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Write and read the start power ratioed value. Must also set SweepType to Power.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>phase.StartRatioedPower(srcPort) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>phase</code></td>
<td>A PhaseControl Object</td>
</tr>
<tr>
<td><code>srcPort</code></td>
<td>(Long Integer) Source port for which to make phase control settings.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use <code>chan.getPortNumber</code> to translate the string into a port number. To learn more see <a href="#">Remotely Specifying a Source Port</a>.</td>
</tr>
<tr>
<td><strong>Value</strong></td>
<td>(Double) Start power ratio value in dBc. Must be within the allowable range of the VNA.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0 dBc</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>phase.StartRatioedPower 1 = -1 ' Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = phase.StartRatioedPower 2 ' Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>[ \text{HRESULT get_StartRatioedPower(long port, double* pVal);} ]</td>
</tr>
<tr>
<td></td>
<td>[ \text{HRESULT put_StartRatioedPower(long port, double newVal);} ]</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPhaseControl</td>
</tr>
</tbody>
</table>

---

**Note:** For a detailed explanation of the source port, see the linked documentation. The `chan.getPortNumber` function is used to convert a string source into a port number when using a balanced port, external source, or one of the Source 2 outputs on a PNA-X model.
## State (GPL) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables and disables Global Power Limiting for the specified port.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>gpl.State(port) = bool</code></td>
</tr>
<tr>
<td><strong>Variable</strong> (Type) - Description</td>
<td></td>
</tr>
<tr>
<td><code>gpl</code></td>
<td>A <code>GlobalPowerLimit</code> (object)</td>
</tr>
<tr>
<td><code>port</code> (Long)</td>
<td>Port number for which power limit state is to be set.</td>
</tr>
<tr>
<td><code>bool</code> (Boolean)</td>
<td>Choose from:</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>Power Limiting is enabled.</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>Power Limiting is disabled.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>gpl.State(1) = True</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>Limit = gpl.State(2)</code> 'Read'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_State(long port, VARIANT_BOOL *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Limit(long port, VARIANT_BOOL newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IGlobalPowerLimit</td>
</tr>
</tbody>
</table>
## State Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns an Object ON and OFF.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( \text{object.State} = \text{value} )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>\textit{object} (Type) - Description</td>
</tr>
<tr>
<td>Applies to any of the following objects:</td>
<td></td>
</tr>
<tr>
<td>FOM</td>
<td></td>
</tr>
<tr>
<td>Gating</td>
<td></td>
</tr>
<tr>
<td>InterfaceControl</td>
<td></td>
</tr>
<tr>
<td>LimitTest</td>
<td></td>
</tr>
<tr>
<td>Port Extension- Superseded. See either:</td>
<td></td>
</tr>
<tr>
<td>- \textbf{FixturingState Property}</td>
<td></td>
</tr>
<tr>
<td>- \textbf{PortExtState Property}</td>
<td></td>
</tr>
<tr>
<td>Segment</td>
<td></td>
</tr>
<tr>
<td>Transform</td>
<td></td>
</tr>
<tr>
<td>Equation</td>
<td></td>
</tr>
<tr>
<td><strong>FIFO</strong></td>
<td></td>
</tr>
</tbody>
</table>

### Notes:

- \textbf{LimitTest.State} - If using Global Pass/Fail status, trigger the VNA AFTER turning Limit testing ON.
- \textbf{Segment.State} - At least ONE segment must be ON or \textit{Sweep Type} is automatically set to \textit{Linear}.

<table>
<thead>
<tr>
<th>value</th>
<th>(boolean) -</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textbf{False}</td>
<td>Turns \textit{obj} OFF</td>
</tr>
<tr>
<td>\textbf{True}</td>
<td>Turns \textit{obj} ON</td>
</tr>
</tbody>
</table>

### Return Type

Boolean
Default  Depends on the object:

0 - FOM

0 - Gating

0 - InterfaceControl

0 - LimitTest

1 - Segment

0 - Transform

0 - Equation

0 - FIFO

Examples

Seg.State = 1 'Turns the segment object ON -Write

tran = Trans.State 'returns the state of Transform -Read

C++ Syntax

HRESULT get_State(VARIANT_BOOL *pVal)

HRESULT put_State(VARIANT_BOOL newVal)

Interface

ISegment

IInterfaceControl

ITransform

IGating

ILimitTest

IFOM

IEquation

IEmbeddedLO

IFIFO
# State Property

Sets and returns the ON / Off state of the specified DC source and port.

**VB Syntax**

```vbnet
dc.State(name, port) = state
```

**Variable (Type) - Description**

- **dc**
  - An DCStimulus (object)
- **name, port**
  - (String) Name of the "DC source, port"

Use **Source Property** to read a list of configured DC source names.

To set the DC source to be always ON, do NOT specify a port.

- **state**
  - Boolean. ON / Off state. Choose from:
    - True - DC source/port enabled.
    - False - DC source/port disabled.

**Return Type** Boolean

**Default** False

**Examples**

- "Set AO1 to always ON"
  ```vbnet
dc.State "AO1",True
  
  'Set MyDCSource to ON when the RF source for Port 1 is ON
  dc.State "MyDCSource,Port 1",ON
  
  'Read state for MyDCSource,Port 1
  dc.State? "MyDCSource,Port 1"
  ```

**C++ Syntax**

```c++
HRESULT get_State(BSTR name, VARIANT_BOOL * pValue);

HRESULT put_State(BSTR name, VARIANT_BOOL newValue);
```

**Interface** IDCStimulus
## About Receiver Leveling

### State (Rx Leveling) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the state of Receiver Leveling for a specific source port.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>RxLevel.State(srcPort) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>RxLevel</code></td>
<td>A ReceiverLeveling Object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean) Choose from:</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>Receiver leveling ON</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>Receiver leveling OFF</td>
</tr>
<tr>
<td><strong>srcPort</strong></td>
<td>(Long Integer) Source port for which to set the state of Receiver Leveling.</td>
</tr>
</tbody>
</table>

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Variant Boolean</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Examples</strong></th>
<th><code>rxLevel.State (1) = True</code> 'Write'</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value = rxLevel.State 2</code></td>
<td>'Read'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>C++ Syntax</strong></th>
<th><code>HRESULT get_State(long port, VARIANT_BOOL* pLevelingState);</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>HRESULT put_State(long port, VARIANT_BOOL LevelingState);</code></td>
<td></td>
</tr>
</tbody>
</table>

| **Interface** | IReceiverLevelingConfiguration |
### State Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns the specified pulse generator ON and OFF.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>pulse.State(n) = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>pulse</code></td>
<td>A PulseGenerator (object)</td>
</tr>
<tr>
<td><code>n</code></td>
<td>(Integer) Pulse generator number. Choose from 0 to 4.</td>
</tr>
<tr>
<td></td>
<td>Or use PulseGeneratorID to refer to an external pulse generator.</td>
</tr>
<tr>
<td></td>
<td>0 is the generator that pulses the ADC.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>True - turns pulse output ON.</td>
</tr>
<tr>
<td></td>
<td>False - turns pulse output OFF.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
<tr>
<td>Examples</td>
<td><code>pulse.State(1) = True</code>  <code>Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = pulse.State(4)</code>  <code>Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_State(integer pulse, VARIANT_BOOL *pVal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_State(integer pulse, VARIANT_BOOL newVal);</td>
</tr>
<tr>
<td>Interface</td>
<td>IPulseGenerator</td>
</tr>
</tbody>
</table>
### StatisticsRange Property

**Description**
Sets the User Range number for calculating measurement statistics. Set the start and stop values for a User Range with UserRangeMin and UserRangeMax.

There are 16 User Ranges per channel. User ranges are applied independently to any measurement.

**VB Syntax**

```vbnet
meas.StatisticsRange = value
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>value</td>
<td>(long integer) - Range Number. Choose any number between 0 and 16</td>
</tr>
</tbody>
</table>

- 0 is Full Span
- 1 - 16 are user-defined ranges

**Return Type**
Long Integer

**Default**
0

**Examples**

```vbnet
meas.StatisticsRange = 2 'Write
statrange = meas.StatisticsRange 'Read
```

**C++ Syntax**

```cpp
HRESULT get_StatisticsRange(long* rangeNumber)
HRESULT put_StatisticsRange(long rangeNumber)
```

**Interface**
IMeasurement
### StatusAsString Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a string that describes the result of the last tuning sweeps.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = embedLODiag.StatusAsString</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> Description</td>
</tr>
<tr>
<td>value</td>
<td><strong>(String)</strong> Variable to store the returned data.</td>
</tr>
<tr>
<td>embedLODiag</td>
<td>An EmbeddedLODiagnostic <strong>(object)</strong></td>
</tr>
<tr>
<td>Return Type</td>
<td><strong>(String)</strong></td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>data= embedLO.StatusAsString</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_StatusAsString(BSTR* status);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IEmbeddedLODiagnostic</td>
</tr>
</tbody>
</table>
## StepRiseTime Property

**Description**
Sets or returns the Rise time of the stimulus in Low Pass Step Mode.

**VB Syntax**
```vba
trans.StepRiseTime = value
```

**Variable (Type) - Description**
- `trans` (object) - A Transform
- `value` (double) - Rise time in seconds. Choose any number between 5.0e-13 and 1.63e-14.

**Return Type**
Double

**Default**
0

**Examples**
```vba
trans.StepRiseTime = 1.0e-14 'sets the step rise time to 100 psec. -Write
rt = trans.StepRiseTime 'Read
```

**C++ Syntax**
```cpp
HRESULT get_StepRiseTime(double *pVal)
HRESULT put_StepRiseTime(double newVal)
```

**Interface**
ITransform
### StepData Property

**Description**
Returns an array of data from the specified tuning sweep.

**VB Syntax**

```vbnet
value = embedLODiag.StepData(n)
```

**Variable**

- **Type** - Description
  - **value** (Variant Array) Variable to store the returned data.
  - **embedLODiag** An EmbeddedLODiagnostic (object)
  - **n** (Long) Tuning sweep number. Use NumberOfSweeps to find the number of sweeps taken.

**Default**
Not Applicable

**Examples**

```vbnet
data= embedLO.StepData 3 'read
```

**C++ Syntax**

```csharp
HRESULT StepData(long sweep, VARIANT* pArray);
```

**Interface**
IEmbededLODiagnostic
### StepTitle Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the title of the specified tuning sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>$value = embedLODiag.StepTitle(n)$</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td>$value$</td>
<td><em>(String)</em> Variable to store the returned data.</td>
</tr>
<tr>
<td><em>embedLODiag</em></td>
<td>An <em>EmbeddedLODiagnostic (object)</em></td>
</tr>
<tr>
<td>$n$</td>
<td><em>(Long)</em> Tuning sweep number. Use <em>NumberOfSweeps</em> to find the number of sweeps taken.</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>data= embedLO.StepTitle 3 'read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT StepTitle(long sweep, BSTR * title);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IEmbededLODiagnostic</td>
</tr>
</tbody>
</table>
### StimulusValues Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the specified X-axis FOM frequency range. The array contains one frequency value for each data point.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = calSet.StimulusValues (range)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>calSet</code></td>
<td>CalSet (object) - Read-only property</td>
</tr>
</tbody>
</table>
| `range` | (Long) FOM frequency range to read.  
  - 0 - returns source frequencies.  
  - 1 - returns response frequencies.  
  - 2 - returns primary frequencies. |
| **Return Type** | 1-dimensional variant array |
| **Default** | Not Applicable |
| **Examples** | `array = CalSet.StimulusValues 'Read` |
| **C++ Syntax** | `HRESULT get_StimulusValues (long range, VARIANT* vals)` |
| **Interface** | ICalSet3 |
### Stop Property

Sets and returns Stop DC value for the specified DC source.

#### VB Syntax

```vbnet
dc.Stop (name, port) = value
```

#### Variable - Description

- **dc**: An DCStimulus (object)  
- **name, port**: (String) Name of the "DC source, port"

Use Source Property to read a list of configured DC source names.

To set the DC source to be always ON, do NOT specify a port.

- **value**: (Double) DC Stop value. Choose a value within the range of the DC source.

#### Return Type

Double

#### Default

1

#### Examples

- **Set AO1 to always ON**
  ```vbnet
dc.Stop "AO1", 3
  ```

- **Read Stop for MyDCSource, Port 1**
  ```vbnet
dc.Stop? "MyDCSource, Port 1"
  ```

#### C++ Syntax

```cpp
HRESULT get_Stop(BSTR name, VARIANT_BOOL * pValue);

HRESULT put_Stop(BSTR name, VARIANT_BOOL newValue);
```

#### Interface

IDCStimulus

---

**2416**
**StopFrequency Property**

**Description**
Sets or returns the stop frequency of the channel. (Channel Object)

Sets or returns the stop frequency of the segment. (Segment Object)

Sets or returns the stop frequency of the FOM Range. (FOMRange Object)

Sets or returns the stop frequency of the Power Sensor coverage (GuidedCalibrationPowerSensor Object)

Sets or returns the stop frequency of the Phase Reference Calibration.

See also Measurement2 interface.

**VB Syntax**

```
object.StopFrequency = value
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>Any of the following:</td>
</tr>
<tr>
<td></td>
<td>Channel (object)</td>
</tr>
<tr>
<td></td>
<td>Segment (object)</td>
</tr>
<tr>
<td></td>
<td>FOMRange (object)</td>
</tr>
<tr>
<td></td>
<td>GuidedCalibrationPowerSensor (object)</td>
</tr>
<tr>
<td></td>
<td>PhaseReferenceCalibration (object)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>value</th>
<th>(double) - Stop frequency in Hertz. Choose any number between 70 (minimum) and maximum frequency limits of the analyzer. If the stop frequency is set less than the start frequency, then the start frequency is set to the stop frequency - frequency span.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
</tbody>
</table>
| Examples | `chan.StopFrequency = 4.5e9` *sets the stop frequency for the channel object* -Write
| stopfreq = Chan.StopFrequency *Read*
| C++ Syntax | HRESULT get_StopFrequency(double *pVal)
| | HRESULT put_StopFrequency(double newVal)
| Interface | IChannel
| | ISegment
| | IFOMRange
| | IGuidedCalibrationPowerSensor
StopFrequency (Cal Set) Property

Description
Returns the stop frequency that is stored in the Cal Set.

VB Syntax
value = CalSet.StopFrequency (range)

Variable (Type) - Description
value (double) - returned Stop frequency in Hertz.
CalSet (object)
range (Long) Choose: 0 (Source and receiver frequency)

Return Type
Double

Default
Not Applicable

Examples
stop = calset.StopFrequency(0) 'Reads the stop frequency stored in the cal set.

C++ Syntax
HRESULT get_StopFrequency(long range, double *pVal)

Interface |CalSet3
StopPower Property

Description
Sets the Stop Power of the analyzer when `sweep type` is set to Power Sweep. Frequency of the measurement is set with `chan.CWFrequency`.

VB Syntax

```vbnet
object.StopPower = value
```

Variable

(Type) - Description

`object` One of the following:

- `Channel` (object)
- `CalSet` (object) - Read-only property

`value` (double) - Stop Power in dB.

**Note:** The range of settable power values depends on the VNA model and if source attenuators are installed. To determine the range of values, use `cap.MaximumSourceALCPower` and `cap.MinimumSourceALCPower`.

Auto attenuation is not allowed in Power Sweep.

Return Type
Double

Default
0

Examples

```
Chan.StopPower = -10 'Write
stppwr = Chan.StopPower 'Read
```

C++ Syntax

```
HRESULT get_StopPower(double *pVal)
HRESULT put_StopPower(double newVal)
```

Interface

IChannel

|CalSet3
## Stop Property

**Description**
Sets or returns the Stop time of either Gating or Time Domain transform windows

**VB Syntax**
```
object.Stop = value
```

**Variable**
- **object** (Type) - Description
- **object** (object) As Gating
  or
- **object** (object) As Transform
- **value** (double) - Start time in seconds. Choose any number between:
  ± (number of points-1) / frequency span

**Return Type**
Double

**Default**
10 ns

**Examples**
- `Trans.Stop = 4.5e-9 'sets the stop time of a transform window - Write`
- `Gate.Stop = 4.5e-9 'sets the stop time of a gating window - Write`
- `stp = Trans.Stop 'Read`

**C++ Syntax**
```
HRESULT get_Stop(double *pVal)
HRESULT put_Stop(double newVal)
```

**Interface**
ITransform
IGating
## Stop Property

**Description**
Returns the stimulus value of the last data point for the measurement. To understand how this property is useful, see IMeasurement2 Interface.

**VB Syntax**

```vbnet
value = meas.Stop
```

**Variable (Type) - Description**

- `value` *(Double)* Variable to store the returned value
- `meas` A Measurement *(object)*

**Return Type**
Double

**Default**
Not Applicable

**Examples**

```vbnet
Print meas.Stop 'prints the stimulus value of the last data point
```

**C++ Syntax**

```cpp
HRESULT get_Stop(double * Val);
```

**Interface**
IMeasurement2
## StopFrequency (PowerCalRange) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and gets the stop frequency for range &lt;m&gt; for source port&lt;n&gt;.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>PwrrCalRange.StopFrequency = freq</code></td>
</tr>
<tr>
<td>Variable</td>
<td><code>freq (Long)</code> - Variable to store the returned stop frequency.</td>
</tr>
<tr>
<td>PwrrCalRange</td>
<td>A PowerCalRange (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Example</td>
<td><code>CalibrateAllChannels.IndependentPowerCalibration(3).PowerCalRange(1).StopFrequency = 21e9</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>freq = CalibrateAllChannels.IndependentPowerCalibration(3).PowerCalRange(1).StopFrequency</code> 'Read</td>
</tr>
</tbody>
</table>

### C++ Syntax

- `HRESULT get_StopFrequency(double* freq);`
- `HRESULT put_StopFrequency(double freq);`

### Interface

- `IPowerCalRange`
### StopPhase Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Write and read the stop value of phase sweep. Must also send <strong>Sweep Type Property</strong> to put the analyzer into phase sweep mode.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>phase.StopPhase(srcPort) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>phase</code></td>
<td>A <strong>PhaseControl</strong> Object</td>
</tr>
<tr>
<td><code>srcPort</code></td>
<td>(Long Integer) Source port for which to make phase control settings.</td>
</tr>
</tbody>
</table>

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see **Remotely Specifying a Source Port** |

| **Value** | (Double) Stop phase value in degrees. Choose a value between -360 and 360. |
| **Return Type** | Double |
| **Default** | 0 degrees |

**Examples**

```
phase.StopPhase 1 = 60 ' Write
value = phase.StopPhase 2 ' Read
```

**C++ Syntax**

```
HRESULT get_StopPhase(long port, double* pVal);
HRESULT put_StopPhase(long port, double newVal);
```

**Interface**

IPhaseControl
**StopPowerEx Property**

**Description**
Sets and reads the power sweep start power value for a specific port. This allows uncoupled forward and reverse power sweep ranges. Must also set `SweepType = naPowerSweep`, `Coupled = False (Off)`, and `StartPowerEx`.

**VB Syntax**

```vbnet
chan.StopPowerEx (srcPort) = value
```

**Variable (Type) - Description**

- `chan` A Channel (object)
- `srcPort` (long integer) – Source port for which to set the Stop power value.

**Note:** If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source VNA-x model, then you must use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

- `value` (double) - Stop Power in dBm.

**Note:** The range of settable power values depends on the VNA model and if source attenuators are installed. To determine the range of values, use `cap.MaximumSourceALCPower` and `cap.MinimumSourceALCPower`.

Auto attenuation is not allowed in Power Sweep.

**Return Type**
Double

**Default**
0 dBm

**Examples**

```csharp
Chan.StopPowerEx 1 = -10 'Write
stopPwr = Chan.StopPowerEx 2 'Read
```

**C++ Syntax**

```cpp
HRESULT get_StopPowerEx(long port, double *pVal)
HRESULT put_StopPowerEx(long port, double newVal)
```

**Interface**
IChannel13
**StopPowerIn3DSweep Property**

**Description**  Set and read the stop power level for a 3D sweep.

**VB Syntax**  
```
HotS22.StopPowerIn3DSweep = level
```

**Variable (Type) - Description**
- `HotS22`: A `ActiveParameterApp (object)`
- `level` (Double): Stop power level in dBm. Choose a value from min power to max power of the hardware.

**Return Type**  Double

**Default**  0 dBm

**Examples**  
- `HotS22.StopPowerIn3DSweep = 0`  'Write
- `level = HotS22.StopPowerIn3DSweep`  'Read

**C++ Syntax**  
- `HRESULT get_StopPowerIn3DSweep(double* level)`
- `HRESULT put_StopPowerIn3DSweep(double level)`

**Interface**  IActiveChannelSettings
## StopRatioedPower Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Write and read the stop power ratioed value. Must also set SweepType to Power.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( \text{phase.StopRatioedPower}(\text{srcPort}) = \text{value} )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>phase</td>
<td>A PhaseControl Object</td>
</tr>
<tr>
<td>srcPort</td>
<td>(Long Integer) Source port for which to make phase control settings.</td>
</tr>
</tbody>
</table>

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use \( \text{chan.getPortNumber} \) to translate the string into a port number. To learn more see Remotely Specifying a Source Port

| value | (Double) Stop power ratio value in dBc. Must be within the allowable range of the VNA. |
| Return Type | Double |
| Default | 0 dBc |

### Examples

- `\text{phase.StopRatioedPower 1 = -1 }` \textit{Write}

- `\text{value = phase.StopRatioedPower 2 }` \textit{Read}

### C++ Syntax

- `\text{HRESULT get\_StopRatioedPower(long port, double* pVal);}`

- `\text{HRESULT put\_StopRatioedPower(long port, double newVal);} `

### Interface

- IPhaseControl
strPort2Pdeembed_S2PFile Property

Description
Sets and returns the 2 port De-embedding .S2P file name for the specified port number. Model is applied when both the file name is specified and User is specified using Port2PdeembedCktModel Property.

Learn more about S2P files.

Note: This command affects ALL measurements on the channel.

VB Syntax
 fixture.strPort2Pdeembed_S2PFile(port) = value

Variable (Type) - Description
fixture A Fixturing (object)
port (Integer) Port number to receive circuit model.
value (String) Full path, file name, and extension (.s2P) of the de-embedding circuit.

Files are typically stored in "D:\".

Return Type String
Default Not Applicable

Examples
 fixture.strPort2Pdeembed_S2PFile(2) = "D:\myFile.s2p" 'Write
value = fixture.strPort2Pdeembed_S2PFile(1) 'Read

C++ Syntax
HRESULT get_strPort2Pdeembed_S2PFile(short port BSTR *bstrFile)
HRESULT put_strPort2Pdeembed_S2PFile(short port BSTR bstrFile)

Interface IFixturing
strPortMatch_S2PFile Property

Description
Sets and returns the Port Matching 'S2P' file name for the specified port number. Model is applied when both the file name is specified and User is specified using PortMatchingCktModel Property.

Learn more about S2P files.

Note: This command affects ALL measurements on the channel.

VB Syntax
```vbnet
fixture.strPortMatch_S2PFile(port) = value
```

Variable (Type) - Description
- `fixture` (A Fixturing object)
- `port` (Integer) Port number to receive circuit model.
- `value` (String) Full path, file name, and extension (.s2P) of the matching circuit.

Files are typically stored in "D:\".

Return Type String

Default Not Applicable

Examples
```vbnet
fixture.strPortMatch_S2PFile(2) = "D:\myFile.s2p" 'Write
value = fixture.strPortMatch_S2PFile(1) 'Read
```

C++ Syntax
```cpp
HRESULT get_strPortMatch_S2PFile(short port BSTR *bstrFile)
HRESULT put_strPortMatch_S2PFile(short port BSTR bstrFile)
```

Interface IFixturing
### SubPointTrigger Property

**Description**  Enables / Disables subpoint triggering. When enabled and performing Point Averaging, each rising edge of P0 triggers a subpoint (one of N acquisitions in an N point average). Must also enable the P0 generator using pulse.State(n).

**VB Syntax**  

```
pulse.SubPointTrigger(n) = value
```

**Variable**  

- **pulse**  A PulseGenerator (object)
- **n**  (Integer) Pulse generator number. **Must be 0** as this is the generator that triggers the ADC.
- **value**  (Boolean) Enable or disable SubPointTrigger

- **True** - turns subpoint triggering ON.
- **False** - turns subpoint triggering OFF.

**Return Type**  Boolean

**Default**  False

**Examples**  

```
pulse.SubPointTrigger(0) = True 'Write
bool = pulse.SubPointTrigger(0) 'Read
```

**C++ Syntax**  

```c++
HRESULT get_SubPointTrigger (integer pulse, VARIANT_BOOL* on_off);
HRESULT put_SubPointTrigger (integer pulse, VARIANT_BOOL on_off);
```

**Interface**  IPulseGenerator2
### SupportedParameters Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a list of supported parameters for the specified measurement class.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = measClassProps.ProperSupportedParameters</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Variant) - Variable to store the returned array of parameters.</td>
</tr>
<tr>
<td><code>measClassProps</code></td>
<td>A <em>MeasurementClassProperties</em> (Object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Variant array</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**Examples**

```vbscript
'Access the MeasurementClassProperties Object

Set app = CreateObject("AgilentPNA835x.Application")
Set cap = app.Capabilities
Set measProps = cap.MeasurementClassProperties("Swept IMD")

'Read the supported parameters

list=measProps.SupportedParameters

dim i
For i = 0 To UBound(list)
    msg = msg & list(i) & vbCrLf
Next
MsgBox msg
```

**C++ Syntax**

```c++
HRESULT get_SupportedParameters(Variant *value);
```

**Interface**

ICapabilities8
# Sweep Delay Property

**Description**
Specifies the time to wait just before acquisition begins for each sweep. This delay is in addition to Dwell Time and the following two External Trigger delays if enabled.

- TriggerDelay (global scope)
- ExternalTriggerDelay (channel scope)

**VB Syntax**

```vbnet
chan.SweepDelay = value
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>chan</th>
<th>Channel (object)</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(double) - Sweep delay in seconds.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
0

**Examples**

```vbnet
chan.SweepDelay = 3e-3 'Write
swpdelay = chan.SweepDelay 'Read
```

**C++ Syntax**

```cpp
HRESULT get_SweepDelay(double *pVal)
```

```cpp
HRESULT put_SweepDelay(double newVal)
```

**Interface**
IChannel
# SweepDwell Property

**Description**  Sets and returns the "Dwell Before Sweep" value for an external DC Device which can be configured as either a DC Meter or a DC Source.

**VB Syntax**  

\[ extDC.SweepDwell = value \]

**Variable**  

- **(Type)** - Description
- **extDC**  An ExternalDCDevice (object)
- **value**  (Double) The dwell time (in seconds) before making a new sweep.

**Return Type**  Double

**Default**  1 milliseconds

**Examples**  

- \[ extDC.SweepDwell = 10e-3 \] \( \text{Write} \)
- \[ dwell = extDC.SweepDwell \] \( \text{Read} \)

**C++ Syntax**  

- HRESULT get_SweepDwell (double *pValue)
- HRESULT put_SweepDwell (double newVal)

**Interface**  IExternalDCDevice
## Read/Write

### SweepEndMode Property

**Description**
Sets and reads the event that will cause the Sweep End line to go to a low state. The line will return to a high state after the appropriate calculations are complete.

*Note:* This line is connected to the HANDLER IO connector.

**VB Syntax**
```vbnet
object.SweepEndMode = value
```

**Variable (Type) - Description**
- `object` *(object)* - A HandlerIO or AuxIO object
- `value` *(enum as NASweepEndMode)* - Choose from:
  - 0 - `naSweep` - the line goes low when each sweep is complete
  - 1 - `naChannelSweep` - the line goes low when all the sweeps for each channel is complete.
  - 2 - `naGlobalSweep` - the line goes low when all sweeps for all triggerable channels are complete.

**Return Type**
Long Integer

**Default**
0 - `naSweep`

**Examples**
```
HWAuxIO.PassFailMode = naSweep 'Write
```
```
value = HWAuxIO.PassFailMode 'Read
```

**C++ Syntax**
```cpp
HRESULT put_SweepEndMode ( tagNASweepEndMode Mode );
```
```
HRESULT get_SweepEndMode ( tagNASweepEndMode* Mode );
```

**Interface**
IHWAuxIO

IHWMaterialHandlerIO
## SweepHoldOff Property

**Description**
Returns a boolean that represents the state of SweepHoldoff line (pin2) of the External Test Set connector.

**VB Syntax**
```
value = ExtIO.SweepHoldOff
```

**Variable (Type) - Description**
- **value** *(boolean)* - Variable to store the returned data
- **ExtIO** *(object)* - An External IO object

**Return Type**
boolean

- **False** - indicates the line is being held at a TTL Low
- **True** - indicates the line is being held at a TTL High

**Default**
Not Applicable

**Examples**
```
value = ExtIO.SweepHoldOff
```

**C++ Syntax**
```cpp
HRESULT get_SweepHoldOff( VARIANT_BOOL* bValue);
```

**Interface**
IHWEexternaTestSetIO
### SweepMode Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the sweep mode per segment. Enable the sweep mode property using the SweepModeOption Property on Segments collection.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>seg.SweepMode = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>seg</code></td>
<td>A Segment (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(NASweepGenerationModes) Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>0 - naSteppedSweep</strong> Sets sweep mode to stepped sweep.</td>
</tr>
<tr>
<td></td>
<td><strong>1 - naAnalogSweep</strong> Sets sweep mode to analog sweep.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>naAnalogSweep</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>seg.SweepMode = naAnalogSweep 'Write</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_SweepMode(tagNASweepGenerationModes* mode)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_SweepMode(tagNASweepGenerationModes mode)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISegment3</td>
</tr>
</tbody>
</table>
**SweepModeOption Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables or disables sweep mode per segment. Value is set using the SweepMode Property of a Segment.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>segs.SweepModeOption = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>segs</code></td>
<td>A Segments collection <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(boolean)</em></td>
</tr>
<tr>
<td>True</td>
<td>Enable sweep mode.</td>
</tr>
<tr>
<td>False</td>
<td>Disable sweep mode.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>segs.SweepModeOption = True</code> '<em>Write</em>'</td>
</tr>
<tr>
<td></td>
<td><code>sweepstate = segs.SweepModeOption</code> '<em>Read</em>'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_SweepModeOption(VARIANT_BOOL* pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_SweepModeOption(VARIANT_BOOL pVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISegments6</td>
</tr>
</tbody>
</table>
SweepOrder Property

Description  Sets and returns the order number of IM products to view when SweepType = NTH is specified. This actually sets the frequency span to DeltaF * N (this command value).

VB Syntax  

\[ ims.SweepOrder = value \]

Variable  (Type) - Description

- \( ims \) An IMSpectrum Object
- \( value \) (Integer) Order number of IM product.

Return Type  Long Integer

Default  9

Examples  

\[ ims.SweepOrder = 5 \ 'Write \]
\[ value = ims.SweepOrder \ 'Read \]

C++ Syntax  

HRESULT get_SweepOrder(long *pVal)
HRESULT put_SweepOrder(long newVal)

Interface  IIMSpectrum
**SweepSpeedMode Property**

**Description**  
Sets and returns the sweep speed mode: FastSweep or Normal.

**VB Syntax**  
`chan.SweepSpeedMode = value`

**Variable**  
*Type* - Description
- `chan` - A Channel (object)
- `value` - (enum as NASweepSpeedMode) - Choose from:
  - 0 - `naSweepSpeedModeNormal` - Standard VNA sweep mode
  - 1 - `naSweepSpeedModeFast` - Fast sweep mode

**Return Type**  
Enum

**Default**  
0 - `naSweepSpeedModeNormal`

**Examples**
- `chan.SweepSpeedMode = naSweepSpeedModeNormal`  
  `'Write`
- `swpSpeed = chan.SweepSpeedMode`  
  `'Read`

**C++ Syntax**
- `HRESULT get_SweepSpeedMode(tagNASweepSpeedModes* pVal)`
- `HRESULT put_SweepSpeedMode(tagNASweepSpeedModes newVal)`

**Interface**  
IChannel14
SweepGenerationMode Property

**Description**
Sets the method used to generate a sweep: continuous ramp (analog) or discrete steps (stepped).

**VB Syntax**

```
object.SweepGenerationMode = value
```

**Variable**

- **object**: Channel (object)

or

- **CalSet (object)**: Read-only property

**value**

- **(enum NASweepGenerationModes)**: Choose either:

  0 - **naSteppedSweep** - source frequency is CONSTANT during measurement of each displayed point. More accurate than Analog. Dwell time can be set in this mode.

  1 - **naAnalogSweep** - source frequency is continuously RAMPING during measurement of each displayed point. Faster than Stepped. Sweep time (not dwell time) can be set in this mode.

**Return Type**
Long Integer

**Default**
Analog

**Examples**

```
Chan.SweepGenerationMode = naAnalogSweep 'Write
swpgen = Chan.SweepGenerationMode 'Read
```

**C++ Syntax**

```
HRESULT get_SweepGenerationMode(tagNASweepGenerationModes* pVal)
HRESULT put_SweepGenerationMode(tagNASweepGenerationModes newVal)
```

**Interface**
IChannel

|CalSet3}
Write/Read

About Sweep Time

SweepTime Property

Description: Sets the Sweep time of the analyzer. If sweep time accuracy is critical, use ONLY the values that are attained using the up and down arrows next to the sweep time entry box.

See Sweep Time.

VB Syntax:

```vbnet
object.SweepTime = value
```

Variable (Type) - Description

- **object**: Channel (object)
- or
- **object**: Segment (object) first set SweepTimeOption to true.
- or
- **CalSet (object)** - Read-only property

- **value**: (double) - Sweep time in seconds. The maximum sweep time of the VNA is 86400 seconds (1 day).

To set the fastest sweep speed possible, set this value to 0.

Return Type: Double

Default: 0

Examples:

```vbnet
chan.SweepTime = 3e-3 'Write
swptme = chan.SweepTime 'Read
```

C++ Syntax:

```c++
HRESULT get_SweepTime(double *pVal)
HRESULT put_SweepTime(double newVal)
```

Interface:

- IChannel
- CalSet3
- ISegment2
### SweepTimeOption Property

**Description**
Enables the Sweep time or Dwell time to be set on individual sweep segments. This property must be set True **before** the sweep or dwell time commands are sent. Otherwise, those commands will be ignored.

**VB Syntax**
```
segs.SweepTimeOption = state
```

**Variable**
- **Type** - Description
  - `segs` A `Segments` collection *(object)*
  - `state` *(boolean)*

  **True** - Enables Sweep or Dwell time to be set independently.
  
  **False** - Disables Sweep or Dwell time from being set independently.

**Return Type**
Boolean

**Default**
False

**Examples**
```
segs.SweepTimeOption = True 'Write
timeOption = SweepTimeOption 'Read
```

**C++ Syntax**
```
HRESULT get_SweepTimeOption(VARIANT_BOOL *pVal)
HRESULT put_SweepTimeOption(VARIANT_BOOL newVal)
```

**Interface**
ISegments3
SweepType Property (IMD Opt S93087A/B)

Description
Sets and returns the type of sweep for a Swept IMD measurement.

See a list of commands that are relevant for each sweep type.

VB Syntax
\[ \text{imd.SweepType} = \text{value} \]

Variable (Type) - Description
\- \textit{imd} A SweptIMD Object
\- \textit{value} (Enum as naSweepTypes) - Choose from:

\- 0 - naIMDToneCWSweep The main tone frequencies (F1 and F2) and power levels (P1 and P2) are held constant. Measurements are taken for the specified number of points.

\- 1 - naIMDTonePowerSweep The main tone frequencies are specified as either F1 and F2, or as FC and DeltaF. These frequencies are held constant while the power of each tone is varied from the Start Power to Stop Power.

\- 2 - naIMDToneCenterFreqSweep Maintaining a constant tone spacing (DeltaF) and tone powers (P1 and P2), the center frequency (FC) is swept from Start to Stop, or can also be specified as Center and Span.

\- 3 - naIMDDeltaFrequencySweep The center frequency (FC) is held constant. The tone spacing is increased from Start DeltaF to Stop DeltaF.

\- 4 - naIMDToneSegmentSweep Same as FCenter sweep, except that the center frequencies for the sweep are constructed using the standard segment sweep commands. (NOT valid for IMDx)

\- 5 - naLOPowerSweep All frequencies are fixed while the LO power is swept. (IMDx ONLY)

Return Type
Enum

Default
2 - naIMDToneCenterFreqSweep

Examples
\texttt{imd.SweepType = naIMDToneCWSweep \ 'Write}
\texttt{swptyp = imd.SweepType \ 'Read}

C++ Syntax
HRESULT get_SweepType(tagNASweepTypes* pVal)
HRESULT put_SweepType(tagNASweepTypes newVal)

Interface
ISweptIMD
SweepType Property (IMSpectrum Opt S93087A/B)

**Description**
Sets and returns the type of sweep for an IMSpectrum measurement.

**VB Syntax**

```vbnet
ims.SweepType = value
```

**Variable (Type) - Description**

- **ims**
  An IMSpectrum Object
- **value**
  (Enum as NAIMSSweepType) - Choose from:

  - **0 - naIMSLinearSpan**
    When Tracking is enabled, allows tuning the Response Settings (receiver) to any values within the frequency range of the VNA. When Tracking is NOT enabled also allows setting the Stimulus (sources) to any values within the frequency range or the VNA.

  - **1 - naIMSSecondOrderSpan**
    The receiver is tuned to view the 2nd order products \((f_2 - f_1\) and \(f_1 + f_2\)) of the main tones that are currently specified in Stimulus Settings. When Tracking is enabled, the main tones are specified in the Swept IMD channel.

  - **2 - naIMSThirdOrderSpan**
    The receiver is tuned to view the 3rd order products \((2f_1 - f_2\) and \(2f_2 - f_1\)) of the main tones that are currently specified in Stimulus Settings. When Tracking is enabled, the main tones are specified in the Swept IMD channel.

  - **3 - naIMSNthOrderSpan**
    The frequency range is set to \(N \times \Delta F\). This algorithm will NOT tune the receivers to see the EVEN order products.

**Return Type**
Enum

**Default**
3 - naIMSNthOrderSpan

**Examples**

```vbnet
ims.SweepType = naIMSNthOrderSpan 'Write
swptyp = ims.SweepType 'Read
```

**C++ Syntax**

```cpp
HRESULT get_SweepType(tagNAIMSSweepType* pVal)
HRESULT put_SweepType(tagNAIMSSweepType newVal)
```

**Interface**
IMSpectrum
Write/Read

**About Sweep Types**

### SweepType Property

**Description**
Sets and returns the type of sweep. First set SweepType, then set sweep parameters such as frequency or power settings.

**VB Syntax**
```
object.SweepType = value
```

**Variable**
**(Type) - Description**
- **object** One of the following:
  - **Channel** (object)
  - **FOMRange** (object) Must be an **UNCOUPLED** range.
  - **CalSet** (object) - Read-only property

**value** *(enum NASweepTypes)* - Choose from:
- 0 - **naLinearSweep**
- 1 - **naLogSweep**
- 2 - **naPowerSweep**
- 3 - **naCWTimeSweep**
- 4 - **naSegmentSweep**
- 5 - **naPhaseSweep**

**Note**: Sweep type cannot be set to Segment sweep if there are no segments turned ON. A segment is automatically turned ON when a application is created.

**Return Type**
Long Integer

**Default**
**naLinearSweep**

**Examples**
```
chan.SweepType = naPowerSweep 'Write
swptyp = chan.SweepType 'Read
```

**C++ Syntax**
```
HRESULT get_SweepType(tagNASweepTypes* pVal)
HRESULT put_SweepType(tagNASweepTypes newVal)
```
Interface  IChannel
|CalSet3
IFOMRange
# SweepType Property

**Description**  
Set and read the sweep type.

**VB Syntax**  
```vbnet
HotS22.SweepType = value
```

**Variable**  
*Type* - Description

- **HotS22**  
  A `ActiveParametersApp (object)`

- **value**  
  (Enum as NAHOTSweepType) - Choose from:
  
  - **0 - naHOTS22LinearFrequencySweep**  
    Frequency linear sweep.
  
  - **1 - naHOTS22LogFrequencySweep**  
    Frequency log sweep.
  
  - **2 - naHOTS22PowerSweep**  
    Power sweep.
  
  - **3 - naHOTS22MultiSweep**  
    Multiple sweep. Sweeps frequency with fixed power and has several sweeps with different power values. This is called a 3D sweep: frequency, power, and phase.

**Return Type**  
Enum

**Default**  
0

**Examples**  
```vbnet
HotS22.SweepType = naHOTS22LinearFrequencySweep 'Write
value = HotS22.SweepType 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_SweepType(tagNAHOTSweepType* value)
HRESULT put_SweepType(tagNAHOTSweepType value)
```

**Interface**  
IActiveChannelSettings
# SystemImpedanceZ0 Property

Sets and returns the impedance for the analyzer.

**VB Syntax**

```vbnet
app.SystemImpedanceZ0 = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) Analyzer Impedance. Choose any number between 0 and 1000 ohms.</td>
</tr>
</tbody>
</table>

**Return Type**

Double

**Default**

50

**Examples**

```vbnet
app.SystemImpedanceZ0 = 75  'Write
z0 = app.SystemImpedanceZ0  'Read
```

**C++ Syntax**

```cpp
HRESULT get_SystemImpedanceZ0(double dSystemZ0)
HRESULT put_SystemImpedanceZ0(double *pdSystemZ0)
```

**Interface**

IApplication
<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the computer name of the VNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>name = app.SystemName</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>name</code></td>
<td>(String) Variable to store the returned computer name.</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>name = app.SystemName</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT SystemName(BSTR* computerName)</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IApplication</td>
</tr>
</tbody>
</table>
**TargetValue Property**

**Description**
Sets the target value for the marker when doing Target Searches (SearchTargetLeft, SearchTarget, SearchTargetRight).

**VB Syntax**
```
mark.TargetValue = value
```

**Variable (Type) - Description**

- **mark** A Marker (object)
- **value** (single) - Target value. Choose any number between: -500 and 500

**Return Type**
Single

**Default**
0

**Examples**
```
mark.TargetValue = 10.5 'Write

target = mark.TargetValue 'Read
```

**C++ Syntax**
```
HRESULT get_TargetValue(float *pVal)
HRESULT put_TargetValue(float newVal)
```

**Interface**
IMarker
# TestPortNames Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a list of test port names including external test set ports.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.TestPortNames</code></td>
</tr>
<tr>
<td><strong>Variable</strong> (Type) - Description</td>
<td></td>
</tr>
<tr>
<td><code>value</code></td>
<td>Variant containing one-dimensional array of string names.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A Capabilities (object)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.TestPortNames</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_TestPortNames(Variant *names);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities13</td>
</tr>
</tbody>
</table>
### TestPortPower Property

**Description**
Sets or returns the RF power level for the channel or
Sets or returns the RF power level of the segment.

**VB Syntax**
```
object.TestPortPower(srcPort) = value
```

**Variable**
- **object**
  - A Channel (object) - to set coupled power, use chan.CouplePorts. If CouplePorts = False, then each port power can be set independently. Otherwise, chanTestPortPower (1) = value sets power level at both ports.
  - or
  - A CalSet (object)
  - or
  - A Segment (object)

- **srcPort**
  - (long integer) - Source Port number.

- **value**
  - (double) - RF Power in dBm.

**Note:** If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source VNA-x model, then you must use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port.

**Note:** The range of settable power values depends on the VNA model and if source attenuators are installed. To determine the range of values, use cap.MaximumSourceALCPower and cap.MinimumSourceALCPower

Actual achievable leveled power depends on frequency.

**Return Type**
Double

**Default**
0

**Examples**
```
chan.TestPortPower(1) = 5 'sets the port 1 RF power level for the channel object -Write
```
```
powerlev = Chan.TestPortPower(1) 'Read
```

**C++ Syntax**
```
HRESULT get_TestPortPower(long port, double *pVal)
HRESULT put_TestPortPower(long port, double newVal)
```
<table>
<thead>
<tr>
<th>Interface</th>
<th>IChannel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ICalSet3</td>
</tr>
<tr>
<td></td>
<td>ISegment</td>
</tr>
</tbody>
</table>
## TestSetType Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Test Set Type (model) that was used for the Cal Set.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>TSType = calset.TestSetType</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>TSType</code></td>
<td>(String) Variable to store the returned test set model.</td>
</tr>
<tr>
<td><code>calset</code></td>
<td>A Cal Set object.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Depends on the test set.</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>TSType = calset.TestSetType</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_OutputPorts(BSTR *mapping);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalset5</td>
</tr>
</tbody>
</table>
# Text Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies an equation or expression to be used on the measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>eq.Text = eqText</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>eq</code></td>
<td>MeasurementEquation <em>(object)</em></td>
</tr>
<tr>
<td><code>eqText</code></td>
<td><em>(String)</em> - Any valid equation or expression.</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>eq.Text = &quot;foo=S11/S21&quot;</code></td>
</tr>
<tr>
<td></td>
<td><code>equation = eq.Text</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_Text(BSTR *equation)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Text(BSTR equation)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurementEquation</td>
</tr>
</tbody>
</table>
## TextFileEnabled Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables/disables text file (*.txt) output.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>sa.dft.TextFileEnabled = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>sa.dft</code></td>
<td>A <code>SpectrumAnalyzerDFT</code> <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Boolean)</em> Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>0 - OFF</strong> - Disable text file output.</td>
</tr>
<tr>
<td></td>
<td><strong>1 - ON</strong> - Enable text file output.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>OFF</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>sa.dft.TextFileEnabled = ON 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = sa.dft.TextFileEnabled 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_TextFileEnabled(VARIANT_BOOL* enable)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_TextFileEnabled(VARIANT_BOOL enable)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISpectrumAnalyzerDFT</td>
</tr>
</tbody>
</table>
**ThruCalMethod Property - Superseded**

**Description**
This command is replaced by PathThruMethod Property.

Sets and returns the method for performing the thru portion of the calibration.

**VB Syntax**

```vbnet
obj.ThruCalMethod = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
</tr>
</tbody>
</table>

*SMCType (object)*

or

*VMCType (object)*

**value**

(String) Specifies the Thru method. Case insensitive - include spaces. Choose from:

"Default"

"Flush Thru" or "FLUSH"

"Unknown Thru" or "UNKN"

"Adapter Removal" or "ADAP"

**Return Type**

String

**Default**

Default

**Examples**

```vbnet
SMC.ThruCalMethod = "UNKN"
```

**C++ Syntax**

```csharp
HRESULT put_ThruCalMethod(enum NAThruCalMethod thruMethod);

HRESULT get_ThruCalMethod(enum NAThruCalMethod *thruMethod);
```

**Interface**

SMCType

VMCType
## ThruCalMethod Property  Superseded

<table>
<thead>
<tr>
<th>Description</th>
<th>This command is replaced by PathThruMethod Property.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sets and returns the method for performing the Cal Method and the THRU portion of the calibration.</td>
</tr>
</tbody>
</table>

### VB Syntax

```vbnet
guidedCal ThruCalMethod = value
```

### Variable (Type) - Description

- **guidedCal**: GuidedCalibration (object)
- **value**: (Enum as NATruCalMethod) Choose from:

  0 - **naDefaultCalMethod** - allow the VNA to choose the best possible method (from the following) depending on whether the device or ECal module is insertable or non-insertable and given the model number of the VNA. (default selection if omitted.)
  1 - **naAdapterRemoval** - Perform Adapter removal calibration.
  2 - **naFlushThru** - Perform Flush Thru calibration.
  3 - **naDefinedThru** - Perform Defined Thru calibration. If performing an ECal, this is the Thru standard in the ECal module.
  4 - **naUnknownThru** - Perform Unknown Thru calibration.
  5 - **naSOLT** - Perform SOLT calibration
  6 - **naTRL** - Perform TRL calibration
  7 - **naQSOLT** - Perform QSOLT calibration.

### Return Type

Enum

### Default

0 - naDefaultCalMethod

### Examples

```vbnet
guided.ThruCalMethod = naDefinedThru
```

### C++ Syntax

```cpp
HRESULT get_ThruCalMethod(enum NAThruCalMethod *thruMethod);
HRESULT put_ThruCalMethod(enum NAThruCalMethod thruMethod);
```

### Interface

IGuidedCalibration
### ThruPortList Property

**Description**

**Note:** Do NOT send this command to rely on SmartCal to determine the most accurate Thru port pairs for the cal. You can send the query form of this command to learn the port pairs determined by SmartCal.

Sets and returns the Thru connection port pairs for the calibration. Send the query form of this command to learn the Thru pairs determined by SmartCal.

See **Thru Pairs Sequence** to learn how to send this and other Thru commands.

Learn more about Thru method and port pairings.

See an example of a 4-port guided calibration using COM.

### VB Syntax

```vb
guidedCal.ThruPortList = t1a, t1b, t2a, t2b, t3a, t3b
```

### Variable (Type) - Description

- **guidedCal** GuidedCalibration (object)
- **t1a, t1b...** (Variant) Port numbers in pairs - a one-dimensional array of Long integers.
  - t1a, t1b (Thru1 - port A and port B)
  - t2a, t2b (Thru2 - port A and port B)
  - t3a, t3b (Thru3 - port A and port B)

### Return Type

**Variant** - a one-dimensional array of Long integers.

### Default

The most accurate port pairs for the cal.

### Example

```vb
thruList = Array(1,2,1,3,1,4)
guided.ThruPortList = thruList
```

Sets the following three thru connections for a 4-port calibration:

- Thru 1 - ports 1 and 2
- Thru 2 - ports 1 and 3
- Thru 3 - ports 1 and 4

### C++ Syntax

```cpp
HRESULT get_ThruPortList(VARIANT* portList);
HRESULT put_ThruPortList(VARIANT portList);
```

### Interface

IGuidedCalibration
<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the Time out value for communication with the external device. An error is returned if communication with the device is not successful within this period of time.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>extDevices.TimeOut = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td>extDevices</td>
<td>An <code>ExternalDevice (object)</code></td>
</tr>
<tr>
<td>value</td>
<td>(Double) Time out in milliseconds.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>20000 milliseconds (20 seconds)</td>
</tr>
<tr>
<td>Examples</td>
<td><code>extDevices.TimeOut = 1000 Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = extDevices.TimeOut 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_TimeOut(Double* value);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_TimeOut(Double newVal);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IExternalDevices</td>
</tr>
</tbody>
</table>
## Title Property

**Description**
Writes or reads a custom title for the window. Newer entries replace (not append) older entries. Turn the title ON and OFF with TitleState.

**VB Syntax**
```
win.Title = string
```

**Variable (Type) - Description**
- `win` (A NaWindow (object))
- `string` (long) - Title limited to 50 characters.

**Return Type**
String

**Default**
Null

**Examples**
```
win.Title = "Hello World" 'Write

titl = win.Title 'Read
```

**C++ Syntax**
- HRESULT get_Title(BSTR *title)
- HRESULT put_Title(BSTR title)

**Interface**
INAWindow
## Title (Channel) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Writes or reads a custom title for the channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan.Title = string</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>A Channel <strong>(object)</strong></td>
</tr>
<tr>
<td><code>string</code></td>
<td>(long) - Title limited to 50 characters.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Null</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chan.Title = &quot;My Channel Title&quot;</code> <strong>Write</strong></td>
</tr>
<tr>
<td></td>
<td><code>titl = chan.Title</code> <strong>Read</strong></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Title(BSTR *title)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Title(BSTR title)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel10</td>
</tr>
</tbody>
</table>
# TitleState Property

**Description**
Turns ON and OFF the window title. Write a window title with `Title`.

**VB Syntax**
```
win.TitleState = state
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>win</code></td>
<td>A NaWindow (object)</td>
</tr>
<tr>
<td><code>state</code></td>
<td>(boolean)</td>
</tr>
</tbody>
</table>

- **True** - Title ON
- **False** - Title OFF

**Return Type**
Boolean

**Default**
False

**Examples**
```
win.TitleState = True 'Write

titlestate = win.TitleState 'Read
```

**C++ Syntax**
```
HRESULT get_TitleState(VARIANT_BOOL* bState)
HRESULT put_TitleState(VARIANT_BOOL bState)
```

**Interface**
INAWindow
# Tolerance Property

Sets and returns the tolerance value for leveling sweeps.

**VB Syntax**

```plaintext
RxLevel.Tolerance(srcPort) = value
```

**Variable** 

- **RxLevel** - A ReceiverLeveling Object
- **srcPort** (Long Integer) Source port for which to set the tolerance value for Receiver Leveling.

**Note:** If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

- **value** (Double) Tolerance level in dB.

**Return Type** (Double)

**Default** .1 dB

**Examples**

```plaintext
rxLevel.Tolerance (1) = .5  ' Write
value = rxLevel.Tolerance 2' Read
```

**C++ Syntax**

```c++
HRESULT get_Tolerance(long port, double* pVal);
HRESULT put_Tolerance(long port, double newVal);
```

**Interface** IReceiverLevelingConfiguration
**TonePower Property**

**Description**
Sets and returns the power level of the Main Tones. Use with IMD sweep types:

- `naIMDToneCWSweep`
- `naIMDToneCenterFreqSweep`
- `naIMDDeltaFrequencySweep`

When tone power is coupled, setting either F1 or F2 power sets both.

**VB Syntax**

```vbnet
object.TonePower (tone) = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object</code></td>
<td>A SweptIMD or IMSpectrum Object</td>
</tr>
<tr>
<td><code>tone</code></td>
<td>(Enum as NAIMDTonePowerID) Choose from:</td>
</tr>
<tr>
<td>0 - <code>naIMDF1Power</code></td>
<td>F1 tone</td>
</tr>
<tr>
<td>1 - <code>naIMDF2Power</code></td>
<td>F2 tone</td>
</tr>
</tbody>
</table>

| `value` | (Double) Tone power level in dBm. Choose a value between +30 dBm and -30 dBm. |

**Return Type**
Double

**Default**
-20 dBm

**Examples**

```vbnet
imd.TonePower(naIMDF1Power) = 0 'Write
value = imd.TonePower(naIMDF2Power) 'Read
```

**C++ Syntax**

```c
HRESULT get_TonePower(tagNAIMDTonePowerID naIMDF1, double * pVal);
HRESULT put_TonePower(tagNAIMDTonePowerID naIMDF1, double newVal);
```

**Interface**
ISweptIMD
IMSpectrum
**TonePowerSetAt Property - Superseded**

**Description**

Note: This command is replaced with LevelingMethod Property

Sets and returns whether tone power is specified at the DUT input or output.

**VB Syntax**

```
object.TonePowerSetAt = value
```

**Variable (Type) - Description**

- `object` A SweptIMD Object or IMSpectrum Object
- `value` (Enum as NAPortMode)

- **0 - naInput** - Specified power level is set at the DUT input.
- **1 - naOutput** - Specified power level is set at the DUT output.

**Return Type**

Enum

**Default**

0 - naInput

**Examples**

```
imd.TonePowerSetAt = naOutput 'Write
value = ims.TonePowerSetAt 'Read
```

**C++ Syntax**

```
HRESULT get_TonePowerSetAt(tagNAPortMode *pVal)
HRESULT put_TonePowerSetAt(tagNAPortMode pVal)
```

**Interface**

ISweptIMD2

IImsSpectrum2
### TonePowerStart Property

**Description**
Sets and returns the start power level of the Main tones. Use with IMD sweep type= `naIMDTonePowerSweep`.

When tone power is coupled, setting either F1 or F2 power sets both.

**VB Syntax**

```
object.TonePowerStart (tone) = value
```

**Variable (Type) - Description**

- `object` A SweptIMD or IMSpectrum Object
- `tone` (Enum as NAIMDTonePowerID) Choose from:
  - `0` - `naIMDF1Power` - F1 tone
  - `1` - `naIMDF2Power` - F2 tone
- `value` (Double) Start power in dBm. Choose a value between +30 dBm and -30 dBm.

**Return Type**
Double

**Default**
-20 dBm

**Examples**

```
imd.TonePowerStart(NAIMDF1Power) = 0 'Write
value = imd.TonePowerStart(NAIMDF2Power) 'Read
```

**C++ Syntax**

```
HRESULT get_TonePowerStart(tagNAIMDTonePowerID naIMDF1, double * pVal);
HRESULT put_TonePowerStart(tagNAIMDTonePowerID naIMDF1, double newVal);
```

**Interface**
ISweptIMD
IMSpectrum
# TonePowerStop Property

**Description**
Sets and returns the stop power level of the Main tones. Use with IMD sweep type=`naIMDTonePowerSweep`.

When tone power is coupled, setting either F1 or F2 power sets both.

**VB Syntax**

```vb
object.TonePowerStop (tone) = value
```

**Variable (Type) - Description**

- **object**
  A SweptIMD or IMSSpectrum Object
- **tone**
  (Enum as NAIMDTonePowerID) Choose from:
  - 0 - `naIMDF1Power` - F1 tone
  - 1 - `naIMDF2Power` - F2 tone
- **value**
  (Double) Stop power in dBm. Choose a value between +30 dBm and -30 dBm.

**Return Type**
Double

**Default**
-20 dBm

**Examples**

```vb
imd.TonePowerStop (naIMDF1Power) = 0 'Write
value = imd.TonePowerStop (naIMDF2Power) 'Read
```

**C++ Syntax**

```cpp
HRESULT get_TonePowerStop (tagNAIMDTonePowerID naIMDF1, double * pVal);
HRESULT put_TonePowerStop (tagNAIMDTonePowerID naIMDF1, double newVal);
```

**Interface**
ISweptIMD
IMSSpectrum
## TotalIterations Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the total number of iterations required by the last SMART sweep. Returns number of power points for a 2D sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>iter = gca.TotalIterations</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>iter</code></td>
<td>(Integer) Variable to store the returned number of iterations.</td>
</tr>
<tr>
<td><code>gca</code></td>
<td>A GainCompression <strong>(object)</strong></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>data = gca.TotalIterations</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_TotalIterations();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IGainCompression2</td>
</tr>
</tbody>
</table>
### TotalNumberOfPoints Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the total number of points a complete GCA measurement will generate.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• For 2D modes, this is Frequency * Power points</td>
</tr>
<tr>
<td></td>
<td>• For SMART Sweep, this is Frequency points.</td>
</tr>
<tr>
<td></td>
<td>The total can NOT exceed the VNA maximum.</td>
</tr>
<tr>
<td></td>
<td>See Frequency and Power points.</td>
</tr>
</tbody>
</table>

**VB Syntax**

```
value = gca.TotalNumberOfPoints
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(integer) Variable to store the returned total number of points</td>
</tr>
<tr>
<td>gca</td>
<td>A GainCompression (object)</td>
</tr>
</tbody>
</table>

**Return Type**

Integer

**Default**

5226 (201 * 26)

**Example**

```
totPoints = gca.TotalNumberOfPoints 'Read
```

**C++ Syntax**

```
HRESULT get_TotalNumberOfPoints(int* pVal)
```

**Interface**

IGainCompression
### Touchscreen Property

**Description**
Sets and reads the state of the PNA-X Touchscreen (ON and OFF).

This setting remains until changed again from the front-panel or remote command.

**VB Syntax**

```vbnet
app.Touchscreen = state
```

**Variable**

- **app** (Type) - Description
  - An Application (object)
- **state** (boolean)
  - **False** (0) - Disables use of Touchscreen
  - **True** (1) - Enables use of Touchscreen

**Return Type**

Boolean

- **False** - OFF
- **True** - ON

**Default**

**TRUE** when shipped from factory.

**Examples**

- `app.Touchscreen = True 'Write`
- `coupl = app.Touchscreen 'Read`

**C++ Syntax**

- `HRESULT put_Touchscreen(VARIANT_BOOL bState)`
- `HRESULT get_Touchscreen(VARIANT_BOOL *bState)`

**Interface**

IApplication12
### TraceHoldType Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the type of trace hold operation to perform.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.TraceHoldType = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td><code>(Enum as NATraceHoldType) - Choose from:</code></td>
</tr>
<tr>
<td><code>naTraceHoldOff</code></td>
<td>Disables the Trace Hold feature.</td>
</tr>
<tr>
<td><code>naTraceHoldMinimum</code></td>
<td>Sets Trace Hold to store the lowest measured data points.</td>
</tr>
<tr>
<td><code>naTraceHoldMaximum</code></td>
<td>Sets Trace Hold to store the highest measured data points.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td><code>naTraceHoldOff</code></td>
</tr>
<tr>
<td><strong>VB Examples</strong></td>
<td><code>meas.TraceHoldType = naTraceHoldOff 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>hold = meas.TraceHoldType 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_TraceHoldType(tag NATraceHoldType *pVal)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_TraceHoldType(tag NATraceHoldType newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement16</td>
</tr>
</tbody>
</table>
TraceMath Property

Description: Performs math operations on the measurement object and the trace stored in memory. (There MUST be a trace stored in Memory to perform math. See Meas.DataToMemory method.)

VB Syntax: `meas.TraceMath = value`

Variable (Type) - Description
- `meas`: A Measurement (object)
- `value`: (enum NAMathOperation) - Choose from:
  - 0 - naDataNormal
  - 1 - naDataMinusMemory
  - 2 - naDataPlusMemory
  - 3 - naDataDivMemory
  - 4 - naDataTimesMemory

Return Type: NAMathOperation

Default: Normal (0)

Examples:
- `meas.TraceMath = naDataMinusMemory`  
  (Write)
- `mathOperation = meas.TraceMath`  
  (Read)

C++ Syntax:
- `HRESULT get_TraceMath(tagNAMathOperation* pMathOp)`
- `HRESULT put_TraceMath(tagNAMathOperation mathOp)`

Interface: I Measurement
### TraceMax Property

**Description**
Maximizes (isolates) or restores the active trace in the active window. When TraceMax is ON, the active trace is the ONLY trace on the display. All other traces are hidden.

**VB Syntax**
\[ meas.TraceMax = state \]

**Variable**
- \( meas \) - A Measurement (object)
- \( state \) - (boolean) - Choose from:
  - **True** - Maximizes / isolates the active trace in the window.
  - **False** - Restores other traces to be viewed in the window.

**Return Type**
Boolean

**Default**
False

**Examples**
\[ meas.TraceMax = True \]
\[ state = meas.TraceMax \]

**C++ Syntax**
- HRESULT get_TraceMax(VARIANT_BOOL bState)
- HRESULT put_TraceMax(VARIANT_BOOL* bState)

**Interface**
IMeasurement10
# TraceTitle Property

**Description**  Writes and reads data for the trace title area.

The trace title is embedded in the trace status field. [Learn more about Trace Titles](#). The title is turned ON and OFF using TraceTitleState.

**VB Syntax**

```
meas.TraceTitle = value
```

**Variable (Type) - Description**

- `meas` A Measurement *(object)*
- `value` *(string)* - Title to be displayed. Any characters (no spaces), enclosed with quotes.

**Return Type**  String

**Default**  Not Applicable

**Examples**

```vbscript
meas.TraceTitle = "My new s11 measurement"
```

**C++ Syntax**

```
HRESULT get_TraceTitle(BSTR *title);
HRESULT put_TraceTitle(BSTR title);
```

**Interface**  IMeasurement8
## TraceTitleState Property

**Description**  
Turns display of the Trace Title ON or OFF. When turned OFF, the previous trace title returns. Create a trace title using `TraceTitle Property`.

**VB Syntax**  
`meas.TraceTitleState = value`

**Variable**  
- **Type** - Description  
  - `meas`  
  - A Measurement *(object)*
  - `value`  
  - *(boolean)* - Choose from:
    - **True** - Turns the trace title ON
    - **False** - Turns the trace title OFF

**Return Type**  
Boolean

**Default**  
False

**Examples**  
```
meas.TraceTitleState = False
```
```
title = TraceTitleState 'Read
```

**C++ Syntax**  
```
HRESULT get_TraceTitleState(VARIANT_BOOL *isTitleON);
HRESULT put_TraceTitleState(VARIANT_BOOL isTitleON);
```

**Interface**  
IMeasurement8
Tracking Property

Description: This property, when on, executes the search function (marker.SearchFunction) every sweep.

In effect, turning Tracking ON is the same as executing one of the immediate, one-time, "Search..." methods (such as SearchMin, SearchMax) for every sweep.

VB Syntax: mark.Tracking = state

Variable Description:
- **mark**: A Marker (object)
- **state**: (boolean) - Tracking state. Choose from:
  - **False**: Tracking OFF
  - **True**: Tracking ON

Return Type: Boolean

Default: False

Examples:
- `mark.Tracking = False` 'Write
- `markTracking = mark.Type` 'Read

C++ Syntax:
- HRESULT put_Tracking(VARIANT_BOOL bOn)
- HRESULT get_Tracking(VARIANT_BOOL * pbOn)

Interface: IMarker
### TrackingChannel Property

**Description**
Sets and returns the IMD channel number to which the IM Spectrum channel is coupled. Use `TrackingEnable` to enable tracking.

**VB Syntax**
```vbnet
ims.TrackingChannel = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ims</code></td>
<td>An <code>IMSpectrum</code> Object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Integer) Existing IMD channel number to which frequency and power settings are coupled.</td>
</tr>
</tbody>
</table>

**Return Type**
Long Integer

**Default**
First existing IMD channel

**Examples**
```vbnet
ims.TrackingChannel = 1 'Write
value = ims.TrackingChannel 'Read
```

**C++ Syntax**
```c++
HRESULT get_TrackingChannel(long *pVal)
HRESULT put_TrackingChannel(long newVal)
```

**Interface**
`IIMSpectrum`
### TrackingEnable Property

**Description**  
When an IMD channel exists, allows the IM Spectrum frequency and power setting to track (couple with) the IMD channel settings. Use `TrackingChannel` to set the channel number to track.

**VB Syntax**  
```vb
ims.TrackingEnable = value
```

**Variable**  
*ims* - An IMSpectrum Object  
*value* - (Boolean) Tracking state. Choose from:

- **True** - IM Spectrum frequency and power settings track the IMD channel settings.
- **False** - IM Spectrum frequency and power settings are specified in the IMS channel.

**Return Type**  
Boolean

**Default**  
False

**Examples**  
```vb
ims.TrackingEnable = True 'Write
value = ims.TrackingEnable 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_TrackingEnable(VARIANT_BOOL* bValue)
HRESULT put_TrackingEnable(VARIANT_BOOL newVal)
```

**Interface**  
IIMSpectrum
## TrackingManualStepEnable Property

**Description**
Sets and returns the step sweep mode for the IM Spectrum channel.

**VB Syntax**
```vbnet
ims.TrackingManualStepEnable = value
```

**Variable**
- **ims** (Type): An IMSpectrum Object
- **value** (Boolean): Choose from:
  - **False** - Automatic Step
  - **True** - Manual Step

**Return Type**
Boolean

**Default**
False - Automatic Step

**Examples**
```vbnet
ims.TrackingManualStepEnable = True 'Write
value = ims.TrackingManualStepEnable 'Read
```

**C++ Syntax**
```c++
HRESULT get_TrackingManualStepEnable(VARIANT BOOL *bVal)
HRESULT put_TrackingManualStepEnable(VARIANT BOOL newVal)
```

**Interface**
IIMSpectrum
### TrackingStepIndex Property

**Description**  
When `TrackingManualStepEnable = True` (Manual step), sets and returns the data point number at which the IM spectrum measurement occurs.

**VB Syntax**  
`ims.TrackingStepIndex = value`

**Variable**  
- **Type** - Description  
  - `ims` An IMSpectrum Object  
  - `value` (Integer) Existing IMD channel number to which frequency and power settings are coupled.

**Return Type**  
Long Integer

**Default**  
1

**Examples**  
- `ims.TrackingStepIndex = 50`  
- `value = ims.TrackingStepIndex`

**C++ Syntax**  
- `HRESULT get_TrackingStepIndex(long *pVal)`  
- `HRESULT put_TrackingStepIndex(long newVal)`

**Interface**  
IIMSpectrum
### TreatMkr10AsReference Property

**Description**
Set and return whether to treat marker 10 as a reference marker.

**VB Syntax**
```
pref.TreatMkr10AsReference = value
```

**Variable**
- **(Type)**: Preference
  - **pref**: A Preferences (object)
- **value**: (Boolean) - Choose from:
  - **True**: Marker 10 is always a reference marker (Pre A.10.40 behavior).
  - **False**: Marker 10 is just another marker. See Reference Marker commands

**Return Type**
Boolean

**Default**
True

**Examples**
```
pref.TreatMkr10AsReference = True 'Write
value = pref.TreatMkr10AsReference 'Read
```

**C++ Syntax**
```
HRESULT get_TreatMkr10AsReference (VARIANT_BOOL* preference);
HRESULT put_TreatMkr10AsReference (VARIANT_BOOL val)
```

**Interface**
IPreferences15
## TriggerSignal Property - Superseded

**Description**  
Note: This command has been replaced by Source Property

Sets or returns the trigger source.

**VB Syntax**  
`app.TriggerSignal = value`

**Variable**  
*(Type)* - Description
- `app`: An Application *(object)*
- `value`: *(enum NATriggerSignal)* - Choose from:
  1. **naTriggerInternal** - free run
  2. **naTriggerExternalPositive** - a trigger signal is generated when a TTL high is sensed on the external trigger pin of the Aux IO connector
  3. **naTriggerExternalNegative** - a trigger signal is generated when a TTL low is sensed on the external trigger pin of the Aux IO connector.
  4. **naTriggerExternalHigh** - a trigger signal is generated when a TTL high is sensed on the external trigger pin of the Aux IO connector
  5. **naTriggerExternalLow** - a trigger signal is generated when a TTL low is sensed on the external trigger pin of the Aux IO connector

**Return Type**  
Long Integer

**Default**  
naTriggerInternal

**Examples**  
- `app.TriggerSignal = naTriggerExternalPositive` *Write*
- `trigsig = app.TriggerSignal` *Read*

**C++ Syntax**  
`HRESULT get_TriggerSignal(tagNATriggerSignal *pSignal)`
`HRESULT put_TriggerSignal(tagNATriggerSignal signal)`

**Interface**  
IApplication
### TriggerType Property - Superseded

**Description**  
**Note:** This property has been replaced with `Scope` Property.

Sets or returns the trigger type which determines the scope of a trigger signal.

**VB Syntax**  
```
app.TriggerType = value
```

**Variable (Type) - Description**
- `app` An Application `(object)`
- `value` `(enum NATriggerType)` - Trigger type. Choose from:
  - 0 - `naGlobalTrigger` - a trigger signal is applied to all triggerable channels
  - 1 - `naChannelTrigger` - a trigger signal is applied to the current channel. The next trigger signal will be applied to the next channel; not necessarily channel 1-2-3-4.

**Return Type**  
Long Integer

**Default**  
`naGlobalTrigger`

**Examples**
```
app.TriggerType = naGlobalTrigger  'Write
trigtyp = app.TriggerType  'Read
```

**C++ Syntax**
```
HRESULT get_TriggerType(tagNATriggerType *pTrigger)
HRESULT put_TriggerType(tagNATriggerType trigger)
```

**Interface**  
`IApplication`
# TriggerADCLevelState Property

**Description**  
Set and read the ON/OFF state of a measurement trigger event that will occur whenever the ADC level is greater than the value specified using the TriggerADCLevelValue command.

**VB Syntax**  
```
sa.TriggerADCLevelState = value
```

**Variable**  
- `sa` *(Type)* - Description  
  - A SpectrumAnalyzer *(object)*
- `value` *(Boolean)* - Choose from:
  - 0 - OFF - ADC measurement trigger OFF.
  - 1 - ON - ADC measurement trigger ON.

Learn about these settings.

**Return Type**  
Boolean

**Default**  
0

**Examples**  
```
sa.TriggerADCLevelState = OFF   Write
value = sa.TriggerADCLevelState  'Read
```

See an example program.

**C++ Syntax**  
```
HRESULT put_TriggerADCLevelState(VARIANT_BOOL bEnable);
HRESULT get_TriggerADCLevelState(VARIANT_BOOL* bEnable);
```

**Interface**  
ISpectrumAnalyzer
# TriggerADCLevelValue Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the ADC trigger level.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>sa.TriggerADCLevelValue = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>sa</code></td>
<td>A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) Choose a value between 0 and 16383.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Default</td>
<td>100</td>
</tr>
<tr>
<td>Examples</td>
<td><code>sa.TriggerADCLevelValue = 256</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>value = sa.TriggerADCLevelValue</code> 'Read</td>
</tr>
</tbody>
</table>

See an example program.

**C++ Syntax**

- `HRESULT put_TriggerADCLevelValue(long val);`
- `HRESULT get_TriggerADCLevelValue(long* val);`

**Interface**

ISpectrumAnalyzer
Write/Read

**TriggerDelay Property**

**Description**
Sets and reads the trigger delay for all measurements (GLOBAL). This delay is only applied while in `app.Source = naTriggerSourceExternal` and `trigsetup.Scope = naGlobalTrigger`. After an external trigger is applied, the start of the sweep is delayed for the specified delay value plus any inherent latency.

To apply a trigger delay for a channel only, use the `ExternalTriggerDelay Property`.

**VB Syntax**
```vbnet
app.TriggerDelay = value
```

**Variable (Type) - Description**
- `app` An Application (object)
- `value` Double - Trigger delay value in seconds. Range is from 0 to 3.

**Return Type**
Double

**Default**
0

**Examples**
```vbnet
app.TriggerDelay = .003  'Write
delay = app.TriggerDelay  'Read
```

**C++ Syntax**
```c++
HRESULT get_TriggerDelay(double *delay);
HRESULT put_TriggerDelay(double delay)
```

**Interface**
IApplication
TriggerInPin Property

Description: When Trigger mode is set to Hardware List for SMU devices, this command sets and returns the B2900 Digital IO pin to use for Trigger IN.

VB Syntax:  
```vbnet
extDC.TriggerInPin = value
```

Variable (Type) - Description:
- `extDC` An `ExternalDCDevice` (object)
- `value` (Long) Pin number for Trigger IN. Choose a pin number from 1 to 14.

Return Type: Long

Default: 1

Examples:
```vbnet
extDC.TriggerInPin = 3 'Write
bool = extDC.TriggerInPin 'Read
```

C++ Syntax:
```cpp
HRESULT get_TriggerInPin (Long *pValue)
HRESULT put_TriggerInPin (long newVal)
```

Interface: IExternalDCDevice2
**TriggerInPolarity Property**

**Description**
Specifies the polarity of the trigger IN signal.

- **AuxTrigger Object** - Sets the polarity to which the rear-panel AuxTrig IN responds.
- **PulseGenerator Object** - Sets the polarity of trigger to which the internal pulse generators will respond when being externally triggered. **Note:** This feature requires DSP version: **4.0 FPGA:34** or higher. Learn more.

**Note:** Used on PNA-X ONLY.

**VB Syntax**

```vbnet
object.TriggerInPolarity = value
```

**Variable** *(Type)* - Description

- `object` An AuxTrigger (object) or
  A PulseGenerator (object)

- `value` *(enum NATriggerPolarity)* - Choose from:
  - `naTriggerPositive` VNA responds to rising edge or HIGH level
  - `naTriggerNegative` VNA responds to falling edge or LOW level.

Set Edge or Level triggering using **TriggerInType Property**

**Return Type**
Enum

**Default**
AuxTriggerIn Object - naTriggerNegative

PulseGenerator Object - naTriggerPositive. Also the polarity used when the PNA-X does not have the required DSP hardware

**Examples**

```vbnet
obj.TriggerInPolarity = naTriggerPositive 'Write
value = obj.TriggerInPolarity 'Read the value
```

**C++ Syntax**

```c++
HRESULT get_TriggerInPolarity(enum NATriggerPolarity *val);
HRESULT put_TriggerInPolarity(enum NATriggerPolarity val);
```

**Interface**
IAuxTrigger

IPulseGenerator3
## TriggerInType Property

**Description**
Specifies the type of trigger input being supplied to the VNA.

- **AuxTrigger Object** - Sets the type to which the rear-panel AuxTrig IN responds.
- **PulseGenerator Object** - Sets the type of trigger to which the internal pulse generators will respond when being externally triggered. **Note:** This feature requires DSP version: 4.0 FPGA:34 or higher. Learn more.

**Note:** Use on PNA-X ONLY.

### VB Syntax

```
obj.TriggerInType = value
```

### Variable

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>An AuxTrigger (object) or</td>
</tr>
<tr>
<td></td>
<td>A PulseGenerator (object)</td>
</tr>
<tr>
<td>value</td>
<td>(enum NATriggerSignalType)</td>
</tr>
<tr>
<td></td>
<td>Choose from:</td>
</tr>
<tr>
<td></td>
<td>naTriggerEdge VNA responds</td>
</tr>
<tr>
<td></td>
<td>to the edge (rising or falling) of a signal</td>
</tr>
<tr>
<td></td>
<td>naTriggerLevel VNA responds</td>
</tr>
<tr>
<td></td>
<td>to the level (HIGH or LOW) of a signal</td>
</tr>
<tr>
<td></td>
<td>Use TriggerInPolarity to set Positive or Negative polarity.</td>
</tr>
</tbody>
</table>

### Return Type

Enum

### Default

**naTriggerLevel** - Also the type used for the PulseGenerator Object when the PNA-X does not have the required DSP hardware

### Examples

```
obj.TriggerInType = naTriggerEdge 'Write
value = obj.TriggerInType 'Read the value
```

### C++ Syntax

```
HRESULT get_TriggerInType(enum NATriggerSignalType *val);
HRESULT put_TriggerInType(enum NATriggerSignalType val);
```

### Interface

IAuxTrigger

IPulseGenerator3
**TriggerMode (ExtendedProperties) Property**

- **Description**: Sets and returns the trigger mode for an external source.

  - **VB Syntax**: `extSource.TriggerMode = value`  
  - **Variable** *(Type)* - Description  
    - `extSource` An ExternalSource Object *(object)*  
    - `value` *(enum NAExtDevTriggerMode)* - Choose from:  
      - 0 - `naExtDevTriggerModeCW`  
      - 1 - `naExtDevTriggerModeHW`  

- **Return Type**: Enum  
  - **Default**: 0 - `naExtDevTriggerModeCW`  

- **Examples**:  
  - `extSource.TriggerMode = naExtDevTriggerModeCW` *(Write)*  
  - `tm = extSource.TriggerMode` *(Read)*  

- **C++ Syntax**:  
  - `HRESULT get_TriggerMode (tagNAExtDevTriggerMode *pMode)`  
  - `HRESULT put_TriggerMode (tagNAExtDevTriggerMode newMode)`  

- **Interface**: IExternalSource
# TriggerOutDuration Property

**Description**  
Specifies the width of the output pulse, which is the time that the Aux trigger output will be asserted.

**VB Syntax**  
`auxTrig.TriggerOutDuration = value`

**Variable**  
- **Type** - Description
- **Type** - An AuxTrigger (object)
- **Value** - (single) - Duration value in seconds. Choose a value between 1E-6 and 1.

**Return Type**  
Double

**Default**  
1E-6 sec

**Examples**  
- `auxTrig.TriggerOutDuration = 1e-3`  
- `value = auxTrig.TriggerOutDuration`

**C++ Syntax**  
- HRESULT get_TriggerOutDuration(double *val);
- HRESULT put_TriggerOutDuration(double val);

**Interface**  
IAuxTrigger
## TriggerOutInterval Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies how often a trigger output signal is sent.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>auxTrig.TriggerOutInterval = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong> (Type) - Description</td>
<td></td>
</tr>
<tr>
<td><code>auxTrig</code></td>
<td>An <code>AuxTrigger</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(single) - Choose from:</td>
</tr>
<tr>
<td><strong>0 - naTriggerModePoint</strong></td>
<td>a single data point is measured with each trigger signal the channel receives. Subsequent trigger signals continue to go to the channel in Point mode until the channel measurements are complete. This is effectively the same as trigger point mode.</td>
</tr>
<tr>
<td><strong>1 - naTriggerModeMeasurement</strong></td>
<td>entire traces are swept with a trigger signal, which and how many traces depends on the Scope setting.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td><strong>1 - naTriggerModeMeasurement</strong></td>
</tr>
</tbody>
</table>
| **Examples** | ```vba
auxTrig.TriggerOutInterval = naTriggerModeMeasurement 'Write
value = auxTrig.TriggerOutInterval 'Read the value
``` |
| **C++ Syntax** | HRESULT get_TriggerOutInterval(enum NATriggerMode *val);
HRESULT put_TriggerOutInterval(enum NATriggerMode val); |
| Interface   | IAuxTrigger                                         |
**TriggerOutPolarity Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the polarity of the trigger output signal being supplied by the VNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>auxTrig.TriggerOutPolarity = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>auxTrig</code></td>
<td>An AuxTrigger (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum NATriggerPolarity) - Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>naTriggerPositive</strong> VNA sends positive going (active HIGH) pulse.</td>
</tr>
<tr>
<td></td>
<td><strong>naTriggerNegative</strong> VNA sends negative going (active LOW) pulse.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum</td>
</tr>
<tr>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Examples</td>
<td><code>auxTrig.TriggerOutPolarity = naTriggerPositive</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>value = auxTrig.TriggerOutPolarity</code> 'Read the value'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_TriggerOutPolarity(enum NATriggerPolarity *val);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_TriggerOutPolarity(enum NATriggerPolarity val);</td>
</tr>
<tr>
<td>Interface</td>
<td>I AuxTrigger</td>
</tr>
</tbody>
</table>
# TriggerOutPosition Property

**Description**  
Specifies whether the Aux trigger out signal is sent Before or After the acquisition.

**VB Syntax**  
```
auxTrig.TriggerOutPosition = value
```

**Variable**  
*(Type)* - Description

- `auxTrig` An AuxTrigger (object)
- `value` (enum NATriggerPosition) Choose from:

  - **naTriggerOutBeforeAcquire**  
    Use if the external device needs to be triggered before the data is acquired, such as a power meter.

  - **naTriggerOutAfterAcquire**  
    Use if the external device needs to be triggered just after data has been acquired, such as an external source. This could be more efficient since it allows the external device to get ready for the next acquisition at the same time as the VNA.

**Return Type**  
Enum

**Default**  
**naTriggerOutAfterAcquire**

**Examples**  
```
auxTrig.TriggerOutPosition = naTriggerOutAfterAcquire  'Write
value = auxTrig.TriggerOutPosition  'Read the value
```

**C++ Syntax**  
```
HRESULT get_TriggerOutPosition(enum NATriggerPosition *val);
HRESULT put_TriggerOutPosition(NATriggerPosition val);
```

**Interface**  
IAuxTrigger
### TriggerOutputEnabled Property - Superseded

**Description**

Use `AUX.Enable` to enable AUXI/O triggering.

Use `Trigger Source= External` to enable Meas Trig Ready output.

Enables the VNA to send trigger signals out the rear-panel TRIGGER OUT connector.

**VB Syntax**

```
trigsetup.TriggerOutputEnabled = boolean
```

**Variable**

- **Type**
  - `A TriggerSetup2 (object)`
  - `boolean`

Choose from:

- **False** - VNA does NOT send output trigger signals.
- **True** - VNA sends output trigger signals.

**Return Type**

`Boolean`

**Default**

`False`

**Examples**

```
trigsetup.TriggerOutputEnabled = True 'Write
atba = trigsetup.TriggerOutputEnabled 'Read
```

**C++ Syntax**

```cpp
HRESULT get_TriggerOutputEnabled( BOOL *pVal);
HRESULT put_TriggerOutputEnabled( BOOL newVal);
```

**Interface**

`ITriggerSetup2`
**TriggerPeriodicCounterState Property**

**Description**
Set and read the ON/OFF state of a measurement trigger event based on the specified period set using the TriggerPeriodicCounterValue command.

**VB Syntax**

```
sa.TriggerPeriodicCounterState = value
```

**Variable (Type) - Description**

- `sa` A SpectrumAnalyzer (object)
- `value` (Boolean) Choose from:
  - 0 - OFF - Periodic counter OFF.
  - 1 - ON - Periodic counter ON.

**Learn about these settings.**

**Return Type**
Boolean

**Default**
0

**Examples**

```
sa.TriggerPeriodicCounterState = OFF   Write
value = sa.TriggerPeriodicCounterState 'Read
```

See an example program.

**C++ Syntax**

```c++
HRESULT put_TriggerPeriodicCounterState(VARIANT_BOOL bEnable);
HRESULT get_TriggerPeriodicCounterState(VARIANT_BOOL* bEnable);
```

**Interface**
ISpectrumAnalyzer
### TriggerPeriodicCounterValue Property

**Description**  Set and read the periodic counter value. This command initiates a measurement trigger event based on the specified period.

**VB Syntax**  
```
sa.TriggerPeriodicCounterValue = value
```

**Variable**  
- `sa`  A `SpectrumAnalyzer` *(object)*
- `value` *(Long)* Choose a value between 0 and 2147483647.

Learn about these settings.

**Return Type**  Long

**Default**  256

**Examples**  
```
sa.TriggerPeriodicCounterValue = 256  'Write

value = sa.TriggerPeriodicCounterValue 'Read
```

See an example program.

**C++ Syntax**  
```c++
HRESULT put_TriggerPeriodicCounterValue(long val);

HRESULT get_TriggerPeriodicCounterValue(long* val);
```

**Interface**  ISpectrumAnalyzer

---

2503
## TriggerPort Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the VNA port through which an external source is to be triggered.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>extSource.TriggerPort = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>extSource</code></td>
<td>An ExternalSource Object (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum NAExtDevTriggerPort) - Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naExtDevTriggerPortBNC1 (VNA 'C' models)</td>
</tr>
<tr>
<td></td>
<td>1 - naExtDevTriggerPortAux1 (PNA-X models)</td>
</tr>
<tr>
<td></td>
<td>2 - naExtDevTriggerPortAux2 (PNA-X models)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum</td>
</tr>
<tr>
<td>Default</td>
<td>For VNA 'C' models - BNC1</td>
</tr>
<tr>
<td></td>
<td>For PNA-X models - Aux1</td>
</tr>
<tr>
<td>Examples</td>
<td><code>extSource.TriggerPort = 1 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>trigpt = extSource.TriggerPort 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_TriggerPort (tagNAExtDevTriggerPort *pValue)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_TriggerPort (tagNAExtDevTriggerPort newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IExternalSource</td>
</tr>
</tbody>
</table>
### TuningIFBW Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set the IF Bandwidth for Broadband and Precise tuning sweeps.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>obj.TuningIFBW = value</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>obj</td>
<td>An EmbeddedLO (object) or A ConverterEmbeddedLO (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Double) IF Bandwidth</td>
</tr>
<tr>
<td>Return Type</td>
<td>(Double)</td>
</tr>
<tr>
<td>Default</td>
<td>30 kHz</td>
</tr>
<tr>
<td>Examples</td>
<td>embedLO.TuningIFBW = 10e3 <code>write</code></td>
</tr>
<tr>
<td></td>
<td>value = embedLO.TuningIFBW <code>read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_TuningIFBW(double* ifbw);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_TuningIFBW(double ifbw);</td>
</tr>
<tr>
<td>Interface</td>
<td>IEmbeddedLO</td>
</tr>
</tbody>
</table>
## TuningMode Property

**Description**  
Sets and returns the method used to determine the embedded LO Frequency.

**VB Syntax**  
`obj.TuningMode = value`

**Variable**  
*(Type)* - Description

- `obj` An EmbeddedLO *(object)* or  
  A ConverterEmbeddedLO *(object)*

- `value` *(Enum as NAEembeddedLOTuningMode)*
  Tuning mode. Choose from:
  
  - 0 - naEmbeddedLOTuningMode_Broadband_And_Precise
  - 1 - naEmbeddedLOTuningMode_Precise_Only
  - 2 - naEmbeddedLOTuningMode_None

**Return Type** *(Enum)*

**Default**  
0 - naEmbeddedLOTuningMode_Broadband_And_Precise

**Examples**

```vbnet
eembedLO.TuningMode = naEmbeddedLOTuningMode_None 'write
value = embedLO.TuningMode 'read
```

**C++ Syntax**

```csharp
HRESULT get_TuningMode(enum NAEembeddedLOTuningMode* mode);

HRESULT put_TuningMode(enum NAEembeddedLOTuningMode mode);
```

**Interface**  
IEmbededLO
# TuningSweepInterval Property

**Description**
Set how often a tuning sweep is performed.

**VB Syntax**
```vbnet
obj.TuningSweepInterval = value
```

**Variable**
<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>obj</code></td>
<td>An <strong>EmbeddedLO</strong> (object) or <strong>ConverterEmbeddedLO</strong> (object)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td><strong>(Long)</strong> Tuning sweep interval.</td>
</tr>
</tbody>
</table>

**Return Type**
**(Long)**

**Default**
1

**Examples**
```vbnet
eMBEDLO.TuningSweepInterval = 3 'write .. tuning is performed every third measurement sweep
value = EMBEDLO.TuningSweepInterval 'read
```

**C++ Syntax**
```cpp
HRESULT get_TuningSweepInterval(long* interval);

HRESULT put_TuningSweepInterval(long interval);
```

**Interface**
**IEmbeddedLO**
# TwoPointGroupDelayAperture Property

**Description**  
Sets the default group delay aperture setting.

**VB Syntax**  
`pref.TwoPointGroupDelayAperture = value`

**Variable**  
- **Type** - Description
  - `pref` A Preferences (object)
  - `value` (Boolean) - Choose from:
    - True - Set the default group delay aperture setting to two points.
    - False - Set the default group delay aperture setting to 11 points.

**Return Type**  
Boolean

**Default**  
False

**Examples**  
- `pref.TwoPointGroupDelayAperture = True`  
- `gda = pref.TwoPointGroupDelayAperture`

**C++ Syntax**  
- `HRESULT get_TwoPointGroupDelayAperture(VARIANT_BOOL* pVal);`
- `HRESULT put_TwoPointGroupDelayAperture(VARIANT_BOOL pVal);`

**Interface**  
IPreferences11
### TriggerMode Property

**Description**  
These settings determine what EACH signal will trigger.

**Note:** Setting Point and EverySweep mode forces \( \text{Trigger.Scope} = \text{naChannelTrigger} \).

**VB Syntax**  
\[ \text{chan.TriggerMode} = \text{value} \]

**Variable**  
- **chan**  
  A Channel (object)

- **value**  
  (enum NATriggerMode) - Choose from:

  0 - **naTriggerModePoint** - Each Manual or External trigger signal causes one data point to be measured.

  1 - **naTriggerModeMeasurement** (superseded - still works but replaced with a more descriptive enum)

  1 - **naTriggerModeChannel** - Each trigger signal causes ALL traces in that channel to be swept.

  2 - **naTriggerModeEverySweep** - Each Manual or External trigger signal causes ALL traces that share a source port to be swept.

  3 - **naTriggerModeTrace** - Allowed ONLY when \( \text{PointSweepState} \) is enabled. Each trigger signal causes two identical measurements to be triggered separately - one trigger signal is required for each measurement. Other trigger mode settings cause two identical parameters to be measured simultaneously.

**Return Type**  
Long Integer

**Default**  
1 - **naTriggerModeChannel**

**Examples**  
\[ \text{chan.TriggerMode} = \text{naTriggerModePoint} \ 'Write} \]

\[ \text{trigtyp} = \text{chan.TriggerMode} \ 'Read} \]

**C++ Syntax**  
HRESULT get_TriggerMode (tagNATriggerMode *pMode)  
HRESULT put_TriggerMode (tagNATriggerMode newMode)

**Interface**  
IChannel
### Type (calstd) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Returns the type of calibration standard.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>calstd.Type = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(enum NACalStandardType) - Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <code>naOpen</code></td>
</tr>
<tr>
<td></td>
<td>1 - <code>naShort</code></td>
</tr>
<tr>
<td></td>
<td>2 - <code>naLoad</code></td>
</tr>
<tr>
<td></td>
<td>3 - <code>naThru</code></td>
</tr>
<tr>
<td>Return Type</td>
<td>Long Integer</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>calstd.Type = naOpen 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>standardtype = calstd.Type 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_Type(tagNACalStandardType *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Type(tagNACalStandardType newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>ICalStandard</td>
</tr>
</tbody>
</table>
## Type (DFT) Property

**Description**: Sets and returns the DFT record size type. The DFT `RecordSize` is based on the `ForceADCRrecordSize` and the DFT record size type. The DFT record size is always equal or larger than the ADC record size.

**VB Syntax**: `sa.dft.Type = value`

**Variable**

- **(Type)** - Description

  - `sa.dft` A `SpectrumAnalyzerDFT (object)`
  - `value` *(enum NASADFTSizeMode)* - Choose from:
    
    0  naSADFT2ToN: (Power of 2) Sets the DFT record size to the next power of 2 greater than or equal to the current ADC record size.
    
    1  naSADFTOptimizeRadix: (Optimized Radix) Sets the DFT record to the next integer number that can be decomposed with 2,3,5,7,11,13 radixes.
    
    2  naSADFTArbitrary: (Arbitrary) Sets DFT record size equal to the ADC record size. If the current ADC record size is a large prime number, then the DFT can be very slow.
    
    3  naSADFTOptimizeSpeed: (Fastest) Sets the DFT record size as close as possible to the ADC record size (larger or equal) while optimizing processing speed.

**Return Type**: Enum as NASADFTSizeMode

**Default**: naSADFTOptimizeSpeed

**Note**: In previous releases the default was naSADFT2ToN (Power of 2).

**Examples**

```vbnet
sa.dft.Type = naSADFTOptimizeRadix 'Write
DFTType = sa.dft.Type 'Read
```

**C++ Syntax**

```c++
HRESULT get_Type(tagNASADFTSizeMode* pVal)
HRESULT put_Type(tagNASADFTSizeMode newVal)
```

**Interface**: ISpectrumAnalyzerDFT
### Type (testset) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the testset model number.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>tset.Type model</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td></td>
</tr>
<tr>
<td><code>model</code></td>
<td>(String) Variable to return the Test set model</td>
</tr>
<tr>
<td><code>tset</code></td>
<td>A <code>TestsetControl</code> object.</td>
</tr>
<tr>
<td>Obtained from the <code>ExternalTestsets</code> collection.</td>
<td></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>testset.type model</code></td>
</tr>
</tbody>
</table>

See [External Testset Program](#).

| **C++ Syntax** | HRESULT `get_Type(BSTR *ptype);` |
| **Interface** | ITestsetControl |
**TZImag Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Returns the TZImag value (the Imaginary Terminal Impedance value) for the calibration standard. Only applicable when &quot;Type&quot; is set to <strong>naArbitraryImpedance</strong>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>To set the other resistance values, use <strong>TZReal</strong>.</td>
<td></td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>calstd.TZImag = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong> (Type) - Description</td>
<td><strong>calstd</strong> is a CalStandard <em>(object)</em>. Use <code>calKit.GetCalStandard</code> to get a handle to the standard. <strong>value</strong> is a Single - Value for TZImag in Ohms.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Single</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>calstd.TZImag = 15 'Write the value of TZImag to 15 Ohms</code></td>
</tr>
<tr>
<td><code>imp0 = calstd.TZImag 'Read the value of TZImag</code></td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_TZImag(float *pVal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_TZImag(float newVal);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalStandard2</td>
</tr>
</tbody>
</table>
### TZReal Property

**Description**
Sets and Returns the TZReal value (the real Terminal Impedance value) for the calibration standard. Only applicable when "Type" is set to **naArbitraryImpedance**.

To set the other resistance values, use **TZImag**

**VB Syntax**
```
calstd.TZReal = value
```

**Variable (Type) - Description**
- `calstd` A CalStandard (**object**). Use calKit.GetCalStandard to get a handle to the standard.
- `value` **(single)** - Value for TZReal in Ohms

**Return Type**
Single

**Default**
Not Applicable

**Examples**
```
calstd.TZReal = 15  'Write the value of TZReal to 15 Ohms
imp0 = calstd.TZReal  'Read the value of TZReal
```

**C++ Syntax**
```
HRESULT get_TZReal(float *pVal);
HRESULT put_TZReal(float newVal);
```

**Interface**
ICalStandard2
About Dynamic Uncertainty

**UncertaintyEnabled Property**

| Description | Sets and returns the ON/OFF state which determines if the calibration that is about to be performed will support Dynamic Uncertainty for S-Parameters (Opt. S93015A/B).

Dynamic Uncertainty for S-Parameters is supported ONLY for calibrations on standard S-Parameter channels. Calibrations performed with that feature enabled do NOT support the use of ALL traditional Guided calibration commands.

These existing commands are used for the performing of the calibration:

- CalKitType
- Initialize
- GenerateSteps
- GetStepDescription
- AcquireStep
- GenerateErrorTerms

These commands might also optionally be used in performing the cal:

- GetCompatibleCalKits
- OrientECALModule
- ECALPortMapEx

Dynamic Uncertainty must be enabled using this command before starting the calibration procedure because this command controls the way connectors and calkits are assigned to ports during calibration. Therefore, this command must be enabled before any of the following commands to ensure that the connector and calkit settings will be setqueried correctly:

- CalKitType
- GetCompatibleCalKits
- ConnectorType
**VB Syntax**

`guidedCal.UncertaintyEnabled = value`

**Variable**

*guidedCal* - An *GuidedCalibration* Object

*value* - (Boolean) Enable state. Choose from:

- **True** - The next calibration initialized for the channel will support Dynamic Uncertainties for S-Parameters.
- **False** - The next calibration initialized for the channel will NOT support Dynamic Uncertainties for S-Parameters.

**Return Type**

Boolean

**Default**

False

**Examples**

`guided.UncertaintyEnabled = True`

See example program

**C++ Syntax**

`HRESULT get_UncertaintyEnabled(VARIANT_BOOL* pState);`

`HRESULT put_UncertaintyEnabled(VARIANT_BOOL state);`

**Interface**

`IGuidedCalibration1`
### UncertaintyFile Property

**Description**
Sets and returns a custom model uncertainty file containing all of the power meter uncertainty properties. When this command is executed, the model name is automatically set to "CustomFile".

**VB Syntax**
```vba
pwrSensor.UncertaintyFile = value
```

**Variable (Type) - Description**
- `pwrSensor` - A PowerSensorUncertainty (Object)
- `value` - (String) - Custom file name.

**Return Type**
String

**Default**
Not Applicable

**Examples**
```vba
pwrSensor.UncertaintyFile = "C:\U8485A_MY55140018.dat" 'Write

value = pwrSensor.UncertaintyFile 'Read
```

**C++ Syntax**
```cpp
HRESULT put_UncertaintyFile(BSTR *pVal);

HRESULT get_UncertaintyFile(BSTR pVal);
```

**Interface**
IPowerSensorUncertainty
UncertaintyModel Property

Description
Sets and returns the name assigned to a specific power meter model among those available for uncertainty (see UncertaintyModelCatalog).

VB Syntax
pwrSensor.UncertaintyModel = value

Variable
(Type) - Description
pwrSensor - A PowerSensorUncertainty (Object)
value - (String) - Name of power meter model.

Return Type
String

Default
Not Applicable

Examples
pwrSensor.UncertaintyModel = "N8488A" 'Write
value = pwrSensor.UncertaintyModel 'Read

C++ Syntax
HRESULT put_UncertaintyModel(BSTR pVal);

HRESULT get_UncertaintyModel(BSTR *pVal);

Interface
IPowerSensorUncertainty
UncertaintyModelCatalog Property

**Description**
Returns a list of available power meters that have power uncertainty.

**VB Syntax**
```vbnet
pwrMtrs = pwrSensor.UncertaintyModelCatalog
```

**Variable**
- **Type**: Description
- **pwrMtrs**: (Variant) Variable to store the returned power meters.
- **pwrSensor**: A PowerSensorUncertainty (Object)

**Return Type**
Array of strings

**Default**
Not Applicable

**Examples**
```vbnet
pwrMtrs = pwrSensor.UncertaintyModelCatalog
```

**C++ Syntax**
```cpp
HRESULT UncertaintyModelCatalog(VARIANT *pwrMtrs);
```

**Interface**
IPowerSensorUncertainty
**Read-only**

### UnusedChannelNumbers Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns an array of channel numbers that are NOT in use. An unused channel has NO measurements subscribed to it.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chanNumbers = chans.UnusedChannelNumbers (NumberOfChannels)</code></td>
</tr>
<tr>
<td><strong>Variable</strong> (Type) - Description</td>
<td></td>
</tr>
<tr>
<td><code>chanNumbers</code></td>
<td>Variable array to store the returned channel numbers</td>
</tr>
<tr>
<td><code>chans</code></td>
<td>A Channel collection (object)</td>
</tr>
<tr>
<td><strong>NumberOfChannels</strong> (Long Integer)</td>
<td>Number of channels that you are requesting.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>One-dimensional array of long integers. The size of the array is specified by the <code>NumberOfChannels</code> parameter.</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chanNumbers = chans.UnusedChannelNumbers(5)</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_UnusedChannelNumbers(long numberRequested, VARIANT* channelNumbers);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannels2</td>
</tr>
</tbody>
</table>
### USBPowerMeterCatalog Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the ID string of power meters / sensors that are connected to the VNA USB. Use the list to select a power sensor for a source power cal.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><em>list = pwrCal.USBPowerMeterCatalog</em></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><em>list</em></td>
<td><em>(String)</em> Variable to store the returned list of USB power meters.</td>
</tr>
<tr>
<td><em>pwrCal</em></td>
<td><em>(object)</em> – A SourcePowerCalibrator (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Comma-delimited strings. Two sensor strings are separated by a semicolon.</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
| **Examples** | Set *pwrCal = pna.SourcePowerCalibrator*  
*list = pwrCal.USBPowerMeterCatalog*’Read* |
| **C++ Syntax** | HRESULT get_USBPowerMeterCatalog(BSTR *pUSBList); |
| **Interface** | ISourcePowerCalibrator6 |
# UseCalWindow Property - Obsolete

**Read/Write**

<table>
<thead>
<tr>
<th>Description</th>
<th>Replaced with Custom Cal Window commands.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Turns Calibration window ON or OFF during a calibration. <a href="#">Learn more.</a></td>
</tr>
</tbody>
</table>

**VB Syntax**
```
guidedCal.UseCalWindow = value
```

**Variable**
- `(Type) - Description`
- `guidedCal` GuidedCalibration (object)
- `value` (Boolean)

- **True**  Show calibration window
- **False** Hide calibration window

**Return Type**
- **Boolean**

**Default**
- True

**Example**
```
guided.UseCalWindow = True
```

**C++ Syntax**
```
HRESULT get_UseCalWindow(VARIANT_BOOL* val);

HRESULT put_UseCalWindow(VARIANT_BOOL newVal);
```

**Interface**
- IGuidedCalibration
UsedChannelNumbers Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns an array of channel numbers that are in use. A used channel has at least one measurement subscribed to it</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>chanNumbers = chans.UsedChannelNumbers</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>chanNumbers</code></td>
<td>Variable array to store the returned channel numbers</td>
</tr>
<tr>
<td><code>chans</code></td>
<td>A Channel collection (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>One-dimensional array of long integers</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>chanNumbers = chans.UsedChannelNumbers</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_UsedChannelNumbers(VARIANT* channelNumbers);</td>
</tr>
<tr>
<td>Interface</td>
<td>IChannels2</td>
</tr>
</tbody>
</table>
Read/Write

**UseMultipleSensors Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Enable and disable the use of multiple power sensors during a guided calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>guidPwrSensors.UseMultipleSensors = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>guidPwrSensors</code></td>
<td>GuidedCalibrationPowerSensors Collection</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean)</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>Use multiple power sensors</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>Do NOT use multiple power sensors</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>guidedPwrSensors.UseMultipleSensors = True</code></td>
</tr>
<tr>
<td></td>
<td><code>value = guidedPwrSensors.UseMultipleSensors</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_UseMultipleSensors(VARIANT_BOOL* val);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_UseMultipleSensors(VARIANT_BOOL newVal);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IGuidedCalibrationPowerSensors</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Specifies if subsequent power readings will use of the loss table. (PowerLossSegments).</td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pwrCal.UsePowerLossSegments = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong> (Type) - Description</td>
<td><code>pwrCal</code> A <code>SourcePowerCalibrator</code> (object)</td>
</tr>
<tr>
<td></td>
<td>A <code>PowerSensorAsReceiver</code> (Object)</td>
</tr>
<tr>
<td></td>
<td>A <code>PowerSensor</code> (Object)</td>
</tr>
<tr>
<td><strong>value</strong> (boolean)</td>
<td><code>False</code> – Do not use loss table</td>
</tr>
<tr>
<td></td>
<td><code>True</code> – Use loss table</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pwrSens.UsePowerLossSegments = True</code></td>
</tr>
<tr>
<td></td>
<td><code>lossTableState = pwrSens.UsePowerLossSegments</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_UsePowerLossSegments(VARIANT_BOOL bState);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_UsePowerLossSegments(VARIANT_BOOL *bState);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISourcePowerCalibrator</td>
</tr>
<tr>
<td></td>
<td>IPowerSensorAsReceiver</td>
</tr>
<tr>
<td></td>
<td>IPowerSensor</td>
</tr>
</tbody>
</table>
### UsePowerSensorFrequencyLimits Property

**Description**
Specifies if subsequent calls to the `AcquirePowerReadings` method will observe frequency values of the `MinimumFrequency` and `MaximumFrequency` properties.

**VB Syntax**
```
pwrCal.UsePowerSensorFrequencyLimits = value
```

**Variable**
- **Type**: A `SourcePowerCalibrator` (object)
- **Description**

**Value**
- **Boolean**

**False** – Do not use power sensor frequency limits. An acquisition will use just one power sensor for the entire sweep, regardless of frequency.

**True** – Use power sensor frequency limits. A requested acquisition will only succeed for those frequency points which fall between the `MinimumFrequency` and `MaximumFrequency` values of that PowerSensor. An acquisition will pause in mid-sweep if the frequency is about to exceed the `MaximumFrequency` value. When the sweep is paused in this manner, a sensor connected to the other channel input of the power meter can be connected to the measurement port in place of the previous sensor, and then the sweep completed by another call to `AcquirePowerReadings`. However, the `MaximumFrequency` specified for the second sensor would need to be sufficient for the sweep to complete.

**Return Type**
- **Boolean**

**Default**
- **False**

**Examples**
- Set `powerCalibrator = pna.SourcePowerCalibrator`
- `powerCalibrator.UsePowerSensorFrequencyLimits = True`  
  ```vbnet
  'Write
  FreqCheck = powerCalibrator.UsePowerSensorFrequencyLimits  
  'Read
  ```

**C++ Syntax**
```
HRESULT put_UsePowerSensorFrequencyLimits(VARIANT_BOOL bState);
HRESULT get_UsePowerSensorFrequencyLimits(VARIANT_BOOL *bState);
```

**Interface**
- `ISourcePowerCalibrator`
## UserCalsetPrefix Property

**Description**
Sets and returns the prefix to be used when saving User Cal Sets that result from the Cal All session. The Meas Class and channel number are appended to this prefix for each calibrated channel.

Use [GeneratedCalsets](#) to read the saved cal set names.

If a Cal Set prefix is NOT set, the cal data for each channel will be saved only to cal registers.

### VB Syntax

```vbnet
calAll.UserCalsetPrefix = prefix
```

### Variable
- **Type** - Description
- **calAll** - A `ICalibrateAllChannels` object
- **prefix** - (String) User CalSet prefix.

### Return Type
String

### Default
Not Applicable

### Examples

```vbnet
calAll.UserCalsetPrefix = "MyCalSet" 'Set
```

```vbnet
csPrefix = calAll.UserCalsetPrefix 'Returns the CalSet prefix
```

### C++ Syntax

```cpp
HRESULT get_UserCalsetPrefix (BSTR calsetPrefix);

HRESULT put_UserCalsetPrefix (BSTR* calsetPrefix);
```

### Interface
- `ICalibrateAllChannels`
UserRange Property

Description
Assigns the marker to the specified User Range. This restricts the marker's x-axis travel to the User Range span, specified with Start and Stop values.

- Each channel has 16 user ranges.
- Markers and trace statistics can be restricted to any user range.
- More than one marker can occupy a user range.
- User ranges can overlap. For example:
  - User range 1: 3 GHz to 5 GHz
  - User range 2: 4 GHz to 6 GHz

Note: User ranges are especially useful in restricting marker searches to specific areas of the measurement.

VB Syntax
mark.UserRange = value

Variable (Type) - Description
mark (A Marker (object))
value (long integer) - User Range. Choose any number between 0 and 16 (0=Full Span)

Return Type
Long Integer

Default
0 - Full Span

Examples
mark.UserRange = 1 'Write
UserRange = mark.UserRange 'Read

C++ Syntax
HRESULT get_UserRange(long *pRangeNumber)
HRESULT put_UserRange(long lRangeNumber)

Interface
IMarker
### UserRangeMax Property

**Description**

Note: This property on the Channel Object is superseded by the same property on the Measurement Object.

Sets the stimulus stop value for the specified User Range.

**VB Syntax**

```vbnet
chan=UserRangeMax(domainType,Rnum) = value - Superseded
meas.UserRangeMax(Rnum) = value
mark.UserRangeMax(Rnum) = value
```

**Variable (Type) - Description**

- `chan` A Channel (object) - Superseded
- `meas` A Measurement (object)
- `mark` A Marker (object)

To assign a marker to a User Range, use the UserRange Property.

**domainType**

This argument is no longer required. The domain type is inferred by the measurement or marker.

- (enum NADomainType) - Choose from:
  - 0 - naDomainFrequency
  - 1 - naDomainTime
  - 2 - naDomainPower

**Rnum (long integer)** - User Range number. Choose any number between 1 and 16 (0=Full Span)

**value (double)** - Stop value. Choose any number within the full span of the channel.

**Return Type**

Double

**Default**

The current stimulus setting for the channel

- `mark.UserRangeMax(1) = 3e9` 'Write
- `meas.UserRangeMax(1) = 3e9` 'Write
- `UseRngeMax = mark.UserRangeMax(1)` 'Read
- `UseRngeMax = meas.UserRangeMax(2)` 'Read
C++ Syntax

HRESULT put_UserRangeMax(long rangeNumber, double maxValue)

HRESULT get_UserRangeMax(long rangeNumber, double *maxValue)

Interface

IMeasurement

IMarker
## UserRangeMin Property

**Description**  
Sets the stimulus start value for the specified User Range.  
This property uses different arguments for the channel and marker objects.

**VB Syntax**  
```vb  
chan.UserRangeMin(domainType, range) = value  
```

or

```vb  
mark.UserRangeMin(range) = value  
```

**Variable**  

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chan</td>
<td>A Channel (object)</td>
</tr>
<tr>
<td>mark</td>
<td>A Marker (object)</td>
</tr>
</tbody>
</table>

To assign a marker to a User Range, use the UserRange Property.

**domainType**  

*(enum NADomainType)*  
Type of sweep currently implemented on the channel - Choose from:

- 0 - naDomainFrequency
- 1 - naDomainTime
- 2 - naDomainPower
- 3 - naDomainPhase

**range**  
*(long)* - User Range number. Choose any number between 1 and 16 (0=Full Span)

**value**  
*(double)* - Start value. Choose any number within the full span of the analyzer

**Return Type**  
Double

**Default**  
The current stimulus setting for the channel

**Examples**

```plaintext  
mark.UserRangeMin(1) = 3e9 'Write  
chan.UserRangeMin(naDomainFrequency, 1) = 3e9 'Write  

UseRngeMin = mark.UserRangeMin 'Read  
UseRngeMin = chan.UserRangeMin 'Read  
```
**C++ Syntax**

HRESULT put_UserRangeMin(tagNADomainType domain, long rangeNumber, double minValue)

HRESULT get_UserRangeMin(tagNADomainType domain, long rangeNumber, double *minValue)

**Interface**

IChannel
### UserDescriptionOfPNA Property

**Description**  
Sets and reads a description of the VNA used to perform the User Characterization. This description is stored with the characterization in the ECal module.

Set this description before sending `Initialize` or the default (empty string) will be used.

**VB Syntax**  
`userChar.UserDescriptionOfPNA = value`

**Variable**  
*(Type)* - Description

*userChar*  
An `IECalUserCharacterizer` Object

*value*  
(String) Descriptive text, limited to 14 characters maximum.

**Return Type**  
String

**Default**  
"" (Empty String)

**Examples**  
`userChar.UserDescriptionOfPNA = "My PNA"`

**C++ Syntax**  
HRESULT get_UserDescriptionOfPNA(BSTR *info);

HRESULT put_UserDescriptionOfPNA(BSTR info);

**Interface**  
IECalUserCharacterizer
**UserName Property**

**Description**
Sets and reads the description of the person and/or company who is producing the ECal user characterization. This description is stored with the characterization in the ECal module.

Set this description before sending `Initialize` or the default (empty string) will be used.

**VB Syntax**

```
userChar.UserName = value
```

**Variable**

*userChar* - An `ECalUserCharacterizer` Object  
value - (String) Descriptive text, limited to 19 characters maximum.

**Return Type**

String

**Default**

"" (Empty String)

**Examples**

```
userChar UserName = "John Doe, Acme Inc."
```

**C++ Syntax**

```cpp
HRESULT get_UserName(BSTR *name);

HRESULT put_UserName(BSTR name);
```

**Interface**

`IECalUserCharacterizer`
Write/Read

About User Preset

UserPresetEnable Property

Description
'Checks' and 'clears' the enable box on the User Preset dialog box. This only affects subsequent Presets from the front panel user interface.

Regardless of the state of the User Preset Enable checkbox, the app.Preset command will always preset the VNA to the factory preset settings, and app.UserPreset will always perform a User Preset.

VB Syntax

```vbnet
app.UserPresetEnable = state
```

Variable (Type) - Description

- **app**: An Application (object)
- **state**: (boolean) Front Panel User Preset State. Choose from:

  - False – User Preset OFF
  - True – User Preset ON

Return Type

Boolean

Default

False

Examples

```vbnet
app.UserPresetEnable = True  'Write
upreset = app.UserPresetEnable  'Read
```

C++ Syntax

```cpp
HRESULT get_UserPresetEnable(VARIANT_BOOL *pVal)
HRESULT put_UserPresetEnable(VARIANT_BOOL newVal)
```

Interface

IApplication6
### Valid Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a boolean value to indicate if the current equation on the measurement is valid. For equation processing to occur, the equation must be valid and ON.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>IsValid = eq.Valid</code></td>
</tr>
</tbody>
</table>
| **Variable** | *(Type)* - Description  
*good* *(Boolean)* Variable to store the returned value.  
**True** (1) - equation is valid  
**False** (0) - equation is NOT valid  
*eq* MeasurementEquation *(object)* |
| **Return Type** | Boolean |
| **Default** | Not Applicable |
| **Examples** | `IsValid = eq.Valid` 'Read' |
| **C++ Syntax** | `HRESULT get_Valid(Boolean *equation)` |
| **Interface** | IMeasurementEquation |


ValidConnectorTypes Property

**Description**
Returns a list of all connector types for which there are calibration kits. Looks for connector types in mechanical cal kits, within VNA disk memory, and within the attached ECal Module memory.

Here are the more common connector types:

<table>
<thead>
<tr>
<th>Type of Connector</th>
<th>Type</th>
<th>Female/Male</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W-band waveguide</td>
<td>Type B</td>
<td>1.00 mm female</td>
<td></td>
</tr>
<tr>
<td>V-band waveguide</td>
<td>Type A (50) female</td>
<td>1.00 mm male</td>
<td></td>
</tr>
<tr>
<td>U-band waveguide</td>
<td>Type A (50) male</td>
<td>1.85 mm male</td>
<td></td>
</tr>
<tr>
<td>R-band waveguide</td>
<td>Type F (75) female</td>
<td>1.85 mm female</td>
<td></td>
</tr>
<tr>
<td>Q-band waveguide</td>
<td>Type F (75) male</td>
<td>2.92 mm female</td>
<td></td>
</tr>
<tr>
<td>K-band waveguide</td>
<td>Type N (75) female</td>
<td>2.92 mm male</td>
<td></td>
</tr>
<tr>
<td>P-band waveguide</td>
<td>Type N (75) male</td>
<td>APC 2.4 female</td>
<td></td>
</tr>
<tr>
<td>X-band waveguide</td>
<td>Type N (50) female</td>
<td>APC 2.4 male</td>
<td></td>
</tr>
<tr>
<td>7-16 female</td>
<td>Type N (50) male</td>
<td>APC 3.5 female</td>
<td></td>
</tr>
<tr>
<td>7-16 male</td>
<td>Type N (50) male</td>
<td>APC 3.5 male</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>APC 7</td>
<td></td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
value = obj.ValidConnectorTypes
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(Variant) List of connector types</td>
</tr>
<tr>
<td>obj</td>
<td>Any of the following:</td>
</tr>
<tr>
<td></td>
<td>ECalUserCharacterizer (object)</td>
</tr>
<tr>
<td></td>
<td>GuidedCalibration (object)</td>
</tr>
<tr>
<td></td>
<td>SMCType (object)</td>
</tr>
<tr>
<td></td>
<td>VMCType (object)</td>
</tr>
</tbody>
</table>

**Return Type**

Variant
<table>
<thead>
<tr>
<th>Default</th>
<th>Not Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td><code>value = SMC.ValidConnectorTypes</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_ValidConnectorTypes(VARIANT* connectorTypes);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IGuidedCalibration</td>
</tr>
<tr>
<td></td>
<td>SMCType</td>
</tr>
<tr>
<td></td>
<td>VMCType</td>
</tr>
</tbody>
</table>
ValidPorts Property

Description Queries available ports for independent power calibration.

VB Syntax

```
value = IndependentPwrrCal.ValidPorts
```

Variable (Type) - Description

```
value (Variant) - Variable to store the returned ports.
```

IndependentPwrrCal A IndependentPowerCalibration (object)

Return Type Variant

Default Not Applicable

Example

```
value = CalibrateAllChannels.IndependentPowerCalibration.ValidPorts
'Read
```

C++ Syntax

```
HRESULT ValidPorts(VARIANT* ports);
```

Interface IIndependentPowerCalibration
Value Property

Description
Write or read a value (setting) for the current element.

This command is used to set both RF and IF Path Configuration.

- See RF Path Configuration (elements, value)
- See IF Path Configuration (elements, value)

VB Syntax

```
pathElement.Value = value
```

Variable

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pathElement</td>
<td>A PathElement (object)</td>
</tr>
<tr>
<td>value</td>
<td>(String) Value for the element. Use pathElement.Values to return a list of valid settings for this element.</td>
</tr>
</tbody>
</table>

Return Type
String

Default
Not Applicable

Examples
See Examples:

- IFPathConfiguration Setup
- RF PathConfiguration Example

C++ Syntax

```
HRESULT get_Value( BSTR* pValue );

HRESULT put_Value( BSTR value );
```

Interface
IPathElement
## Values Property

**Description**
Returns an array of valid settings that can be used with the element object.

See a list of configurable elements and settings for various VNA models.

**VB Syntax**
```
values = pathElement.Values
```

**Variable**

<table>
<thead>
<tr>
<th>(Type) - Description</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Variant array)</td>
<td>Variable to store the array of valid settings for the element.</td>
</tr>
</tbody>
</table>

**pathElement**
A PathElement (object)

**Return Type**
Variant array

**Default**
Not Applicable

**Examples**
```
settings=pathElement.Values
```

**C++ Syntax**
```
HRESULT Values(VARIANT* pValues);
```

**Interface**
IPathElement
About Vector Averaging

VectorAverageEnable Property

**Description**
Set and read the ON/OFF state of the vector averaging.

**VB Syntax**

```vbnet
sa.coherence.VectorAverageEnable = value
```

**Variable (Type) - Description**

- **sa.coherence** → A SpectrumAnalyzerCoherence (object)
- **value** → (Boolean) Choose from:
  - **0** - OFF - Vector averaging is set to OFF.
  - **1** - ON - Vector averaging is set to ON.

Learn about these settings.

**Return Type**
Boolean

**Default**
0

**Examples**

```vbnet
sa.coherence.VectorAverageEnable = OFF 'Write
value = sa.coherence.VectorAverageEnable 'Read
```

**C++ Syntax**

```cpp
HRESULT put_VectorAverageEnable(VARIANT_BOOL bEnable);
HRESULT get_VectorAverageEnable(VARIANT_BOOL* bEnable);
```

**Interface**
ICoherenceSA
## VectorAverageValue Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the vector averaging value.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>sa.coherence.VectorAverageValue = value</code></td>
</tr>
<tr>
<td><strong>Variable (Type)</strong> - Description</td>
<td></td>
</tr>
<tr>
<td><code>sa.coherence</code></td>
<td>A <code>SpectrumAnalyzerCoherence</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Choose a value between 0 and 65536.</td>
</tr>
</tbody>
</table>

Learn about these settings.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Double</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>1 (no averaging)</td>
</tr>
</tbody>
</table>

### Examples

```vbnet
sa.coherence.VectorAverageValue = 1 'Write
value = sa.coherence.VectorAverageValue 'Read
```

See an example program.

### C++ Syntax

```cpp
HRESULT put_VectorAverageValue(double val);
HRESULT get_VectorAverageValue(double* val);
```

### Interface

ICoherenceSA
## VectorAverageValueMax Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the current maximum available vector averaging value.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = sa.coherence.VectorAverageValueMax</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>sa.coherence</code></td>
<td>A <code>SpectrumAnalyzerCoherence</code> <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long)</em> Variable to store the returned value.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>value = sa.coherence.VectorAverageValueMax</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_VectorAverageValueMax(long* val);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>ICoherenceSA</code></td>
</tr>
</tbody>
</table>
VelocityFactor Property

Description
Sets the velocity factor to be used with Electrical Delay, Port Extensions, and Time Domain marker distance calculations.

VB Syntax
app.VelocityFactor = value

Variable (Type) - Description

app   An Application (object)
value (double) - Velocity factor. Choose a number between: 0 and 10 (.66 polyethylene dielectric; .7 PTFE dielectric)

Return Type
Double

Default
1

Examples
app.VelocityFactor = .66 'Write
RelVel = app.VelocityFactor 'Read

C++ Syntax
HRESULT get_VelocityFactor(double *pVal)
HRESULT put_VelocityFactor(double newVal)

Interface
IApplication
### VideoAveragingCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Reads the number of video bandwidth sweeps that are averaged together. This readout is displayed on the SA setup page.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>value = sa.VideoAveragingCount</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(Long) Variable to store the returned averaging count.</td>
</tr>
<tr>
<td>sa</td>
<td>A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>N/A</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>value = sa.VideoAveragingCount 'Read</td>
</tr>
</tbody>
</table>

See an example program.

**C++ Syntax**

```cpp
HRESULT get_VideoAveragingCount(long* count);
```

**Interface**

ISpectrumAnalyzer
VideoAveragingType Property

Description
Set and read the video averaging type.

VB Syntax

\[
sa.VideoAveragingType = value
\]

Variable (Type) - Description

\[
sa \quad \text{A SpectrumAnalyzer (object)}
\]

\[
value \quad \text{(Enum as NASAVideoAveragingType) Choose from:}
\]

0 - naPower
1 - naLog
2 - naVoltage
3 - naVoltageMax
4 - naVoltageMin

Learn about these settings.

Return Type
Enum

Default
0 - naPower

Examples

\[
\begin{align*}
\text{sa.VideoAveragingType} & = \text{naLog} \quad \text{\texttt{\textcolor{teal}{\textbf{\textit{Write}}}}} \\
\text{value} & = \text{sa.VideoAveragingType} \quad \text{\texttt{\textcolor{teal}{\textbf{\textit{Read}}}}}
\end{align*}
\]

See an example program.

C++ Syntax

HRESULT put_VideoAveragingType(tagNASAVideoAveragingType type);
HRESULT get_VideoAveragingType(tagNASAVideoAveragingType* type);

Interface
ISpectrumAnalyzer
### VideoBWMax

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the maximum video bandwidth.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = sa.VideoBWMax</code></td>
</tr>
<tr>
<td>Variable</td>
<td><code>sa</code> - A SpectrumAnalyzer (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td></td>
<td><code>value = sa.VideoBWMax</code></td>
</tr>
</tbody>
</table>

See an example program.

**C++ Syntax**

```
HRESULT get_VideoBWMax(tagNASAVideoBWMax* maxbw);
```

**Interface**

I SpectrumAnalyzer
**VideoBWMin**

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the minimum video bandwidth.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = sa.VideoBWMin</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td></td>
<td><code>sa</code> A SpectrumAnalyzer <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td></td>
<td><code>value = sa.VideoBWMin 'Read</code></td>
</tr>
</tbody>
</table>

See an example program.

**C++ Syntax**

```cpp
HRESULT get_VideoBWMin(tagNASAVideoBWMin* minbw);
```

**Interface**

`ISpectrumAnalyzer`
## VideoBW Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the Video bandwidth. Also set VideoBWMODE to naManual.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>\texttt{sa.VideoBW = value}</td>
</tr>
<tr>
<td>Variable</td>
<td>\texttt{sa} \quad \textbf{A SpectrumAnalyzer (object)}</td>
</tr>
<tr>
<td>\textit{value}</td>
<td>\textbf{(Double)} \quad \text{Choose a value between 3 Hz and 3 MHz. Going outside this range places the trace into a hold mode.}</td>
</tr>
<tr>
<td>Learn about these settings.</td>
<td></td>
</tr>
<tr>
<td>Return Type</td>
<td>\textbf{Double}</td>
</tr>
<tr>
<td>Default</td>
<td>100 kHz</td>
</tr>
<tr>
<td>Examples</td>
<td>\texttt{sa.VideoBW = 1e5} \quad \textbf{Write}</td>
</tr>
<tr>
<td></td>
<td>\texttt{value = sa.VideoBW} \quad \textbf{Read}</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>\texttt{HRESULT put_VideoBW(double freq);}</td>
</tr>
<tr>
<td></td>
<td>\texttt{HRESULT get_VideoBW(double* freq);}</td>
</tr>
<tr>
<td>Interface</td>
<td>\texttt{ISpectrumAnalyzer}</td>
</tr>
</tbody>
</table>

See an example program.
## VideoBWMode Property

**Description**  
Set and read how the video bandwidth is set. When ON, video bandwidth is set based on RBW/VBW ratio.

**VB Syntax**  
`sa.VideoBWMode = value`

**Variable**  
- **(Type)** - Description
  - `sa` (Type) - A SpectrumAnalyzer (object)
  - `value` (Enum as NAModes) Choose from:
    - **1 - naMANUAL** - Video BW is set manually using VideoBW Property
    - **0 - naAUTO** - Video BW is set automatically.

Learn about these settings.

**Return Type**  
Enum

**Default**  
0 - naAUTO

**Examples**  
- `sa.VideoBWMode = naMANUAL`  Write
- `value = sa.VideoBWMode`  Read

See an example program.

**C++ Syntax**  
- `HRESULT put_VideoBWMode(tagNAModes mode);`
- `HRESULT get_VideoBWMode(tagNAModes* mode);`

**Interface**  
ISpectrumAnalyzer
## View Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets (or returns) the type of trace displayed on the screen.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.View = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A measurement (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum NAView) - Type of trace. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naData</td>
</tr>
<tr>
<td></td>
<td>1 - naDataAndMemory</td>
</tr>
<tr>
<td></td>
<td>2 - naMemory</td>
</tr>
<tr>
<td></td>
<td>3 - naNoTrace</td>
</tr>
</tbody>
</table>

**Note:** The **naData** trace may reflect the result of a **TraceMath** operation.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>NAView</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>naData</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meas.View = naData</code> <em>Write</em></td>
</tr>
<tr>
<td></td>
<td><code>trceview = meas.View</code> <em>Read</em></td>
</tr>
</tbody>
</table>

| **C++ Syntax** | HRESULT `get_View(tagNAView* pView)` |
| | HRESULT `put_View(tagNAView newView)` |
| **Interface** | IMeasurement |
### Visible Property

**Description**  
Makes the Network Analyzer application visible or not visible. In the Not Visible state, the analyzer cycle time for making measurements can be significantly faster because the display does not process data.

**VB Syntax**  
```
app.Visible = state
```

**Variable**  
 *(Type) - Description*

- **app**  
  An Application *(object)*

- **state**  
  *(boolean)*

  - **False** - Network Analyzer application NOT visible
  - **True** - Network Analyzer application IS visible

**Return Type**  
Boolean

**Default**  
True

**Examples**
```
app.Visible = False 'Write
vis = app.Visible 'Read
```

**C++ Syntax**  
```
HRESULT get_Visible(VARIANT_BOOL * bVisible)
HRESULT put_Visible(VARIANT_BOOL bVisible)
```

**Interface**  
IApplication
VoltageLimit Property

Description
Sets and returns the maximum output voltage value of the external DC Source. This command supports Keysight B2900A and N6700 series devices only.

VB Syntax
`extDC.VoltageLimit (devicename) = value`

Variable (Type) - Description
- `extDC` (A ExternalDCDevice (object))
- `devicename` (String) Name of the device.
- `value` (Double) Voltage limit value.

Return Type
Double

Default
0

Examples
- `extDC.VoltageLimit("myDCDevice") = 4` 'Write
- `limit = extDC.VoltageLimit("myDCDevice")` 'Read

C++ Syntax
- `HRESULT get_VoltageLimit(BSTR devicename, double *vLimit)`
- `HRESULT put_VoltageLimit(BSTR devicename, double newLimit)`

Interface
IEternalDCDevice2
### vsa.ADCDither Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the dither state.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>vsa.ADCDither = state</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>vsa</code></td>
<td>A <strong>VSA (object)</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(Boolean)</strong> Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Turn dithering off.</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Turn dithering on.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>True</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>vsa.ADCDither = False</code> <em>Write</em></td>
</tr>
<tr>
<td></td>
<td><code>value = vsa.ADCDither</code> <em>Read</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_Dithering(VARIANT_BOOL bEnable);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_Dithering(VARIANT_BOOL* bEnable);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IVSA</td>
</tr>
</tbody>
</table>
vsa.ADCFilter Property

**Description**
Sets and reads the ADC filter width. The entered frequency value is rounded to the closest value supported by the VNA (11 MHz or 38 MHz).

**VB Syntax**
vsa.ADCFilter = mode

**Variable**
(Type) - Description
vsa (object) - A VSA object
value (Double) Choose 11 MHz or 38 MHz.

Learn about these settings.

**Return Type**
Double

**Default**
38 MHz

**Examples**
vsa.ADCFilter = 11e6  'Write
value = vsa.ADCFilter  'Read

**C++ Syntax**
HRESULT get_ADCFilter(double* cutfreq);
HRESULT put_ADCFilter(double cutfreq);

**Interface**
IVSA
### vsa.ADCSampleRate Property

**Description**  
Sets or returns the ADC sampling frequency setting. The entered frequency is rounded to the closest value supported by the VNA (25 MHz or 100 MHz).

**VB Syntax**  
```vb
vsa.ADCSampleRate = value
```

**Variable (Type) - Description**
- **vsa** A VSA (object)
- **value** (Double) Choose from 100 MHz or 25 MHz.

Learn about these settings.

**Return Type**  
Double

**Default**  
100 MHz

**Examples**
- `vsa.ADCSampleRate = 100MHz`  
  `value = vsa.ADCSampleRate`  

**C++ Syntax**
- `HRESULT get_ADCSampleRate(double* rate)`
- `HRESULT put_ADCSampleRate(double rate)`

**Interface**  
IVSA
### vsa.ADCStacking Property

**Description**
Sets or returns the ADC stack range. If unspecified, value is set to 0 (no stacking).

**VB Syntax**
```vba
vsa.ADCStacking = value
```

**Variable (Type) - Description**

- `vsa` A VSA (object)
- `value` (Long) - Value of stacking number.

Learn about these settings.

**Return Type**
Long

**Default**
0

**Examples**
```
vsa.ADCStacking = 100  'Write
value = vsa.ADCStacking  'Read
```

**C++ Syntax**
```
HRESULT get_ADCStacking(double *StackVal)
HRESULT put_ADCStacking(double StackVal)
```

**Interface**
IVSA

---

---
# vsa.CenterFrequency Property

Sets or returns the VSA center frequency which determines the LO Freq (LO Freq = VSA Center Freq ± IF Offset) and the center frequency of the VNA and VSA.

## VB Syntax

```vbnet
vsa.CenterFrequency = value
```

## Variable (Type) - Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vsa</td>
<td>VSA (object)</td>
<td></td>
</tr>
<tr>
<td>value</td>
<td>double</td>
<td>Center frequency in Hertz. Choose any number between the minimum and maximum frequencies of the analyzer.</td>
</tr>
</tbody>
</table>

Learn about these settings.

## Return Type

Double

## Default

3.021e9

## Examples

```vbnet
vsa.CenterFrequency = 3.024e9 'sets the center frequency - Write
value = vsa.CenterFrequency 'Read
```

## C++ Syntax

```cpp
HRESULT get_CenterFrequency(double *pVal)
HRESULT put_CenterFrequency(double newVal)
```

## Interface

IVSA
## vsa.IFOffset Property

Sets or returns the IF offset used to calculate the LO frequency.

### VB Syntax

```vbnet
vsa.IFOffset = value
```

### Variable (Type) - Description

- **vsa**: A **VSA** (object)
- **value**: A **double** - IF offset value in Hertz. The range is 0 Hz to 21 MHz.

Learn about these settings.

### Return Type

Double

### Default

21e6

### Examples

```vbnet
vsa.IFOffset = 20e6  'Write
value = vsa.IFOffset 'Read
```

### C++ Syntax

```cpp
HRESULT get_IFOffset(double *pVal)
HRESULT put_IFOffset(double newVal)
```

### Interface

IVSA
vsa.LOSide Property

Description
Sets and reads the LO side.

VB Syntax
vsa.LOSide = mode

Variable
(Type) - Description
vsa (object) - A VSA object
value (Enum as NALevel)

Choose from:
0 - naLow
1 - naHigh

Learn about these settings.

Return Type
Enum

Default
Low

Examples
vsa.LOSide = naHigh 'Write

value = vsa.LOSide 'Read

C++ Syntax
HRESULT get_LOSide(tagNALevel* pVal);

HRESULT put_LOSide(tagNALevel pVal);

Interface
IVSA
### vsa.StreamDataToFile Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Set and read the save data state.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>vsa.StreamDataToFile = state</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td></td>
<td><code>vsa</code> A VSA (object)</td>
</tr>
<tr>
<td></td>
<td><code>value</code> (Boolean) Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Do not allow saving data on disk.</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Allow saving data on disk.</td>
</tr>
<tr>
<td><strong>Learn about these settings.</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>vsa.StreamDataToFile = True 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = vsa.StreamDataToFile 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_StreamDataToFile(VARIANT_BOOL bEnable);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_StreamDataToFile(VARIANT_BOOL* bEnable);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IVSA</td>
</tr>
</tbody>
</table>
### vsa.VSAConnection Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the VSA connection state.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>vsa.VSAConnection = state</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td></td>
<td><code>vsa</code> A VSA (object)</td>
</tr>
<tr>
<td></td>
<td><code>value</code> (Boolean) Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Disconnect from VSA.</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Connect to VSA.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
<tr>
<td>Examples</td>
<td><code>set app = CreateObject(&quot;AgilentPNA835x.Application&quot;)</code></td>
</tr>
<tr>
<td></td>
<td><code>app.CreateCustomMeasurementEx 2, &quot;VSA&quot;, &quot;Port 2(B)&quot;</code></td>
</tr>
<tr>
<td></td>
<td><code>set vsa = app.ActiveChannel.CustomChannelConfiguration</code></td>
</tr>
<tr>
<td></td>
<td><code>vsa.VSAConnection = True</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>value = vsa.VSAConnection</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT put_VSAConnection(VARIANT_BOOL bEnable);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_VSAConnection(VARIANT_BOOL* bEnable);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IVSA</td>
</tr>
</tbody>
</table>
### WGCutoffFreq Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the value of the waveguide cut off frequency.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.WGCutoffFreq = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(double)</em> - Frequency in Hertz.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>Print meas.WGCutoffFreq  'prints the value of the waveguide cut off frequency</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_WGCutoffFreq(double *pVal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_WGCutoffFreq(double newVal);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
## WideBandDectionState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the pulse mode detection method.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pulseMeas.WideBandDectionState = bool</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>pulseMeas</code></td>
<td>A <code>PulseMeasurementControl</code> (object)</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><strong>False</strong> - Narrowband mode. <strong>True</strong> - Wideband mode</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Based on pulse width.</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pulse.WideBandDectionState = True</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>value = pulse.WideBandDectionState</code> 'Read'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_WideBandDectionState(VARIANT_BOOL *pVal);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_WideBandDectionState(VARIANT_BOOL newVal);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IPulseMeasurementControl</code></td>
</tr>
</tbody>
</table>
# Write/Read

## About PNA-X Pulsed Capabilities

### Width Property

**Description**
Sets the pulse width - the amount of time that the pulse is ON.

**VB Syntax**

\[
pulse.Width(n) = value
\]

**Variable**

- **pulse** (Type) - Description
  A PulseGenerator (object)
  
- **n** (Integer) Pulse generator number. Choose from 0 to 4.
  
  0 is the generator that pulses the ADC.

- **value** (Double) Pulse width in seconds. Choose a value from about 33ns to about 70 seconds.

**Return Type**
Double

**Default**
1e-4 sec

**Examples**

- `pulse.Width = 1ms` 'Write
- `value = pulse.Width` 'Read

**C++ Syntax**

- `HRESULT get_Width(integer pulse, double* width);`
- `HRESULT put_Width(integer pulse, double width);`

**Interface**
IPulseGenerator
## WindowNumber Property

### Description
Returns the window number. You might use this property to identify a particular window so that you can create a new Measurement in that window.

### VB Syntax
```
value = win.WindowNumber
```

### Variable
- **`win`** *(Type) - Description*
  - A NAWindow (object)
- **`value`** *(long integer) - Variable to store the returned window number*

### Return Type
Long Integer

### Default
Not Applicable

### Examples
```
value = app.ActiveNAWindow.WindowNumber
```

### C++ Syntax
```
HRESULT (long* windowNumber);
```

### Interface
INAWindow
About Arranging Windows

**WindowState Property**

**Description**
Sets or returns the window setting of Maximized, Minimized, or Normal. To arrange all of the windows, use `app.ArrangeWindows`.

**VB Syntax**
```vbnet
object.WindowState = value
```

**Variable**
- **(Type)** - Description
  - `object` - An Application (object) - main window or
    - A NaWindow (object) - data windows
  - `value` - (enum NAWindowStates) - The window state. Choose from:
    - 0 - naMinimized - Minimizes the window to an Icon on the lower toolbar
    - 1 - naMaximized - Maximizes the window
    - 2 - naNormal - changes the window size to the user defined setting (between Max and Min).

**Return Type**
Long Integer

**Default**
naMaximized

**Examples**
```vbnet
app.WindowState = naMinimized 'changes the Network Analyzer application window to an icon. -Write
win.WindowState = naNormal 'changes the window defined by the win object variable to user defined settings. -Write
winstate = app.WindowState 'Read
```

**C++ Syntax**
```cpp
HRESULT get_WindowState(tagNAWindowStates *pVal)
HRESULT put_WindowState(tagNAWindowStates newVal)
```

**Interface**
INAWindow
IApplication
### XAxisAnnotation Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the X-Axis annotation of the specified tuning sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = embedLODiag.XAxisAnnotation(n)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(String)</strong> Variable to store the returned data.</td>
</tr>
<tr>
<td><code>embedLODiag</code></td>
<td>An EmbeddedLODiagnostic <strong>(object)</strong></td>
</tr>
<tr>
<td><code>n</code></td>
<td>(Long) Tuning sweep number. Use <code>NumberOfSweeps</code> to find the number of sweeps taken.</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>data = embedLO.XAxisAnnotation 3 'read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT XAxisAnnotation(long sweep, BSTR* annotation);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IEmbededLODiagnostic</td>
</tr>
</tbody>
</table>
# XAxisDomain Property

**Description**
Sets and returns the X-Axis domain of the selected DIQ measurement.

**VB Syntax**
```vbnet
Meas.XAxisDomain = value
```

**Variable**
- **(Type)** - Description
  - `Meas` a Measurement (object)
  - `value` String - Domain that is displayed on the X-axis.

<table>
<thead>
<tr>
<th>Choose one of these:</th>
<th>Then set X-Axis Source (XAxis Property) using one of these as the argument.</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Frequency&quot;</td>
<td>&quot;F1&quot;, &quot;F2&quot;, etc.</td>
</tr>
<tr>
<td>&quot;Power&quot;</td>
<td>Source port: &quot;Port 1&quot;, &quot;Port 2&quot;, etc.</td>
</tr>
<tr>
<td>&quot;Phase&quot;</td>
<td>Source port: &quot;Port 1&quot;, &quot;Port 2&quot;, etc.</td>
</tr>
<tr>
<td>&quot;DC&quot;</td>
<td>DC Source: &quot;AO1&quot;, &quot;AO2&quot;</td>
</tr>
<tr>
<td>&quot;Points&quot;</td>
<td>&quot;Points&quot;</td>
</tr>
</tbody>
</table>

**Example**
1. `Meas.XAxisDomain = "Frequency"
2. `Meas.XAxis = "F2"

```csharp
domain = Meas.XAxisDomain 'Read
```

**C++ Syntax**
```csharp
HRESULT put_XAxisDomain(BSTR domain);
HRESULT get_XAxisDomain(BSTR *domain);
```

**Interface**
IMeasurement17
**XAxisPointSpacing Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets X-axis Point Spacing for the display traces measured with segment sweeps on the active channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan.XAxisPointSpacing = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>A Channel (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Enum as naStates) - Choose from:</td>
</tr>
<tr>
<td>0 - naOFF</td>
<td>Turns X-axis Point Spacing OFF</td>
</tr>
<tr>
<td>1 - naON</td>
<td>Turns X-axis Point Spacing ON</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0 - naOFF</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chan.XAxisPointSpacing = naOFF</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>xspac = chan.XAxisPointSpacing</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_XAxisPointSpacing (tagNAStates *pState);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_XAxisPointSpacing (tagNAStates newState);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel2</td>
</tr>
</tbody>
</table>
# XAxis Property

**Description**  
Sets the X-axis of the selected measurement to a DC Source. This command does not change the default setting for new traces.

**VB Syntax**  
```vb  
Meas.XAxisSource = value  
```

**Variable** (Type) - Description

- **Meas**  
  A Measurement (object)

- **value**  
  (String) - Not case-sensitive. For all channels EXCEPT DIQ, choose from the following:

  - "Default" - The default X-axis setting for the selected measurement. For Application measurements, the X-Axis domain is set with specific commands.
  - "AO1" - Internal DC source #1
  - "AO2" - Internal DC source #2

**Note:** For DIQ channels, see XAxisDomain Property

**Example**  
```vb  
Meas.XAxis = "Default" 'Write  
value = Meas.XAxis 'Read  
```

**C++ Syntax**  
```c++  
HRESULT put_XAxis(BSTR source);  
HRRESULT get_XAxis(BSTR* source);  
```

**Interface**  
IMeasurement17
## XAxisStart Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the X-Axis start value of the specified tuning sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( \text{value} = \text{embedLODiag.XAxisStart}(n) )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>( \text{value} )</td>
<td>(Double) Variable to store the returned data.</td>
</tr>
<tr>
<td>( \text{embedLODiag} )</td>
<td>An EmbeddedLODiagnostic (object)</td>
</tr>
<tr>
<td>( n )</td>
<td>(Long) Tuning sweep number. Use \text{NumberOfSweeps} to find the number of sweeps taken.</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>\text{data}= \text{embedLO.XAxisStart 3 'read}</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT XAxisStart (long sweep, double* start);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IEmbededLODiagnostic</td>
</tr>
</tbody>
</table>
### XAxisStop Property

**Description**  
Returns the X-Axis stop value of the specified tuning sweep.

**VB Syntax**  
```
value = embedLODiag.XAxisStop (n)
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Double</td>
<td>Variable to store the returned data.</td>
</tr>
</tbody>
</table>

**embedLODiag**  
An EmbeddedLODiagnostic (object)

**n** (Long) Tuning sweep number. Use NumberOfSweeps to find the number of sweeps taken.

**Default**  
Not Applicable

**Examples**  
```
data= embedLO.XAxisStop 3 'read
```

**C++ Syntax**  
```
HRESULT XAxisStop (long sweep, double* start);
```

**Interface**  
IEmbededLODiagnostic
# YAxisAnnotation Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Y-Axis annotation of the specified tuning sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = embedLODiag.YAxisAnnotation(n)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(String)</strong> Variable to store the returned data.</td>
</tr>
<tr>
<td><code>embedLODiag</code></td>
<td>An EmbeddedLODiagnostic <strong>(object)</strong></td>
</tr>
<tr>
<td><code>n</code></td>
<td>(Long) Tuning sweep number. Use <code>NumberOfSweeps</code> to find the number of sweeps taken.</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>data= embedLO.YAxisAnnotation 3 'read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT YAxisAnnotation(long sweep, BSTR* annotation);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IEmbededLODiagnostic</td>
</tr>
</tbody>
</table>
YScale Property

Description: Sets or returns the Y-axis Per-Division value of the active trace.

VB Syntax: `trace.YScale = value`

Variable (Type) - Description:
- `trace` A Trace (object)
- `value` (double) - Scale /division number. Units and range depend on the current data format.

Return Type: Double

Default: 10 (db)

Examples:
- `trac.YScale = 5` 'Write
- `yscl = trac.YScale` 'Read

C++ Syntax:
- `HRESULT get_YScale(double *pVal)`
- `HRESULT put_YScale(double newVal)`

Interface: ITrace
**Z0 Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Returns the characteristic impedance for the calibration standard.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>calstd.Z0 = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(single) - Impedance in Ohms</td>
</tr>
<tr>
<td>Return Type</td>
<td>Single</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>calstd.Z0 = 50</code> <em>'Write</em></td>
</tr>
<tr>
<td></td>
<td><code>impedance = calstd.Z0</code> <em>'Read</em></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_Z0(float *pVal)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_Z0(float newVal)</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ICalStandard</td>
</tr>
</tbody>
</table>
## Abort Method

**Description**
Ends the current measurement sweep on the channel.

**VB Syntax**
`chan.Abort [sync]`

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>chan</code></td>
<td>(object)</td>
<td>A Channel object</td>
</tr>
<tr>
<td><code>sync</code></td>
<td>(boolean)</td>
<td>wait (or not) for the analyzer to stop before processing subsequent commands. Optional argument; if unspecified, value is set to False. Choose from: True - synchronize - the analyzer will not process subsequent commands until the current measurement is aborted. False - continue processing commands immediately</td>
</tr>
</tbody>
</table>

**Return Type**
None

**Default**
None

**Examples**
- `chan.abort True`
- `chan.abort`

**C++ Syntax**
`HRESULT Abort(VARIANT_BOOL bSynchronize);`

**Interface**
IChannel
## AbortPowerAcquisition Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Aborts a source power cal acquisition sweep that is currently in progress.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>powerCalibrator.AbortPowerAcquisition</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>powerCalibrator</code></td>
<td><em>(object) - A SourcePowerCalibrator object</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>powerCalibrator.AbortPowerAcquisition</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT AbortPowerAcquisition();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISourcePowerCalibrator</td>
</tr>
</tbody>
</table>
AcquireCalStandard Method - **Superseded**

**Description**

*Note:* This command has been replaced by *AcquireCalStandard2 Method*, which provides for acquisition of sliding load standards. All other functionality is identical.

**VB Syntax**

`cal.AcquireCalStandard std[,index]`

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cal</code></td>
<td>A Calibrator <em>(object)</em></td>
</tr>
<tr>
<td><code>std</code></td>
<td><em>(enum NACalClass)</em> Standard to be measured. Choose from:</td>
</tr>
</tbody>
</table>

1. naClassA
2. naClassB
3. naClassC
4. naClassD
5. naClassE
6. naReferenceRatioLine
7. naReferenceRatioThru

**SOLT Standards**

1. naSOLT_Open
2. naSOLT_Short
3. naSOLT_Load
4. naSOLT_Thru
5. naSOLT_Isolation

**TRL Standards**

1. naTRL_Reflection
2. naTRL_Line_Reflection
3. naTRL_Line_Tracking
4. naTRL_Thru
5. naTRL_Isolation

| `index` | *(long integer)* number of the standard. Optional argument - Used if there is more than one standard required to cover the necessary frequency range. If unspecified, value is set to 1. |

*Note* The behavior has changed with VNA revisions as follows:
Before 6.01: Accepted 0 and changed it to 1

6.01 to 6.04: Did NOT accept 0

6.04.11 and higher: Accepts 0 and changes it to 1

Return Type
None

Default
Not Applicable

Examples
`Cal.AcquireCalStandard naSOLT_Thru 'Write`

C++ Syntax
`HRESULT AcquireCalStandard(tagNACalClass enumClass, short standardNumber)`

Interface
ICalibrator
Write-only

About Calibration Standards

AcquireCalStandard2 Method

Description
Measures the specified standard from the selected calibration kit. The calibration kit is selected using app.CalKitType.

For 2-port calibration, it is also necessary to specify direction with AcquisitionDirection.

To omit Isolation from a 2-port calibration, do not Acquire a cal standard for naSOLT_Isolation.

For using two sets of standards, see Simultaneous2PortAcquisition Property.

Note: This command replaces AcquireCalStandard. This command provides for the acquisition of a sliding load cal. All other functionality is identical.

VB Syntax
`cal.AcquireCalStandard2 std[,index][,slide]`

Variable (Type) - Description

*cal*  A Calibrator (object)

*std*  (enum NACalClass) Standard to be measured. Choose from:

1 - naClassA
2 - naClassB
3 - naClassC
4 - naClassD
5 - naClassE
6 - naReferenceRatioLine
7 - naReferenceRatioThru

**SOLT Standards**
1 - naSOLT_Open
2 - naSOLT_Short
3 - naSOLT_Load
4 - naSOLT_Thru
5 - naSOLT_Isolation

**TRL Standards**
1 - naTRL_Reflection
2 - naTRL_Line_Reflection

2583
3 - naTRL_Line_Tracking
4 - naTRL_Thru
5 - naTRL_Isolation

[index] (long integer) Number of the standard. Optional argument - Used if there is more than one standard required to cover the necessary frequency range. If unspecified, value is set to 1.

Note The behavior has changed with VNA revisions as follows:

- Before 6.01: Accepted 0 and changed it to 1
- 6.01 to 6.04: Did NOT accept 0
- 6.04.11 and higher: Accepts 0 and changes it to 1

[slide] (enum as NACalStandardSlidingState) Optional argument. State of the sliding load. The slide should be set a minimum of five times. Seven is the maximum that can be stored. Choose from:

0 - naNotSlidingStd - not using a sliding load - Default if not specified.
1 - naSlideIsSet - slide is set for acquisition
2 - naSlideIsDone - this next acquisition will be the last. Calculations will then be performed.

Return Type None
Default Not Applicable
Examples

C++ Syntax

<table>
<thead>
<tr>
<th>Examples</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal.AcquireCalStandard2 naSOLT_Thru</td>
<td>measures the second standard listed in the class of naSOLT_Thru</td>
</tr>
<tr>
<td>Cal.AcquireCalStandard2 naSOLT_Thru,2,naNotSlidingStd</td>
<td></td>
</tr>
</tbody>
</table>

Interface ICalibrator

[2584]
# AcquireCalConfidenceCheckECALEx Method

**Description**

This method replaces AcquireCalConfidenceCheckECAL

Transfers confidence data from the specified ECAL module into the measurement's memory trace. The data is transferred to the specified S-parameter on the same channel as this Calibrator object.

The characterization within the ECAL module that the confidence data will be read from is specified by ECALCharacterizationEx. The default value is 0.

**Note:** A confidence check can NOT be performed remotely from User Characterizations that are stored on the VNA disk.

**VB Syntax**

```vbnet
cal.AcquireCalConfidenceCheckECALEx Sparam [,ecalModule]
```

**Variable**

<table>
<thead>
<tr>
<th>(Type) - Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>cal</td>
<td>A Calibrator (object)</td>
</tr>
<tr>
<td>Sparam</td>
<td>S-parameter to transfer confidence data to. This parameter must be present on the same channel as the calibrator object.</td>
</tr>
<tr>
<td>ecalModule</td>
<td>(Integer) – Optional argument. ECAL module.</td>
</tr>
</tbody>
</table>

Choose from modules **1** through **8**

Use IsECALModuleFoundEx to determine the number of modules connected to the VNA

Use GetECALModuleInfoEx to return the model and serial number of each module.

**Return Type**

None

**Default**

Not applicable

**Examples**

Cal.AcquireCalConfidenceCheckECALEx "S11", 2

**C++ Syntax**

```cpp
HRESULT AcquireCalConfidenceCheckECALEX(BSTR strParameter, long moduleNumber = 1);
```

**Interface**

ICalibrator4
### AcquirePowerReadingsEx Method

<table>
<thead>
<tr>
<th>Description</th>
<th>This command replaces AcquirePowerReadings Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Initiates a source power cal acquisition.</td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
powerCalibrator.AcquirePowerReadingsEx calMethod, acqdevice [,sync]
```

**Variable**

- **powerCalibrator** *(object)* - A `SourcePowerCalibrator` object
- **calMethod** *(enum NASourcePowerCalMethod)* Selects the calibration method to be used for the source power cal acquisition.
  - 0 – **naPowerMeter** Use power meter for all readings.
  - 1 - **naPowerMeterAndReceiver** Power meter for the first iteration; then use the reference receiver for remaining readings if necessary.
  - 2 - **naReceiver** Use VNA measurement receiver for all readings.
- **acqdevice** *(String)* The specific acquisition device to be used. NOT case sensitive. Choose from:
  - If `calMethod` = **naPowerMeter** or **naPowerMeterAndReceiver**, choose from:
    - “ASEN” – Sensor on power meter channel A.
    - “BSEN” – Sensor on power meter channel B.
    - To use the sensor that currently corresponds to the frequency of interest, use the value from the `PowerAcquisitionDevice` property.
  - If `calMethod` = **naReceiver**, choose from:
    - The receiver names for your specific VNA using either physical receiver notation or logical receiver notation. For example, "a1" or "A".
    - Any configured PMAR device name. Learn more about PMAR Devices. See PMAR commands
- **[sync]** *(boolean)* Optional argument. If not specified, value is set to False. Choose from:
  - **True (1)** – The method does not return until this acquisition has completed (the program calling this method is halted while waiting for the method to return).
False (0) – The method initiates an acquisition then returns immediately (while the acquisition still proceeds). The program calling this method can then perform other operations during the acquisition.

| Return Type | None |
| Default     | Not Applicable |
| Examples    |
|             | `powerCalibrator.AcquirePowerReadingsEx naPowerMeter, "asen", True` |
|             | `powerCalibrator.AcquirePowerReadingsEx naReceiver, "b2"` |
|             | `powerCalibrator.AcquirePowerReadingsEx naReceiver, "MyPMAR"` |

**C++ Syntax**

```
HRESULT AcquirePowerReadingsEx (tagNASourcePowerCalMethod enumCalMethod, BSTR bstrAcqDevice, VARIANT_BOOL bSync);
```

**Interface**

ISourcePowerCalibrator4
### AcquireStep Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Acquire the measurement data for the specified step in the calibration process. For an ECal User characterization this measures the ECal module.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note</td>
<td>Guided Cal allows you to measure standards in any order. See an example.</td>
</tr>
<tr>
<td>VB Syntax</td>
<td><code>obj.AcquireStep (n)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>obj</code></td>
<td>Any of the following:</td>
</tr>
<tr>
<td></td>
<td>GuidedCalibration (object)</td>
</tr>
<tr>
<td></td>
<td>SMCType (object)</td>
</tr>
<tr>
<td></td>
<td>VMCType (object)</td>
</tr>
<tr>
<td></td>
<td>ECalUserCharacterizer (object) - Currently, only ONE step is required to measure the ECal module.</td>
</tr>
<tr>
<td><code>n</code></td>
<td>Step number in the calibration process.</td>
</tr>
<tr>
<td></td>
<td>Use GenerateSteps to determine the total number of steps.</td>
</tr>
<tr>
<td></td>
<td>Use GetStepDescription to read the description of each step.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>VMC.AcquireStep (3)</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT put_AcquireStep(long step);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>SMCType</td>
</tr>
<tr>
<td></td>
<td>VMCType</td>
</tr>
<tr>
<td></td>
<td>IGuidedCalibration</td>
</tr>
<tr>
<td></td>
<td>IECalUserCharacterizer</td>
</tr>
</tbody>
</table>
Write-only

Activate Method

Description
Makes an object the Active Object. When making a measurement active, the channel and window the measurement is contained in becomes the active channel and active window.

In order to change properties on any of the active objects, you must first have a "handle" to the active object using the Set command. For more information, See Getting a Handle to an Object.

You do not have to make an object "Active" to set or read its properties remotely. But an object must be "Active" to change its values from the front panel.

VB Syntax

object.Activate

Variable
(Type) - Description

object Measurement (object)
or
Marker (object)

Return Type
Not Applicable

Default
Not Applicable

Examples
meas.Activate
mark.Activate

C++ Syntax

HRESULT Activate()

Interface
IMeasurement
IMarker
## About Markers

**ActivateMarker Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Makes a marker the Active Marker. Use <code>meas.ActiveMarker</code> to read the number of the active marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.ActivateMarker(Mnum)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement <em>(object)</em></td>
</tr>
<tr>
<td><code>Mnum</code></td>
<td><em>(long integer)</em> - the number of the marker to make active. Choose any marker number from 1 to 15.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meas.ActivateMarker(1)</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT ActivateMarker(long lMarkerNumber)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IMeasurement</code></td>
</tr>
<tr>
<td><strong>Remarks</strong></td>
<td>Use <code>ReferenceMarkerState</code> to control the Reference marker.</td>
</tr>
</tbody>
</table>
ActivateWindow Method

**Description**
Makes a window object the Active Window.

In order to change properties on any of the active objects, you must first have a "handle" to the active object using the **Set** command. For more information, See Programming the Analyzer Object Model.

You do not have to make an object "Active" to set or read its properties remotely. But an object must be "Active" to change its values from the front panel.

**VB Syntax**
```
app.ActivateWindow n
```

**Variable**
(Type) - Description

**app**
An **Application** (**object**)  

**n**
(**long**) Number of the window to make active

**Return Type**
Window Object

**Default**
Not Applicable

**Examples**
```
app.ActivateWindow 4
```

**C++ Syntax**
```
HRESULT ActivateWindow(long WindowNumber)
```

**Interface**
IApplication

See the VNA Object Model
Add (channels) Method

Description  Creates a channel and returns a handle to it. If the channel already exists, it returns the handle to the existing channel.

VB Syntax  

```vbnet
chans.Add (item)
```

Variable  

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>chans</code></td>
<td>A Channel collection (object)</td>
</tr>
<tr>
<td><code>item</code></td>
<td>(variant) - Channel number.</td>
</tr>
</tbody>
</table>

Return Type  Channel

Default  Not Applicable

Examples  

```vbnet
chans.Add 3 'Creates channel 3
```

C++ Syntax  

```cpp
HRESULT Add(VARIANT numVal, IChannel** pChannel)
```

Interface  IChannels
Add (measurement) Method

Description: Adds a Measurement to the collection.

**Note:** This command is supported ONLY in a standard measurement channel.

Measurements can be added to ALL measurement class types using `CreateCustomMeasurementEx` Method.

VB Syntax:

```vbnet
meas.Add channel,param,source[,window]
```

- **meas** A Measurements collection (object)
- **channel** (long) - Channel number of the new measurement.
- **param** (string) - New parameter. Case insensitive.

**For S-parameters:**

Any S-parameter that can be measured by your VNA.

Single-digit port numbers can be separated by "_" (underscore). For example: "S21" or "S2_1"

Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

**For Ratioed measurements:**

Any two receivers in your VNA separated by "/". For example: "A/R1"

See the block diagram showing the receivers in YOUR VNA.

**For Unratioed (absolute power) measurements:**

Any receiver in the VNA. For example: "A"

See the block diagram showing the receivers in YOUR VNA.

**Ratioed** and **Unratioed** measurements can also use **logical receiver notation** to refer to receivers. This notation makes it easy to refer to receivers with an external test set connected to the VNA. You do not need to know which physical receiver is used for each test port. Learn more.

**For ADC measurements**

Any ADC receiver in the VNA followed by a comma, then the source port.
For example: "AI1_2" indicates the Analog Input1 with source port of 2.

Learn more about ADC receiver measurements.

For Balanced S-parameter measurements:

"topology:Sabxy"

**topology** - Choose from:

- **sbal** - single-ended to balanced
- **ssb** - single-ended / single-ended to balanced
- **bbal** - balanced to balanced

**Sabxy** -

Where

- **a** - device output (receive) mode
- **b** - device input (source) mode

(choose from the following for both **a** and **b**):

- **d** - differential
- **c** - common
- **s** - single ended

- **x** - device output (receive) logical port number
- **y** - device input (source) logical port number

For example: "**sbal:sdd42**"

See an example program

For **Imbalance** and **Common Mode Rejection** measurements:

"topology:parameter" Choose from:
<table>
<thead>
<tr>
<th>Choose this:</th>
<th>To get this:</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;SBAL:IMBSB&quot;</td>
<td>single-ended to balanced</td>
<td>imbalance</td>
</tr>
<tr>
<td>&quot;SBAL:CMRRSB1&quot;</td>
<td>single-ended to balanced</td>
<td>common mode rejection (Sds21/Scs21)</td>
</tr>
<tr>
<td>&quot;SBAL:CMRRSB2&quot;</td>
<td>single-ended to balanced</td>
<td>common mode rejection (Ssd12/Ssc12)</td>
</tr>
<tr>
<td>&quot;SSB:IMB1SSB&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>imbalance 1</td>
</tr>
<tr>
<td>&quot;SSB:IMB2SSB&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>imbalance 2</td>
</tr>
<tr>
<td>&quot;SSB:CMRRSSB1&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>common mode rejection (Sds31/Scs31)</td>
</tr>
<tr>
<td>&quot;SSB:CMRRSSB2&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>common mode rejection (Sds32/Ssc32)</td>
</tr>
<tr>
<td>&quot;BBAL:IMB1BB&quot;</td>
<td>balanced to balanced</td>
<td>imbalance 1</td>
</tr>
<tr>
<td>&quot;BBAL:IMB2BB&quot;</td>
<td>balanced to balanced</td>
<td>imbalance 2</td>
</tr>
<tr>
<td>&quot;BBAL:CMRRBB&quot;</td>
<td>balanced to balanced</td>
<td>common mode rejection (Sdd21/Scc21)</td>
</tr>
</tbody>
</table>

**source** *(long integer)* - Source port number; if unspecified, value is set to 1. Only used for non-s-parameter measurements; ignored if s-parameter.

**window** *(long integer)* - Optional argument. Window number of the new measurement. If unspecified, the S-Parameter will be created in the Active Window. Choose between 1 and the maximum number of windows allowed on the VNA. If unspecified, the measurement will be created in the Active Window.

See also Traces, Channels, and Windows on the VNA

Return Type: None  
Default: None  
Examples: `meass.Add 3, "A/R1", 1, 1` - Adds A/R1 measurement to channel 3 in window 1
C++ Syntax

HRESULT Add(long ChannelNum, BSTR strParameter, long srcPort, VARIANT_BOOL bNewWindow)

Interface
IMeasurements
Add (NAWindows) Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Add a window to the display. Does not add a measurement. The window number must not already exist.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>wins.Add [item]</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description <em>(variant)</em> - Window number. Choose between 1 and the maximum number of windows allowed on the VNA.</td>
</tr>
<tr>
<td><code>wins</code></td>
<td>A NAWindow collection <em>(object)</em></td>
</tr>
<tr>
<td><code>item</code></td>
<td></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Object</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>wins.Add 3 'Creates a window number 3</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Add(long windowNumber )</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>INAWindows</td>
</tr>
</tbody>
</table>
Add (PowerLossSegment) Method

**Description**

Adds a PowerLossSegment to the PowerLossSegments collection. Also, adds a PowerLossSegmentPMAR to the PowerLossSegmentsPMAR collection.

To ensure predictable results, it is best to remove all segments before defining a new list of segments. For each segment in the collection, do a seg.Remove.

Segments and values can also be added using the CharacterizeAdaptor Macro.

**VB Syntax**

```vb
segs.Add (item [ size])
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>- Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>segs</code></td>
<td>A PowerLossSegments collection</td>
</tr>
<tr>
<td></td>
<td>A PowerLossSegmentsPMAR collection.</td>
</tr>
<tr>
<td><code>item</code></td>
<td>(variant) - Number of the new segment. If it already exists, a new segment is inserted at the requested position.</td>
</tr>
<tr>
<td><code>size</code></td>
<td>(long integer) - Optional argument. The number of segments to add, starting with item. If unspecified, value is set to 1. Add up to 9999 segments.</td>
</tr>
</tbody>
</table>

**Return Type**

None

**Default**

Not Applicable

**Examples**

```vb
segs.Add 1, 4 'Adds segments 1,2,3 and 4
```

**C++ Syntax**

```cpp
HRESULT Add(VARIANT index, long size);
```

**Interface**

IPowerLossSegments
Add (PowerSensorCalFactorSegment) Method

Description
Adds a PowerSensorCalFactorSegment to the CalFactorSegments collection. Also
adds a PowerSensorCalFactorSegmentPMAR to the CalFactorSegmentsPMAR collection.

To ensure predictable results, it is best to remove all segments before defining a new list of segments. For each segment in the collection, do a seg.Remove.

VB Syntax
segs.Add (item [ size])

Variable (Type) - Description
segs - A CalFactorSegments (collection) or a CalFactorSegmentsPMAR (collection)
item - (variant) - Number of the new segment. If it already exists, a new segment is inserted at the requested position.
size - (long integer) - Optional argument. The number of segments to add, starting with item. If unspecified, value is set to 1.

Return Type
None

Default
Not Applicable

Examples
segs.Add 1, 4 'Adds segments 1,2,3 and 4

C++ Syntax
HRESULT Add(VARIANT index, long size);

Interface
ICalFactorSegments
ICalFactorSegmentsPMAR
## Add (segment) Method

**Description**  Adds segments to the Segments collection, but does not turn the segments ON.

**VB Syntax**  

```vb
segs.Add (item, [size])
```

- `segs`  A segments collection *(object)*
- `item`  *(variant)* Number of the new segment. If it already exists, a new segment is inserted at the requested position.
- `size`  *(long integer)* Optional argument. The number of segments to add, starting with `item`. If unspecified, value is set to 1.

**Return Type**  None

**Default**  None

**Examples**  

```vb
Segs.Add 1, 4 'Adds segments 1, 2, 3, and 4. (does NOT automatically turn segments ON)
```

**C++ Syntax**  

```c++
HRESULT Add(VARIANT index, long size);
```

**Interface**  ISegments

**Remarks**  To ensure predictable results, it is best to remove all segments before defining a segment list. For each segment in the collection, do a `seg.Remove`. 
### Add (External Device) Method

**Description**  
Adds an external device to the system. This is the same as clicking the **New** button and editing the name on the Configure an External Device dialog.  

Upon creation, all settings on the new device are set to the defaults. The device is not active until set using `Ext.Dev.Active`.

**VB Syntax**  
`extDevices.Add name`

**Variable**  
*(Type)* - Description

- `extDevices`  
  An `ExternalDevices` *(collection)*
- `name`  
  *(String)* - Name of the new external device.

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`extDevices.Add 'MySource'`  
*Creates a new external device*

**C++ Syntax**  
`HRESULT Add (BSTR name)`

**Interface**  
`IExternalDevices`
### Add (GuidedPowerSensors) Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Adds a power sensor name and item number to be used during a source power calibration. Use when multiple power sensors are to be used to calibrate the entire frequency span. The Name is used to recognize the sensor in the User Interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Item numbers in the GuidedCalibrationPowerSensors collection are used to refer to the power sensor remotely. Use the Count Property to return the number of power sensor items that are configured for use on the channel.</td>
</tr>
<tr>
<td></td>
<td>The port number to be calibrated is set using the PerformPowerCalibration Property.</td>
</tr>
<tr>
<td>VB Syntax</td>
<td><code>sensors.Add (name)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>sensors</strong> (Type) - Description</td>
</tr>
<tr>
<td></td>
<td><strong>name</strong> (String) - Name of the power sensor to add. The power sensor must be already configured as a PMAR device using this name. Learn how to remotely configure a PMAR device.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>sensors.Add &quot;pmar2&quot;</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT Add(BSTR name);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IGuidedCalibrationPowerSensors</td>
</tr>
</tbody>
</table>
### Add (Testset) Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Adds a testset to the ExternalTestsets Collection and loads the configuration file.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>testsets.Add(model,address)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>testsets</code></td>
<td>An ExternalTestsets (collection)</td>
</tr>
<tr>
<td><code>model</code></td>
<td>(String) Model of the testset to be added, NOT case-sensitive.</td>
</tr>
</tbody>
</table>

There is no COM command to read a list of currently-supported test sets. However, the following SCPI command can be used with the following format:

```vbnet
string = SCPIStringParser.Execute("SENSe:MULTiplexer:CATalog?")
```

<table>
<thead>
<tr>
<th><strong>address</strong></th>
<th>(Integer) Address of the testset to be added.</th>
</tr>
</thead>
</table>

| **Return Type** | Not Applicable |
| **Default** | Not Applicable |

| **Examples** | `testsets.Add("Z5623AK66",12) ' add Z5623AK66 test at address 12 to testsets collection` |

See an example program

| **C++ Syntax** | HRESULT Add(BSTR typename, long address) |
| **Interface** | IExternalTestsets |
AddPowerCalRange Method

<table>
<thead>
<tr>
<th>Description</th>
<th>This command adds a power cal range for a specific port &lt;n&gt;. Note that external sources are valid and specifying a source port is the same as remote commands. By default this will create a range with the preset start/stop frequency and 201 points. The maximum number of ranges that can be added is 100 (same as the maximum number of segments).</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>IndependentPwrrCalPort.AddPowerCalRange</code></td>
</tr>
<tr>
<td>Variable</td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>IndependentPwrrCalPort</code></td>
<td>A <code>IndependentPowerCalibrationPort</code> object</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Example</td>
<td><code>CalibrateAllChannels.IndependentPowerCalibrationPort(3).AddPowerCalRange</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT AddPowerCalRange</code></td>
</tr>
<tr>
<td>Interface</td>
<td><code>IIndependentPowerCalibrationPort</code></td>
</tr>
</tbody>
</table>
### AddSegment Method

**Description**  
Adds the specified number of segments to the scratch mixer at the index position. All segments are added with default settings.

**VB Syntax**  
`conv.AddSegment index,count`

**Variable**  
<table>
<thead>
<tr>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>conv</code></td>
</tr>
<tr>
<td><code>index</code></td>
</tr>
<tr>
<td><code>count</code></td>
</tr>
</tbody>
</table>

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`mxr.AddSegment 1,5 'Adds 5 segments beginning at the first position.`

See example program

**C++ Syntax**  
`HRESULT AddSegment(long index, long count);`

**Interface**  
IConverter5
## AllowAllEvents Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets event filtering to monitor all events in the analyzer. This is the default setting when subscribing to events. This could slow the measurement speed of the analyzer significantly.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.AllowAllEvents</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>app.AllowAllEvents</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT AllowAllEvents()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>
### AllowChannelToSweepDuringCalAcquisition Method

**Description**
Specifies the channel to sweep during a Calibration.

When this command is sent, the `SwpChan` channel is 'flagged' to be swept during calibration. The flag is cleared when the channel is deleted, if the Measurement Class is changed, or if all measurements are deleted from the channel. If the same channel number is recreated, this command must be sent again to sweep the channel during a calibration. The flag is NOT saved with an instrument state.

A Preset or Instrument State Recall deletes the channel.

**VB Syntax**

```vb
calMgr.AllowChannelToSweepDuringCalAcquisition (CalChan, SwpChan, State)
```

**Variable (Type) - Description**

- **`calMgr` (object)** - A `CalManager` object
- **`CalChan` (long)** - Channel to be calibrated.
- **`SwpChan` (long)** - The channel to sweep when waiting to measure a standard.
- **`state` (Boolean)** - Channel sweep state. Choose from:
  - **True** - Sweep the channel during calibration.
  - **False** - Do NOT sweep the channel during calibration.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**

```vb
calMgr.AllowChannelToSweepDuringCalAcquisition 2,1,True
```

See example using this command

**C++ Syntax**

```c++
HRESULT AllowChannelToSweepDuringCalAcquisition ( long CalChannel, long SwpChannel, VARIANT_BOOL bVal);
```

**Interface**
ICalManager5
AllowEventCategory Method

**Description**
Sets event filtering to monitor a category of event.

**VB Syntax**
`app.AllowEventCategory, category, state`

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
<tr>
<td>category</td>
<td>Category to monitor. Choose from list in Working with the Analyzer's Events</td>
</tr>
<tr>
<td>state</td>
<td>(boolean)</td>
</tr>
<tr>
<td></td>
<td>True - monitor</td>
</tr>
<tr>
<td></td>
<td>False - do not monitor</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
`app.AllowEventCategory`

**C++ Syntax**
`HRESULT AllowEventCategory(tagNAEventCategory category, VARIANT_BOOL bAllow )`

**Interface**
IApplication
AllowEventMessage Method

Description  Sets event filtering to monitor specific events.

VB Syntax  

\[ app.\text{AllowEventMessage} \text{ event} \]

Variable  

(Type) - Description

\( app \)  An Application (object)

\( event \)  Event to monitor. Refer to list in Working with the Analyzer's Events

\( state \)  (boolean)

True - monitor

False - do not monitor

Return Type  Not Applicable

Default  Not Applicable

Examples  

\[ app.\text{AllowEventMessage} \]

C++ Syntax  

\[
\text{HRESULT AllowEventMessage( tagNAEventID eventID, VARIANT_BOOL bAllow) }
\]

Interface  IApplication
### AllowEventSeverity Method

**Description**  
Sets event filtering to monitor levels of severity.

**VB Syntax**  
```vb
app.AllowEventSeverity severity, state
```

**Variable (Type) - Description**

- `app` An Application (object)
- `severity` (enum `naEventSeverity`)  
  - Choose from: `naEventSeverityERROR`
  - `naEventSeverityINFORMATIONAL`
  - `naEventSeveritySUCCESS`
  - `naEventSeverityWARNING`
- `state` (boolean)  
  - `True` - monitor
  - `False` - do not monitor

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`app.AllowEventSeverity`

**C++ Syntax**  
```cpp
HRESULT AllowEventSeverity(tagNAEventSeverity severity, VARIANT_BOOL bAllow)
```

**Interface**  
IApplication
## Apply Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Applies the mixer setup and turns the channel ON. (Performs the same function as the Apply button on the mixer setup dialog box.) Learn about the Scratch and Applied mixer properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>obj.Apply</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td></td>
<td>obj A Mixer Interface pointer to the Measurement (object)</td>
</tr>
<tr>
<td></td>
<td>Or</td>
</tr>
<tr>
<td></td>
<td>A Converter Object</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>mxr.Apply</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT Apply()</td>
</tr>
<tr>
<td>Interface</td>
<td>IMixer3</td>
</tr>
<tr>
<td></td>
<td>IConverter</td>
</tr>
</tbody>
</table>
ApplyDeltaMatchFromCalSet Method

Description

Specifies a Cal Set as a source of delta match correction.

If ‘GUID’ is not supplied then the Global Delta Match Cal Set is assumed. An error is returned if the specified Cal Set does not meet the following Delta Match criteria. The Global Delta Match Cal can ALWAYS be applied.

- Must have been performed using ECal or as a guided mechanical cal (not Unguided).
- Must have the same start freq, stop freq, and number of points as the channel being calibrated.
- Must calibrate the ports that are required by the TRL or Unknown Thru Cal as indicated by PortsNeedingDeltaMatch Property.

Learn more about Delta Match calibration.

VB Syntax

guided.ApplyDeltaMatchFromCalSet [GUID]

Variable (Type) - Description

**guided** GuidedCalibration (object)

**GUID** Optional Argument. GUID of the Cal Set to use. If unspecified, the Global Delta Match Cal Set is used.

Return Type Not Applicable

Default Not Applicable

Examples

```vbnet
guided.ApplyDeltaMatchFromCalSet "{2B893E7A-971A-11d5-8D6C-00108334AE96}"
```

C++ Syntax

HRESULT ApplyDeltaMatchFromCalSet(BSTR calsetGUID);

Interface IGuidedCalibration2
# ApplyPowerCorrectionValuesEx Method

**Description**

This command replaces `ApplyPowerCorrectionValues Method`. Applies the array of power correction values to the channel memory and turns correction ON. Perform after completing a source power cal acquisition sweep.

This command does NOT save the correction values. To save correction values, save an instrument / calibration state (*.cst file) after performing a source power cal.

 Optionally, as part of the source power calibration, perform calibration of the reference receiver used in the power calibration. Learn more.

**VB Syntax**

```vbnet
powerCalibrator.ApplyPowerCorrectionValuesEX [rRec]
```

**Variable (Type) - Description**

- `powerCalibrator` *(object)* - A `SourcePowerCalibrator` object
- `rRec` *(Enum as NASourcePowerApplyCorrectionOption)* Optional argument. Choose from:
  - **0 - naSourcePowerApplyCorrectionDefault** Do NOT perform and save a calibration of the reference receiver. (Default if not specified).
  - **1 - naIncludeReferenceReceiverPowerCal** Perform and save a calibration of the reference receiver. The Cal Set, which includes only the reference receiver cal, is saved to the destination specified by `RemoteCalStoragePreference`.

**Return Type**

None

**Default**

Not Applicable

**Examples**

```vbnet
powerCalibrator.ApplyPowerCorrectionValuesEX
powerCalibrator.ApplyPowerCorrectionValuesEX (naIncludeReferenceReceiverPowerCal)
```

**C++ Syntax**

```cpp
HRESULT ApplyPowerCorrectionValuesEx(enum NASourcePowerApplyCorrectionOption option);
```

**Interface**

`ISourcePowerCalibrator`
About Source Power Calibration

ApplySourcePowerCalibrationTo Method

Description
Copies and applies an existing Source Power Calibration to another channel.

VB Syntax
`chan.ApplySourcePowerCalibrationTo (fromPortNum, targetChan, targetPortNum);`

Variable (Type) - Description
- `chan` - A Channel (object)
- `fromPortNum` - (Long) Port number of the existing source power correction.
- `targetChan` - (Long) Channel number to which the source power correction will be copied.
- `targetPortNum` - (Long) Port number to which the source power correction will be applied.

Return Type
Not Applicable

Default
Not Applicable

Examples
`chan.ApplySourcePowerCalibrationTo 1, 2, 1`

C++ Syntax
`HRESULT ApplySourcePowerCalibrationTo (long fromPortNumber, long otherChannelNumber, long portNumber);`

Interface
IChannel11
AssignSourceToRole Method

Description
Assigns a configured source to the specified role.

Use chan.RoleDevice for non-converter channels.

VB Syntax
conv.AssignSourceToRole role, source

Variable (Type) - Description
conv A Converter Object
role (String) Role to which the external source is assigned. Choose from:

For IMDX and IMSX, choose from:
"RF2"
"LO1"
"LO2"

For all other converter applications, choose from:
"LO1"
"LO2"

source (String) Source name from Source Configuration dialog.

Return Type Not Applicable
Default Not Applicable

Examples conv.AssignSourceToRole "LO1", "LO1Name"

C++ Syntax HRESULT AssignSourceToRole(BSTR roleID, BSTR deviceName);

Interface IConverter
# AutoOrient Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the ECal port that is connected to the specified VNA port. A calibration does not have to be in process.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>ecalPortNumber = ecal.AutoOrient(chanNum, pnaPort, ecalCharNum)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>ecalPortNumber</code></td>
<td>(Long) Variable to store the returned ECal port number that is connected to the specified VNA port number. The returned ECal port number is a 1-based number: 1 = Port A, 2 = Port B, 3 = Port C, 4 = Port D. Zero (0) is returned when the auto-orientation routine is unable to resolve the orientation.</td>
</tr>
<tr>
<td><code>ecal</code></td>
<td>A ECalModule Object (object)</td>
</tr>
<tr>
<td><code>chanNum</code></td>
<td>(Long) Channel number that contains the frequency range that will be calibrated.</td>
</tr>
<tr>
<td><code>PNAPort</code></td>
<td>(Long) VNA port number.</td>
</tr>
<tr>
<td><code>ecalCharNum</code></td>
<td>(Long) User Characterization number that matches the physical adapters/fixtures that are on the ECal module. This aids in determining the orientation of the ECal module.</td>
</tr>
<tr>
<td>Choose from:</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>Factory characterization (no adapters - data that was stored in the ECal module by Keysight)</td>
</tr>
<tr>
<td>1</td>
<td>User characterization #1</td>
</tr>
<tr>
<td>2</td>
<td>User characterization #2</td>
</tr>
<tr>
<td>...and so forth up to:</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>User characterization #12</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long Integer</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
| Examples | ```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
```
Dim pna
pna.Preset
Const chanNum = 1
pna.Channels(chanNum).StopFrequency = 20E9 ' for a 20 GHz ECal mod
Const pnaPortNumber = 1
Const ecalCharacterizationNum = 0
Dim calMgr
Set calMgr = pna.GetCalManager
Dim ecalPortNumber ' The returned ECal port number is a 1-based number
' (1 = Port A, 2 = Port B, etc)
ecalPortNumber = calMgr.ECalModules(1).AutoOrient(chanNum, pnaPortNumber, ecalCharacterizationNum)
MsgBox "ECal port number attached to PNA port 1 = " & ecalPortNumber

C++ Syntax   HRESULT AutoOrient( long channel, long pnaPortNumber, long characterization, long *pECalPortNumber);

Interface   IECalModule
## AutoPortExtMeasure Method

**Description**
Measures either an OPEN or SHORT standard. When this command is sent, the VNA acquires the measurement with which to set automatic port extensions. Learn more about choosing which standard to measure.

**VB Syntax**
```
fixture.AutoPortExtMeasure value
```

**Variable**
- **Type** - Description
- **fixture** - A Fixturing (object)
- **value** - (Enum as NAAutoPortExtMeasure)

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>naAPEM_OPEN - Measure OPEN</td>
</tr>
<tr>
<td>1</td>
<td>naAPEM_SHORT - Measure SHORT</td>
</tr>
</tbody>
</table>

**Return Type**
ENUM

**Default**
Not Applicable

**Examples**
```vbnet
fixture.AutoPortExtMeasure naAPEM_OPEN
```

**C++ Syntax**
```
HRESULT get_AutoPortExtMeasure(tagNAAutoPortExtMeasure *pVal);
```

**Interface**
IFixturing2
AutoPortExtReset Method

Description: Clears old port extension delay and loss data in preparation for acquiring new data. Send this command prior to sending a new series of measurements using AutoPortExtMeasure Method. If acquiring both OPEN and SHORT standards, do not send this command between those acquisitions.

VB Syntax: `fixture.AutoPortExtReset`

Variable: `(Type) - Description`

- `fixture`: A `Fixturing (object)`

Return Type: Not Applicable

Default: Not Applicable

Examples: `fixture.AutoPortExtReset`

C++ Syntax: `HRESULT AutoPortExtReset();`

Interface: `IFixturing2`
## Autoscale Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Trace Object - Autoscales only the ONE trace on which Autoscale is being called.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NAWindow Object - Scales ALL of the traces to fit in the same window. This is equivalent to &quot;Autoscale All&quot; from the front panel.</td>
</tr>
<tr>
<td></td>
<td>Autoscale (both trace and window) behaves differently when scale coupling is enabled. How it behaves depends on the scale coupling method. Learn more.</td>
</tr>
</tbody>
</table>

### VB Syntax

```vbnet
object.Autoscale
```

### Variable

- **(Type)** - Description
- **object**  
  - Trace *(object)*
  - or
  - NAWindow *(object)*

### Return Type

Not Applicable

### Default

Not Applicable

### Examples

- Trac.Autoscale 'Autoscales the trace
- Win.Autoscale 'Autoscales all the traces in the window

### C++ Syntax

```cpp
HRESULT AutoScale()
```

### Interface

INAWindow
ITrace
## AveragingRestart Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Clears and restarts averaging of the measurement data.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan.AveragingRestart</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>chan</code></td>
</tr>
<tr>
<td><strong>(Type)</strong></td>
<td>A Channel <strong>(object)</strong></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chan.AveragingRestart</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT AveragingRestart()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel</td>
</tr>
</tbody>
</table>

*Note: Channel object is used in the context of measurement data.*
### BandnoiseData Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the band noise level in dBm/Hz from the band noise marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = mkr.BandnoiseData</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>Variable to store the returned Y-axis value.</td>
</tr>
<tr>
<td><code>mkr</code></td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>See example program</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_BandnoiseData(double* pVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker6</td>
</tr>
</tbody>
</table>
### BandpowerData Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the band power level from the band power marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = mkr.BandpowerData</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td>Variable to store the returned Y-axis value.</td>
</tr>
<tr>
<td><code>mkr</code></td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>See example program</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_BandpowerData(double* pVal)</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker6</td>
</tr>
</tbody>
</table>
# BuildHybridKit Method

**Description**
Use this method when you have different port connectors. This is a convenient way to combine two kits that match the connectors on your DUT.

**VB Syntax**
```vb
app.BuildHybridKit port1Kit, p1sex, port2Kit, p2sex, adapter, user kit
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>object</td>
<td>An Application</td>
</tr>
<tr>
<td><code>port1Kit</code></td>
<td>enum NACalKit</td>
<td>Specifies the two kits to be used to build the hybrid kit. Choose from:</td>
</tr>
<tr>
<td><code>port2Kit</code></td>
<td></td>
<td>Choose from:</td>
</tr>
<tr>
<td><code>p1sex</code></td>
<td>enum NAPortSex</td>
<td>Specifies the sex of the connector at that port. Choose from:</td>
</tr>
<tr>
<td><code>p2sex</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>adapter</code></td>
<td>enum NAAdapter</td>
<td>Choose from:</td>
</tr>
<tr>
<td><code>userKit</code></td>
<td>enum NACalKit</td>
<td>The Hybrid kit - Choose from the previous list of kits</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vb
app.BuildHybridKit
naCalKit_85033E_3_5, naMale, naCalKit_85038A_7_16,
naFemale, naUserkit, naCalKit_User8
```
C++ Syntax

HRESULT BuildHybridKit(tagNACalKit port1Kit, tagNAPortSex port1Sex, tagNACalKit port2Kit, tagNAPortSex port2Sex, tagNAAdapter adapter, tagNACalKit userKit)

Interface

IApplication
### CalculateErrorCoefficients Method

**Description**
This method is the final call in a calibration process. It calculates error-correction terms, turns error-correction ON and saves the error-correction terms to the channel’s Cal Register or a User Cal Set.

Do NOT use this command during an ECAL.

**Note:** The destination (Cal Register or User Cal Set) is determined by the setting of the RemoteCalStoragePreference property.

**VB Syntax**
```
cal.CalculateErrorCoefficients
```

**Variable (Type) - Description**
- **cal** Calibrator (object)

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
Cal.CalculateErrorCoefficients
```

**C++ Syntax**
```
HRESULT CalculateErrorCoefficients()
```

**Interface**
ICalibrator
#### Calculate Method

**Description**
Calculates the Input or Output frequencies of the mixer setup, applies the mixer setup to the mixer object, and turns the channel ON.

**Note:** There is also a Calculate Method on the Converter Object

**VB Syntax**

\[
\text{obj}.\text{Calculate} \ (\text{port})
\]

**Variable**

Type - Description

- **obj**  A Mixer Interface pointer to the Measurement (object)
- **port** (enum as MixerCalculation) Port of the mixer for which to calculate start and stop frequencies. Choose from:

<table>
<thead>
<tr>
<th>enum</th>
<th>1st or only stage requires:</th>
<th>In addition, 2nd stage requires:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>naCalculateINPUT</td>
<td>• IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>• Output Start and Stop frequencies</td>
<td>• 2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td>• LO frequency</td>
<td>• IF sideband (High or Low)</td>
</tr>
<tr>
<td></td>
<td>• Output sideband (High or Low)</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>naCalculateINPUT AndOUTPUT Where (2 stage mixers ONLY)</td>
<td>• IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>NA</td>
<td>• Both LO frequencies</td>
</tr>
<tr>
<td>2</td>
<td>naCalculateOUTPUT</td>
<td>• IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>• Input Start and Stop frequencies</td>
<td>• 2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td>• LO frequency</td>
<td>• IF sideband (High or Low)</td>
</tr>
<tr>
<td></td>
<td>• Output sideband (High or Low)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>naCalculateLO1</td>
<td>• IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>• Input Start and Stop frequencies</td>
<td>• 2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td>• Output frequency</td>
<td>• IF sideband (High or Low)</td>
</tr>
<tr>
<td></td>
<td>• Output sideband (High or Low)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>naCalculateLO2</td>
<td>• Input Start and stop</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
<td></td>
</tr>
<tr>
<td>Examples</td>
<td><code>obj.Calculate (mixCalculateOUTPUT)</code></td>
<td></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT Calculate()</td>
<td></td>
</tr>
<tr>
<td>Interface</td>
<td>IMixer</td>
<td></td>
</tr>
</tbody>
</table>

- 1st LO start and stop frequencies
- Output frequency
- IF sideband (High or Low)
- Output sideband (High or Low)
**Calculate Method**

**Description**: Calculates the Input or Output frequencies of the mixer setup, applies the mixer setup to the mixer object, and turns the channel ON.

**Note**: There is also a Calculate Method on the IMixer Interface.

**VB Syntax**

`obj.Calculate (port)`

**Variable**

(Type) - Description

- `obj` A Converter Object
- `port` (enum as ConverterCalculation) Port of the mixer for which to calculate start and stop frequencies. Choose from:

<table>
<thead>
<tr>
<th>enum</th>
<th>1st or only stage requires:</th>
<th>In addition, 2nd stage requires:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>naCalculateINPUT</td>
<td>Output Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LO frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output sideband (High or Low)</td>
</tr>
<tr>
<td>1</td>
<td>naCalculateINPUT AndOUTPUT</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>(2 stage mixers ONLY)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>naCalculateOUTPUT</td>
<td>Input Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LO frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output sideband (High or Low)</td>
</tr>
<tr>
<td>3</td>
<td>naCalculateLO1</td>
<td>Input Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output sideband (High or Low)</td>
</tr>
<tr>
<td>4</td>
<td>naCalculateLO2</td>
<td>NA</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
<td></td>
</tr>
<tr>
<td>Examples</td>
<td><code>obj.Calculate (naCalculateOUTPUT)</code></td>
<td></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT Calculate()</td>
<td></td>
</tr>
<tr>
<td>Interface</td>
<td>IConverter</td>
<td></td>
</tr>
</tbody>
</table>

- 1st LO start and stop frequencies
- Output frequency
- IF sideband (High or Low)
- Output sideband (High or Low)
### ChangeParameter Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Changes the parameter of the measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>meas.ChangeParameter(param,src)</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>param</td>
<td>(string) - New parameter. Case insensitive.</td>
</tr>
</tbody>
</table>

For **S-parameters and Applications parameters**:

Single-digit port numbers can be separated by "_" (underscore). For example: "S21" or "S2_1"

Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

For **Ratioed receiver measurements**:

Any two receivers in your VNA separated by "/". For example: "A/R1"

See the block diagram showing the receivers in YOUR VNA.

For **Unratioed (absolute power) measurements**:

Any receiver in the VNA. For example: "A"

See the block diagram showing the receivers in YOUR VNA.

**Ratioed** and **Unratioed** measurements can also use **logical receiver notation** to refer to receivers. This notation makes it easy to refer to receivers with an external test set connected to the VNA. You do not need to know which physical receiver is used for each test port. Learn more.

For **ADC measurements**

Any ADC receiver in the VNA.

For example: "AI1" indicates the Analog Input1.

Learn more about ADC receiver measurements.

For **Balanced S-parameter measurements**:
"topology:Sabxy"

topology - Choose from:

- **sbal** - single-ended to balanced
- **ssb** - single-ended / single-ended to balanced
- **bbal** - balanced to balanced

Sabxy -

Where

- **a** - device output (receive) mode
- **b** - device input (source) mode

(choose from the following for both a and b:)

- **d** - differential
- **c** - common
- **s** - single ended

- **x** - device output (receive) logical port number
- **y** - device input (source) logical port number

For example: "sbal:sdd42"

See an example program

For Imbalance and Common Mode Rejection measurements:

"topology:parameter" Choose from:
<table>
<thead>
<tr>
<th>Choose this:</th>
<th>To get this:</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;SBAL:IMBSB&quot;</td>
<td>single-ended to balanced</td>
<td>imbalance</td>
</tr>
<tr>
<td>&quot;SBAL:CMRRSB1&quot;</td>
<td>single-ended to balanced</td>
<td>common mode rejection (Sds21/Scs21)</td>
</tr>
<tr>
<td>&quot;SBAL:CMRRSB2&quot;</td>
<td>single-ended to balanced</td>
<td>common mode rejection (Ssd12/Ssc12)</td>
</tr>
<tr>
<td>&quot;SSB:IMB1SSB&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>imbalance</td>
</tr>
<tr>
<td>&quot;SSB:IMB2SSB&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>imbalance</td>
</tr>
<tr>
<td>&quot;SSB:CMRRSSB1&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>common mode rejection (Sds31/Scs31)</td>
</tr>
<tr>
<td>&quot;SSB:CMRRSSB2&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>common mode rejection (Sds32/Scs32)</td>
</tr>
<tr>
<td>&quot;BBAL:IMB1BB&quot;</td>
<td>balanced to balanced</td>
<td>imbalance</td>
</tr>
<tr>
<td>&quot;BBAL:IMB2BB&quot;</td>
<td>balanced to balanced</td>
<td>imbalance</td>
</tr>
<tr>
<td>&quot;BBAL:CMRRBB&quot;</td>
<td>balanced to balanced</td>
<td>common mode rejection (Sdd21/Scc21)</td>
</tr>
</tbody>
</table>

**src** (long integer)

- Ignored if `param` is an S-Parameter

- Source port if `param` is a ratioed or unratioed receiver measurement (including ADC measurements).

**Note:** If the port is defined by a string name, such as an external source, a balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources, then you must use `chan.getPortNumber` to translate the string.
into a port number. To learn more see Remotely Specifying a Source Port.

Return Type  Not Applicable
Default       Not Applicable
Examples
meas.ChangeParameter "S11",2 '2 is ignored
meas.ChangeParameter "VC21",1 '1 is ignored
meas.ChangeParameter "A/R1",2 '2 is the source port
meas.ChangeParameter "a1/b1",1 '1 is the source port
meas.ChangeParameter "R1",2 '2 is the source port

'to change to a parameter with a string name
Dim app
Set app = CreateObject("Agilentpna835x.application")
Dim capabilities
Set capabilities = app.Capabilities
Dim portnum
Portnum = Capabilities.GetPortNumber("Src2 Out1")
App.activemeasurement.ChangeParameter "A", portnum

C++ Syntax HRESULT ChangeParameter (BSTR parameter, long lPort)
Interface IMeasurement
CheckPower Method

**Description**
Measures power at a specified frequency. Use this method to test power level before and/or after applying a source power calibration.

**VB Syntax**
```vbnet
pow = pwrCal.CheckPower(device, freq [,unit])
```

**Variable**
- **pow** *(double)* Variable to store power value returned by this method.
- **pwrCal** *(SourcePowerCalibrator)*
- **device** *(enum NAPowerAcquisitionDevice)* The specific sensor on the power meter to be used for the acquisition. Choose from:
  - 0 – naPowerSensor_A
  - 1 – naPowerSensor_B

To use the sensor that currently corresponds to the frequency of interest, use the value from the `PowerAcquisitionDevice` property.

- **freq** *(double)* Frequency (Hz) at which the sensor is to read the power.

- **unit** *(enum NAPowerUnit)* Optional argument. Choose from:
  - `naDBM` – Returns the power in dBm. (default)
  - `naWATT` – Returns the power in Watts.

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```vbnet
watt = powerCalibrator.CheckPower(naPowerSensor_A, 1E9, naWATT)
```

**C++ Syntax**
```cpp
HRESULT put_CheckPower(tagNAPowerAcquisitionDevice enumAcqDevice, double dFreq, tagNAPowerUnit enumPowerUnit, double *pdPower);
```

**Interface**
ISourcePowerCalibrator2
## Clear Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Clears the current diagnostic information.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>embedLODiag.Clear</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>embedLODiag</code></td>
<td>An <code>EmbeddedLODiagnostic (object)</code></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>embedLO.Clear 'write</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT Clear();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IEmbededLODiagnostic</code></td>
</tr>
</tbody>
</table>
### Clear Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Clears the FIFO data buffer.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>fifo.Clear</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>fifo</code>&lt;br&gt;<strong>Type</strong>&lt;br&gt;A FIFO <strong>(object)</strong></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>fifo.Clear</code> <strong>'write</strong></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Clear();</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFIFO</td>
</tr>
</tbody>
</table>
CloseCalSet Method  **Superseded**

**Description**
This command is no longer necessary. The CalSet.get... and put... commands that required this command have been replaced.

Closes read/write access to the Cal Set.

See OpenCalSet for an explanation of gaining access to the Cal Set.

When you are finished reading and writing data from or to the Cal Set, close the Cal Set. Subsequent read/writes will require a new OpenCal Set call.

Reading and writing Cal Set data is performed with the PutStandard, GetStandard, PutErrorTerm, GetErrorTerm method calls. These methods are provided by the ICal Set and ICalData2 interfaces.

**VB Syntax**
```vbnet
CalSet.CloseCalSet
```

**Variable**  
(Type) - Description

*CalSet*  
(object) - A Cal Set object

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**

CSet.CloseCalSet

**C++ Syntax**
```cpp
HRESULT CloseCalSet
```

**Interface**  
ICalSet
## Close Method

**Description**  Closes the specified remote VISA session. VISA sessions should always be closed when you are finished communicating with the remote device. Use this command to close (end) each VISA session that was opened successfully using the Open method. If you have more than one open session, and need to close them all at the same time, it may be faster and easier to use the Reset method.

**VB Syntax**  

```vbnet
Vpassthru.Close visaID
```

**Variable (Type) - Description**

- **Vpassthru**  *(object)* - A VISAPassthrough object
- **visaID**  *(long)* - VISA session number (see Open method).

**Return Type**  Not Applicable

**Default**  Not Applicable

**Examples**  

```vbnet
Vpassthru.Close 2
```

**C++ Syntax**  

```cpp
HRESULT Close(long session_num)
```

**Interface**  IVISAPassthrough
About Cal Sets

**ComputeErrorTerms Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Computes error terms for the caltype specified by a preceding OpenCal Set call.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The Cal Set must first be opened using OpenCalSet. If this call has not been made, the following error is issued:</td>
</tr>
<tr>
<td></td>
<td>E_NA_Cal Set_ACCESS_DENIED</td>
</tr>
<tr>
<td></td>
<td>The standards data required for the CalType must be available in the Cal Set or this error will be returned: E_NA_STANDARD_NOT_FOUND.</td>
</tr>
</tbody>
</table>

**Note:** Error term computation requires data for the actual calibration kit standards from the current kit definition. ComputeErrorTerms assumes that the standards were acquired using only one standard per class.

**VB Syntax**  
`CalSet.ComputeErrorTerms`

**Variable**  
- **CalSet** (object) - A Cal Set object

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`CalSet.ComputeErrorTerms`

**C++ Syntax**  
`HRESULT ComputeErrorTerms()`  
**Interface**  
ICalSet
**ConfigEnhancedNB2 Method**

**Description**

Note: This command replaces ConfigEnhancedNB Method.

This subroutine determines, then returns, the proper configuration for pulsed measurements on the PNA-X ONLY using the spectral nulling technique. The configuration returned needs to be sent to the VNA and any other related external equipment.

The routine will take a desired Pulse Repetition Frequency (PRF) and measurement IFBW and return a possibly modified PRF and IFBW for proper pulsed operation on the VNA.

Note: If an error is returned suggesting that nulling has not been found, add a small offset to the PRF (for example, 2.1 MHz instead of 2 MHz) or set Fixed PRF to False.

**VB Syntax**

```vbnet
Pulsed.ConfigEnhancedNB2 (PRF, BW, PhysicalIF, NCO, ClockFreq, Stage1TapArray, Stage2TapArray, Stage3TapArray, FixedPRF, GateDelay, GateWidth, SWGateDelay, SWGateWidth, SWGateRamp)
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulsed</td>
<td>(interface) An interface to the agilentpnapulsed.dll application interface.</td>
</tr>
<tr>
<td>PRF</td>
<td>(Double) The Pulse Repetition Frequency.</td>
</tr>
<tr>
<td></td>
<td>[out] The pulse repetition frequency that has been optimized for use with the</td>
</tr>
<tr>
<td></td>
<td>VNA. NOTE: This value may be different from the value requested.</td>
</tr>
<tr>
<td></td>
<td>[in] The desired pulse repetition frequency.</td>
</tr>
<tr>
<td>BW</td>
<td>(Long) The VNA IF Bandwidth.</td>
</tr>
<tr>
<td></td>
<td>[out] The VNA IF bandwidth that has been optimized for use with the VNA.</td>
</tr>
<tr>
<td></td>
<td>NOTE: This value may be different from the value requested. Zero (0) is</td>
</tr>
<tr>
<td></td>
<td>returned if no solution is found for the specified PRF and BW.</td>
</tr>
<tr>
<td></td>
<td>[in] The desired VNA IF bandwidth.</td>
</tr>
<tr>
<td>PhysicalIF</td>
<td>(Double)</td>
</tr>
<tr>
<td></td>
<td>[out] Returns physical intermediate frequency.</td>
</tr>
</tbody>
</table>
NCO (Double)

[out] Returns numeric controlled oscillator frequency.

ClockFreq (Double)

[out] Returns the clock frequency (in Hz) of the PNA-X.

Stage1TapArray (Long array)

[out] Returns the stage 1 filter coefficients

Stage2TapArray (Long array)

[out] Returns the stage 2 filter coefficients

Stage3TapArray (Long array)

[out] Returns the stage 3 filter coefficients

FixedPRF (Boolean)

[in]

- 1 (True) Signals the .DLL routine to NOT adjust the PRF value; rather adjust ONLY the IF Bandwidth. This is the default setting.
- 0 (False) Adjust both the PRF and IF Bandwidth values as necessary.

GateDelay (Double)

[in] Highest delay value in seconds used in any of the receiver gates.

GateWidth (Double)

[in] Widest pulse width value in seconds used in any of the receiver gates.

SWGateDelay (Double)

[out] Returns the SW gate delay in seconds.

SWGateWidth (Double)

[out] Returns the SW Gate width in seconds.

SWGateRamp (Long)

[out] Returns the SW Gate ramp

ReturnType Not Applicable

Default Not Applicable
Example  See an example using this command.

**C++ Syntax**

```c++
HRESULT ConfigEnhancedNB2(double *pPRF, long *pBW, double *pIF, double *pNCO, double *clock, double *pStg1, double *pStg2, double *pStg3, VARIANT_BOOL fixPRF, double gateDelay, double gateWidth, double *SWgateDelay, double *SWgateWidth, long *SWgateRamp)
```

**Interface**

AgilentPNAPulsed.Application
## ConfigEnhancedNBIFAtten Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets PNA-X receivers to auto gain setting.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><em>Pulsed.ConfigEnhancedNBIFAtten</em> <em>(PRF, RxWidth, IFAtten)</em></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><em>Pulsed</em></td>
<td><em>(interface)</em> An interface to the agilentpnapulsed.dll application interface.</td>
</tr>
<tr>
<td><em>PRF</em></td>
<td><em>(Double)</em></td>
</tr>
<tr>
<td><em>RxWidth</em></td>
<td><em>(Double)</em></td>
</tr>
<tr>
<td><em>IFAtten</em></td>
<td><em>(Long Integer)</em></td>
</tr>
<tr>
<td></td>
<td>[out] IF attenuation value.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>See an example using this command.</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT <em>ConfigEnhancedNBIFAtten</em>(double *pPRF, double *pWidth, long *pIF)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>AgilentPNAPulsed.Application</td>
</tr>
</tbody>
</table>
Pulsed Application

ConfigNarrowBand3 Method

Description

Note: This method replaces ConfigNarrowBand2 Method. The BW argument now returns 0 if no solution is found for the specified PRF and BW. In addition, adjustments were made to the filter finder algorithm.

This subroutine determines, then returns, the proper configuration for pulsed measurements on the VNA using the spectral nulling technique. The configuration returned needs to be sent to the VNA and any other related external equipment such as pulse generators. The routine will take a desired Pulse Repetition Frequency (PRF) and measurement IFBW and return a possibly modified PRF and IFBW for proper pulsed operation on the VNA. The routine will also return the Sample Rate, Number of Taps, and Offset that must be sent to the VNA to configure it in pulsed mode using the spectral nulling technique.

Although the example below uses COM programming to communicate with the VNA, these commands can be replaced with SCPI equivalents.

Note: The pulsed application may set the offset frequency (option S93080A/B) of the VNA to some value other than zero (the default value). If the stop frequency is set to the maximum of the VNA model, then an error message may appear on the VNA stating that the response frequency has exceeded the maximum allowed frequency. To fix this, set the stop frequency to a value that is at least 2 KHz less than the maximum allowed.

For example, if you have a 20 GHz VNA, and the stop frequency is set to 20 GHz, and the error message appears, then set the stop frequency to 19.999998 GHz.

VB Syntax

Pulsed ConfigNarrowBand (PRF, NumTaps, BW, OffSet, SampleRate, Precision, FixedPRF, PG81110)

Variable (Type) - Description

Pulsed (interface) An interface to the agilentpnapulsed.dll application interface.

PRF (Double) The Pulse Repetition Frequency.

[out] The pulse repetition frequency that has been optimized for use with the VNA. NOTE: This value may be different from the value requested.

[in] The desired pulse repetition frequency.

NumTaps (Long) The number of taps to send to the VNA for pulsed operation.
**BW** *(Long)* The VNA IF Bandwidth.

[out] The VNA IF bandwidth that has been optimized for use with the VNA. NOTE: This value may be different from the value requested. Zero (0) is returned if no solution is found for the specified *PRF* and *BW*.

[in] The desired VNA IF bandwidth.

**Offset** *(Double)* The offset value to send to the VNA for pulsed operation. The offset value is used to adjust the VNA for the two different possible sample rates that may be returned.

**SampleRate** *(Double)*

[out] The sample rate to send to the VNA for pulsed operation.

[in] Passing a value of 6.2 us will make sure that the offset frequency is not shifted and therefore could be used with converter measurements. Otherwise enter 0.

**Precision** *(Double)* The precision variables sets the precision that will be used to decrement the PRF when running the configuration routines. This variable can be set to the precision required by the external pulse generators so that the configuration routine will not return a PRF that is not within the precision limits of the pulse generators.

**FixedPRF** *(Boolean)*

1 (True) Signals the .DLL routine to NOT adjust the PRF value; rather adjust ONLY the IF Bandwidth. This is the default setting.

0 (False) Adjust both the PRF and IF Bandwidth values as necessary.

**PG81110** *(Boolean)*

1 (True) You are using an Keysight 81110 as the pulse generator. This allows increased accuracy in adjustments for offset and PRF.

0 (False) Not using an Keysight 81110.

**Return Type** Not Applicable

**Default** Not Applicable

**Example** Removed example from help file.

**C++ Syntax**

HRESULT ConfigNarrowBand(double *pPRF, long *pNumTaps, long *pBW, double *pOffset, double *pSampleRate, int Precision)

**Interface** AgilentPNAPulsed.Application
## Configure Method

**Description**
Restarts as an "N-port" VNA using the specified multiport test set.

See other commands to configure multiport test sets.

**VB Syntax**
```vbnet
app.Configure (model, address)
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
<tr>
<td>model</td>
<td>String - Model of the test set with which to restart. Use &quot;Native&quot; to restart without a test set.</td>
</tr>
<tr>
<td>address</td>
<td>Integer - GPIB Address of the test set. Use 0 for native restart.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```vbnet
app.Configure ("N44xx",18)
```

**C++ Syntax**
```cpp
HRESULT Configure(BSTR model, long address);
```

**Interface**
IApplication10
## Continuous Method

<table>
<thead>
<tr>
<th>Description</th>
<th>The channel continuously responds to trigger signals.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>This command does <strong>NOT</strong> change TriggerSignal to Continuous.</td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan.Continuous</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>chan</code> - Channel (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chan.Continuous</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Continuous()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel</td>
</tr>
</tbody>
</table>


**Copy Method**

**Description**
Creates a new Cal Set and copies the current Cal Set data into it. Therefore, you now have a clone Cal Set with a different ID. Use this command to manipulate data on a Cal Set without corrupting the original cal data.

**VB Syntax**

```
CalSet.Copy
```

**Variable**

- *(Type)* - Description
  - `CalSet` - A Cal Set object

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```vbs
Dim mgr As CalManager
Dim ocalset As CalSet
Dim newcalset As CalSet
Set mgr = pna.GetCalManager

'Create a new (empty) Cal Set.
Set ocalset = mgr.CreateCalSet(1)
oCalset.Description = "original calset"
pna.Channel(1).SelectCalSet ocalset.GetGUID, True

'Launch the cal wizard and allow the user to perform the calibration.
If pna.LaunchCalWizard(False) Then
  'If the Launch returns true then the calibration finished.
  ocalset.Save

  'Copy the Cal Set to the new one.
  Set newcalset = ocalset.Copy
  newcalset.Description = "copy of original calset"
Else
  'If the cal doesn't finish, delete the old Cal Set
  'so it isn't taking up unnecessary memory.
  mgr.DeleteCalSet ocalset.GetGUID
End If
```

As a result, the programmer can manipulate the data in the new Cal Set and always revert back to the old Cal Set as needed.

**C++ Syntax**

```
HRESULT Copy( ICalSet** pCalSet);
```

**Interface**

ICalSet
CopyFrom Method

Description
Copies the mechanical switch and attenuator settings from the specified channel to the calling channel.

To avoid potential conflicts, all port couplings in the calling channel will be turned OFF and all port attenuator settings will be set to manual before copying the switch or attenuator settings. The two channels CAN be of different measurement classes.

Use CopyToChannel to copy ALL settings from one channel to another.

VB Syntax

```
pathConfig.CopyFrom (chanNum)
```

Variable (Type) - Description

| pathConfig | A PathConfiguration (object) |
| chanNum    | (long integer) Channel number to copy to the calling channel. |

Return Type
None

Default
Not Applicable

Examples

```
Dim chan, pathConfig
Set chan = app.ActiveChannel
Set pathConfig = chan.PathConfiguration
pathConfig.CopyFrom 2
```

C++ Syntax

```
HRESULT CopyFrom(long chanNum);
```

Interface
IPathConfiguration2

Last modified:

New topic
CopyToChannel Method

Description
Copies ALL settings from this channel to the specified channel.

Use CopyFrom to copy ONLY the mechanical switch and attenuator settings.

VB Syntax
`chan.CopyToChannel(lChanNum)`

Variable (Type) - Description

`chan` A Channel (object)

`IChanNum` (long integer) – Number of the channel to become a copy of <chan>.

Return Type
None

Default
Not Applicable

Examples
```vba
Dim chan
Set chan = PNAapp.ActiveChannel
chan.CopyToChannel 2
```

C++ Syntax
`HRESULT CopyToChannel(long lChanNum);`

Interface
IChannel2
# CreateS-Parameter Method

**Description**  
This method creates a new S-Parameter measurement in an existing or new window.

**VB Syntax**  
```vbnet
app.CreateSParameter chan,receiver,source,[window]
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>Application (object)</td>
<td></td>
</tr>
<tr>
<td><code>chan</code></td>
<td>long integer</td>
<td>Channel number of the new measurement</td>
</tr>
<tr>
<td><code>receiver</code></td>
<td>long integer</td>
<td>Port number of the receiver (1 or 2)</td>
</tr>
<tr>
<td><code>source</code></td>
<td>long integer</td>
<td>Port number of the source (1 or 2)</td>
</tr>
<tr>
<td><code>window</code></td>
<td>long integer</td>
<td>Optional argument. Window number of the new measurement. Choose 1 to 4. If unspecified, the S-Parameter will be created in the Active Window.</td>
</tr>
</tbody>
</table>

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**

- `app.CreateSParameter 1,2,1,1` 'Creates a new S21 measurement in channel 1 and New window(1)
- `app.CreateSParameter 1,2,1` 'Creates a new S21 measurement in channel 1 and in the active window

**C++ Syntax**  
```cpp
HRESULT CreateSParameter(long ChannelNum, long RcvPort, long SrcPort, long windowNumber)
```

**Interface**  
IApplication
CreateCalSet Method

Description

Creates a new Cal Set.

The new cal set is initialized with the stimulus settings from the channel whose number is passed as the argument to this method. Stimulus settings include frequency, bandwidth, number of points, and so forth.

Use this method when you want to manually upload data to the Cal Set using the returned ICal Set interface handle.

The channel number does not restrict the usage of this Cal Set on any other channel. It simply provides a link to the originating channel so that the stimulus values can be stored in the Cal Set.

Note: Be sure to SAVE the CalSet you are creating. Use ICalSet::Save.

VB Syntax

```vbnet
calMgr.CreateCalSet(chan)
```

Variable

- **Type**: Description
- **calMgr**: (object) - A CalManager object
- **chan**: (long) - channel number of the new Cal Set.

Return Type

ICal Set Interface

Default

Not Applicable

Example

```
calMgr.CreateCalSet 1
```

C++ Syntax

```c
HRESULT CreateCalSet( long ChannelNumber, ICal Set** pCal Set);
```

Interface

ICalManager
### CreateCustomCal Method

**Description**
Creates a custom cal object.

**VB Syntax**
```vba
calMgr.CreateCustomCal(CalType)
```

**Variable (Type) - Description**
- `calMgr` Cal Manager (Object)
- `CalType` (String) Name of the calibration. Choose from:
  - "VMC" or "VectorMixerCal.VMCType"
  - "SMC" or "ScalarMixerCal.SMCType"

**See Also**
- SMCType Object
- VMCType Object

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vba
Dim CalMgr As ICalManager2
Dim SMC As ISMCType
Set SMC = CreateCustomCal("SMC")
```

See SMC and VMC examples using this command.

**C++ Syntax**
```cpp
HRESULT CreateCustomCal( BSTR CustomCal)
```

**Interface**
ICalManager2
CreateCustomCalEx Method

**Description**

Returns IGuidedCalibration for the specified channel.

With a handle to the IGuidedCalibration interface, you can query it for the following objects for properties appropriate to the calibration setup for the particular meastype (channel).

IGuidedCalibration interface is used to configure a calibration (specify connectors, cal kits, and so forth). It is also used to access any custom calibration properties required for unique application channels like Noise Figure or Gain Compression. To access these special properties, make this call on the IGuidedCalibration interface:

```
CustomInterface = IGuidedCalibration.CustomCalConfiguration();
```

The interface returned by this call can be used to set and get the custom properties on the following application cal objects:

- NoiseCal Object
- GainCompressionCal Object
- SweptIMDCal Object

**Note:** Use CreateCustomCal_Ex Method to create FCA calibration objects.

**VB Syntax**

calmgr.CreateCustomCalEx (chan)

**Variable (Type) - Description**

- `calMgr` Cal Manager (Object)
- `chan` (long integer) Channel number in which to create the Cal object.

**Return Type**

IGuidedCalibration

**Default**

Not Applicable

**Examples**

```vbnet
Dim guidedcal
Set guidedcal = CalManager.CreateCustomCalEx(1)
```

**See Also**

- Noise Figure example
- Gain Compression example
C++ Syntax  HRESULT CreateCustomCalEx(long channel, IDispatch** ppObject);

Interface  ICalManager5
**CreateCustomMeasurementEx Method**

**Description**
Creates a new custom measurement or a new 'standard' S-Parameter measurement.

**VB Syntax**
```vb
app.CreateCustomMeasurementEx chanNum, MeasClass, MeasName [, window]
```

**Variable (Type) - Description**
- **app** (object) - An Application object
- **chanNum** (long) - Channel number used by the new measurement; can exist or be a new channel.
- **MeasClass** (string) - Measurement class of the new custom measurement object. The Measurement Class must be installed and registered on the VNA.

Choose from the following (click or scroll down to view valid MeasNames for each MeasClass):

- "Standard"
- "Active Hot Parameters"
- "Vector Mixer/Converter"
- "Scalar Mixer/Converter"
- "Gain Compression"
- "Gain Compression Converters"
- "Modulation Distortion"
- "Noise Figure Cold Source"
- "Noise Figure Converters"
- Phase Noise
- "Swept IMD"
- "IM Spectrum"
- "Swept IMD Converters"
- "IM Spectrum Converters"
- "Differential I/Q"
**Spectrum Analyzer**

**MeasName** (variant) Measurement names to create:

<table>
<thead>
<tr>
<th>Meas Class</th>
<th>Measurement Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Standard&quot;</td>
<td>&quot;S11&quot;, &quot;S21&quot;, and so forth</td>
<td>S-parameter name</td>
</tr>
<tr>
<td></td>
<td>&quot;A_1&quot;, &quot;A_2&quot;, and so forth</td>
<td>Unratioted parameter names with notation:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;receiver_source port&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See Balanced S-parameter measurement names</td>
</tr>
<tr>
<td>Active Hot Parameters</td>
<td>&quot;HotS11&quot;</td>
<td>Learn about Active Hot parameters</td>
</tr>
<tr>
<td></td>
<td>&quot;HotS31&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;HotS13&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;HotS33&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;IPwr&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;OPwr&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Gamma&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Pmax&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Xs(3,3)&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Xt(3,3)&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Xf(3,1)&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;DeltaOPwr&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;Vector Mixer/Converter&quot;</td>
<td>&quot;S11&quot;</td>
<td>Learn about VMC parameters</td>
</tr>
<tr>
<td></td>
<td>&quot;VC21&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;S22&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;Vector Mixer/Converter&quot;</td>
<td>&quot;S11&quot;</td>
<td>Learn about VMC parameters</td>
</tr>
<tr>
<td></td>
<td>&quot;SC21&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;SC12&quot;</td>
<td></td>
</tr>
</tbody>
</table>
### Gain Compression

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CompIn21</td>
<td>Input power at the compression point.</td>
</tr>
<tr>
<td>CompOut21</td>
<td>Output power at the compression point.</td>
</tr>
<tr>
<td>CompGain21</td>
<td>Gain at the compression point.</td>
</tr>
<tr>
<td>CompS11</td>
<td>Input Match at the compression point.</td>
</tr>
<tr>
<td>RefS21</td>
<td>Linear Gain</td>
</tr>
<tr>
<td>DeltaGain21</td>
<td>CompGain21 - Linear Gain</td>
</tr>
<tr>
<td>S11, S21, S12, S22</td>
<td>Standard S-parameters; measured at port 1 and port 2</td>
</tr>
</tbody>
</table>

### GCX - All Gain Compression parameters (except S21 and S12) plus the following:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S11</td>
<td>Mixer parameters</td>
</tr>
<tr>
<td>SC21</td>
<td>Power Out</td>
</tr>
<tr>
<td>SC12</td>
<td>Power In</td>
</tr>
<tr>
<td>S22</td>
<td>Modulation Signal Out</td>
</tr>
</tbody>
</table>

### Modulation Distortion:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>POut2</td>
<td>Power Out</td>
</tr>
<tr>
<td>PIn1</td>
<td>Power In</td>
</tr>
<tr>
<td>MSig2</td>
<td>Modulation Signal Out</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>&quot;MDist2&quot;</td>
<td>Modulation Distortion Out</td>
</tr>
<tr>
<td>&quot;MGain21&quot;</td>
<td>Modulation Gain</td>
</tr>
<tr>
<td>&quot;MComp21&quot;</td>
<td>Modulation Compression Power Gain</td>
</tr>
<tr>
<td>&quot;PGain21&quot;</td>
<td>Linear Input Match</td>
</tr>
<tr>
<td>&quot;S11&quot;</td>
<td>Linear Gain</td>
</tr>
<tr>
<td>&quot;S21&quot;</td>
<td>Linear Input Match</td>
</tr>
<tr>
<td>&quot;LPIn1&quot;</td>
<td>Linear Reflected Power In</td>
</tr>
<tr>
<td>&quot;LPOut1&quot;</td>
<td>Linear Power Out</td>
</tr>
<tr>
<td>&quot;LPOut2&quot;</td>
<td>Port 1 test port receiver</td>
</tr>
<tr>
<td>&quot;A&quot;, &quot;b1&quot;</td>
<td>Port 2 test port receiver</td>
</tr>
<tr>
<td>&quot;B&quot;, &quot;b2&quot;</td>
<td>Port 1 reference receiver</td>
</tr>
<tr>
<td>&quot;R1&quot;, &quot;a1&quot;</td>
<td>Port 2 reference receiver</td>
</tr>
<tr>
<td>&quot;R2&quot;, &quot;a2&quot;</td>
<td>Input Band Power</td>
</tr>
<tr>
<td>&quot;CarrIn1&quot;</td>
<td>Output Band Power</td>
</tr>
<tr>
<td>&quot;CarrOut2&quot;</td>
<td>Band Power Gain</td>
</tr>
<tr>
<td>&quot;CarrGain21&quot;</td>
<td>ACP at input</td>
</tr>
<tr>
<td>&quot;ACPIn1&quot;</td>
<td>ACP at output</td>
</tr>
<tr>
<td>&quot;ACPOut2&quot;</td>
<td>ACP distortion, Added by DUT</td>
</tr>
<tr>
<td>&quot;ACPDist21&quot;</td>
<td>EVM Equalized Distortion, Added by DUT</td>
</tr>
<tr>
<td>&quot;EVMDistEq21&quot;</td>
<td>EVM Unequalized Distortion, Added by DUT</td>
</tr>
<tr>
<td>&quot;EVMDistUn21&quot;</td>
<td></td>
</tr>
<tr>
<td>Noise Figure AND NFX:</td>
<td></td>
</tr>
<tr>
<td>&quot;NF&quot;</td>
<td>Noise figure</td>
</tr>
<tr>
<td>&quot;ENR&quot;</td>
<td>Validate noise source measurements.</td>
</tr>
<tr>
<td>&quot;T-Eff&quot;</td>
<td>Effective noise temperature.</td>
</tr>
<tr>
<td>&quot;DUTRNP&quot;</td>
<td>DUT noise power ratio. (Noise power expressed in Kelvin divided by 290).</td>
</tr>
<tr>
<td>&quot;DUTRNPI&quot;</td>
<td>System noise power ratio</td>
</tr>
<tr>
<td>&quot;SYSRNP&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;SYSRNPI&quot;</td>
<td></td>
</tr>
</tbody>
</table>
### Noise Figure Cold Source

- **"DUTNPD"** DUT noise power density. (Noise power expressed in dBm/Hz).
- **"DUTNPDI"**
- **"SYSNPD"** System noise power density.
- **"SYSNPDI"**
- **"OvrRng"** (Opt 029 Only) Indication that the noise receiver is being over powered.
- **"T-Rcvr"** (Opt 029 Only) Temperature reading (in Kelvin) of the noise receiver board.

### Noise Figure ONLY - NOT NFX:

- **"A_1", "A_2" ...and so forth.** Unratioed parameters; with notation: "receiver, source port"
- **"GammaOpt"** Optimum Complex Reflection Coefficient
- **"Rn"** Noise Resistance
- **"NFMin"** Minimum noise figure that occurs at GammaOpt

### NFX ONLY:

- **"S11"**
- **"SC21"**
- **"SC12"**
- **"S22"** Mixer parameters
- **"Ipwr"**
- **"RevIPwr"**
- **"Opwr"**
### Unratioed parameters with notation:

- "receiver_source port"

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;RevOPwr&quot;</td>
<td>Test port receiver at LO1 frequency</td>
</tr>
<tr>
<td>&quot;ALOI&quot;,&quot;BLO1&quot;</td>
<td>...and so forth.</td>
</tr>
<tr>
<td>&quot;R1_1&quot;,&quot;B_2&quot;</td>
<td>...and so forth.</td>
</tr>
<tr>
<td>&quot;IMD Spectrum&quot;</td>
<td>There are over 150 possible Swept IMD parameters, too many to list here.</td>
</tr>
<tr>
<td>&quot;PwrMainLo&quot;</td>
<td>Absolute power of the Low tone at the DUT output.</td>
</tr>
<tr>
<td>&quot;IM3&quot;</td>
<td>Power of the third product relative to the average power of the f1 and f2 tones measured at the DUT output.</td>
</tr>
<tr>
<td>&quot;OIP3&quot;</td>
<td>Theoretical power level at which the third product will be the same power level as the average of the main tones at the output of the DUT.</td>
</tr>
<tr>
<td>&quot;Output&quot;</td>
<td>View signals OUT of the DUT and into VNA port 2 (B receiver).</td>
</tr>
<tr>
<td>&quot;Input&quot;</td>
<td>View signals IN to the DUT (R1 receiver).</td>
</tr>
<tr>
<td>Module</td>
<td>Method Description</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>&quot;Reflection&quot;</strong></td>
<td>View signals reflected off the DUT input and back into VNA port 1 (A receiver)</td>
</tr>
<tr>
<td><strong>&quot;Output&quot;</strong></td>
<td>View signals OUT of the DUT and into VNA port 2 (B receiver)</td>
</tr>
<tr>
<td><strong>&quot;Differential I/Q&quot;</strong></td>
<td>Create custom parameters using DefineParameter Method, then specify your custom parameter name here.</td>
</tr>
<tr>
<td><strong>&quot;Spectrum Analyzer&quot;</strong></td>
<td>Reference receiver</td>
</tr>
<tr>
<td><strong>&quot;ImageReject&lt;n&gt;&quot;</strong></td>
<td>Test port receiver</td>
</tr>
</tbody>
</table>

**window** *(long)* Optional argument. Number of the window the new custom measurement will be placed in. Choose between 1 and the maximum number of windows allowed on the VNA.. If unspecified, the measurement is placed in the active window.

**Return Type** IMeasurement

**Default** Not Applicable

**Examples**

```
To create a scalar mixer measurement in channel 2:
Dim MyMeas as Agilent835x.Measurement
Set MyMeas = app.CreateCustomMeasurementEx (2, "Scalar Mixer/Converter", "SC21")
```
To create a vector mixer measurement in channel 2:
Dim MyMeas as Agilent835x.Measurement
Set MyMeas = app.CreateCustomMeasurementEx (2, "Vector Mixer/Converter", "VC21")

C++ Syntax
HRESULT put_CreateCustomMeasurementEx (long ChannelNum, BSTR guid, VARIANT initData, long windowNumber, IMeasurement** ppMeasurement );

Interface
IApplication3
CreateMeasurement Method

Description: Creates a new measurement.

VB Syntax: `app.CreateMeasurement chanNum, param, IPort[, window]`

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>Application (object)</td>
</tr>
<tr>
<td><code>chanNum</code></td>
<td>(long) - Channel number of the new measurement; can exist or be a new channel</td>
</tr>
<tr>
<td><code>param</code></td>
<td>(string) - New parameter. Case insensitive.</td>
</tr>
</tbody>
</table>

For S-parameters:

- Any S-parameter that can be measured by your VNA.
- Single-digit port numbers can be separated by "_" (underscore). For example: "S21" or "S2_1"
- Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

For Ratioed measurements:

- Any two receivers in your VNA separated by "/". For example: "A/R1"

See the block diagram showing the receivers in YOUR VNA.

For Unratioed (absolute power) measurements:

- Any receiver in the VNA. For example: "A"

See the block diagram showing the receivers in YOUR VNA

**Ratioed** and **Unratioed** measurements can also use **logical receiver notation** to refer to receivers. This notation makes it easy to refer to receivers with an **external test set** connected to the VNA. You do not need to know which physical receiver is used for each test port. Learn more.

For ADC measurements

- Any ADC receiver in the VNA.

For example: "AI1" indicates the Analog Input1.
Learn more about ADC receiver measurements.

For **Balanced S-parameter measurements**:

"topology: Sabxy"

**topology** - Choose from:

- **sbal** - single-ended to balanced
- **ssb** - single-ended / single-ended to balanced
- **bbal** - balanced to balanced

**Sabxy** -

Where

- **a** - device output (receive) mode
- **b** - device input (source) mode

(choose from the following for both a and b:)

- **d** - differential
- **c** - common
- **s** - single ended

- **x** - device output (receive) logical port number
- **y** - device input (source) logical port number

For example: "**sbal:sdd42**"

See an example program

For **Imbalance and Common Mode Rejection measurements**:

"topology: parameter" Choose from:
<table>
<thead>
<tr>
<th>Choose this:</th>
<th>To get this:</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;SBAL:IMBSB&quot;</td>
<td>single-ended to balanced</td>
<td>imbalance</td>
</tr>
<tr>
<td>&quot;SBAL:CMRRSB1&quot;</td>
<td>single-ended to balanced</td>
<td>common mode rejection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sds21/Scs21)</td>
</tr>
<tr>
<td>&quot;SBAL:CMRRSB2&quot;</td>
<td>single-ended to balanced</td>
<td>common mode rejection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Ssd12/Ssc12)</td>
</tr>
<tr>
<td>&quot;SSB:IMB1SSB&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>imbalance 1</td>
</tr>
<tr>
<td>&quot;SSB:IMB2SSB&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>imbalance 2</td>
</tr>
<tr>
<td>&quot;SSB:CMRRSSB1&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>common mode rejection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sds31/Scs31)</td>
</tr>
<tr>
<td>&quot;SSB:CMRRSSB2&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>common mode rejection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sds32/Scs32)</td>
</tr>
<tr>
<td>&quot;BBAL:IMB1BB&quot;</td>
<td>balanced to balanced</td>
<td>imbalance 1</td>
</tr>
<tr>
<td>&quot;BBAL:IMB2BB&quot;</td>
<td>balanced to balanced</td>
<td>imbalance 2</td>
</tr>
<tr>
<td>&quot;BBAL:CMRRBB&quot;</td>
<td>balanced to balanced</td>
<td>common mode rejection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sdd21/Scc21)</td>
</tr>
</tbody>
</table>

**IPort (long)**

- **Ignored** if `param` is an S-Parameter, balanced, imbalance, or CMRR parameter.
- **Source port** if `param` is ratioed or unratioed (including ADC) measurements.
- Use 0 for N5264B.

**Note:** If the port is defined by a string name, such as an external source, a
balanced port, or one of the Source 2 outputs on the 2-port 2-source VNA-x model, then you must use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port.

<table>
<thead>
<tr>
<th>window</th>
<th>(long) Optional argument. Window number of the new measurement. Choose between 1 and the maximum number of windows allowed on the VNA. If unspecified, the measurement will be created in the Active Window.</th>
</tr>
</thead>
</table>

See also Traces, Channels, and Windows on the VNA

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Not Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app.CreateMeasurement(1,&quot;A/R1&quot;,1,0)</code></td>
</tr>
<tr>
<td><code>app.CreateMeasurement(1,&quot;a1/b1&quot;,1,0)</code></td>
</tr>
<tr>
<td><code>app.CreateMeasurement(1,&quot;bbal:Sdd21&quot;,1)</code></td>
</tr>
<tr>
<td><code>app.CreateMeasurement(1,&quot;AI2&quot;,2)</code></td>
</tr>
<tr>
<td><code>app.CreateMeasurement(1,&quot;R1&quot;,0) ' for N5264B</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C++ Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>HRESULT CreateMeasurement(long ChannelNum, BSTR strParameter, long lPort, long windowNumber)</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>IApplication</td>
</tr>
</tbody>
</table>
# DataToMemory Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Stores the active measurement data into memory creating a memory trace. The memory can then be displayed or used in calculations with the measurement data.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>meas.DataToMemory</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>meas.DataToMemory</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT DataToMemory()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>

2672
**Deembed Method**

**Description**
De-embeds a fixture from an existing Cal Set based on an S2P file. A new Cal Set is created with the effects of the fixture removed.

When the new Cal Set is applied to a channel, the effects of the fixturing are removed from the measurement data. Do NOT enable fixturing. The effects of the fixture are removed when the new Cal Set is selected and correction is turned ON.

**VB Syntax**
```
calMgr.Deembed (cs1,cs2,s2p,port, compPwr,extrap)
```

**Variable (Type) - Description**
- **calMgr** *(object)* - A CalManager object
- **cs1** *(String)* - Name of an existing Cal Set which resides on the VNA.
- **cs2** *(String)* - Name of new Cal Set which contains updated error terms with fixture de-embedded.
- **s2p** *(String)* - Name of the S2P file which characterizes the adapter/fixture.
- **port** *(Long Integer)* - Port number from which fixture will be de-embedded.
- **compPwr** *(Boolean)*
  - **True** - When the Cal Set contains a power correction array for the fixture port, that array will be compensated for the fixture loss.
  - **False** - Do not compensate for loss in source power through the fixture.
- **extrap** *(Boolean)*
  - **True** - Applies a simple extrapolation when the S2P file has a narrower frequency range than the Cal Set. The values for the first and last data points are extended in either direction to cover the frequency range of the Cal Set.
  - **False** - Extrapolation is NOT performed (default setting).

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**
```
calMgr.Deembed
"MyCalSet","MyNewCalSet","Fixture.s2p",1,True,True
```
C++ Syntax    HRESULT Deembed (BSTR srcSet, BSTR destSet, BSTR s2p, long port, BOOL compPwr, BOOL extrap);

Interface    ICalManager8
# DefineParameter Method

**Description**  
Create a new parameter for Differential IQ channel. Use **CreateCustomMeasurementEx Method** to create a new trace with the new parameter.

Use **Change Parameter Method** to change the parameter of one of the existing traces to the new parameter.

**VB Syntax**  
```vbnet
DIQ.DefineParameter (name, expression)
```

**Variable**  
- **(Type)** - **Description**
  - **DIQ** (A DIQ Object)
  - **name** (String) Parameter name.

**Note:** Do not use underscores in the parameter name. For example, b2_f1 cannot be used as a parameter name. However, b2f1 is a valid parameter name.

**expression** (String) Parameter expression using receiver names and mathematical expressions.

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
```vbnet
diq.DefineParameter "myNewParam","(a1_F1+b1_F2)/c1"
app.ActiveMeasurement.ChangeParameter "myNewParam",1
```

See example program

**C++ Syntax**  
```cpp
HRESULT DefineParameter(BSTR name,BSTR expression);
```

**Interface**  
IDIQ
### Delete Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Deletes the measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>meas.Delete</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>meas</td>
<td>The Measurement object to delete (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>meas.Delete</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT Delete()</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
### DeleteMarker Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Deletes a marker from the measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.DeleteMarker(Mnum)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><code>Mnum</code></td>
<td>(long) - Any existing marker number in the measurement</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meas.DeleteMarker(1)</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT DeleteMarker(long lMarkerNumber)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
# DeleteAllMarkers Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Deletes all of the markers from the measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.DeleteAllMarkers</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>meas</code></td>
<td><em>(object)</em> - The Measurement object from which markers will be deleted.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meas.DeleteAllMarkers</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT DeleteAllMarkers()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IMeasurement</code></td>
</tr>
</tbody>
</table>
About Mixer/Converter Settings

DeleteAllSegments Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Removes all segments from the scratch mixer.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>conv.DeleteAllSegments()</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td></td>
<td><code>conv</code> A Converter Object</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mxr.DeleteAllSegments()</code></td>
</tr>
<tr>
<td></td>
<td>See example program</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT DeleteAllSegments()</td>
</tr>
<tr>
<td>Interface</td>
<td>IConverter5</td>
</tr>
</tbody>
</table>
DeleteCalSet Method

**Description**
Deletes a Cal Set from the set of available Cal Sets. This method immediately updates the Cal Set file on the hard drive. If the Cal Set is currently being used by a channel or does not exist, this request will be denied and an error is returned.

Using the Cal Sets collection is a convenient way to manage Cal Sets.

**VB Syntax**
```vbnet
calMgr.DeleteCalSet (calset)
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>calMgr</code></td>
<td>A CalManager object</td>
</tr>
<tr>
<td><code>calset</code></td>
<td>Cal Set to be deleted. Specify the Cal Set by GUID or Name. Use EnumerateCalSets to list the available Cal Sets by name.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**
```vbnet
Set pna=CreateObject("AgilentPNA835x.Application")
Set cmgr = pna.GetCalManager
cmgr.DeleteCalSet ("MyCalSet")
```

**C++ Syntax**
```cpp
HRESULT DeleteCalSet( BSTR strCalset);
```

**Interface**
ICalManager
### DeleteConfiguration Method

**Description**
Deletes the specified configuration name from the VNA. The factory configurations cannot be deleted. This is the only method of programmatically distinguishing a factory configuration from a user-named configuration.

**VB Syntax**
```
pathMgr.DeleteConfiguration name
```

**Variable**
*Type* - Description

- `pathMgr` PathConfigurationManager *(object)*
- `name` *(String)* Configuration name to be deleted.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
path.DeleteConfiguration "myMixer"
```

**C++ Syntax**
```
HRESULT StoreConfiguration (long channelNum, BSTR configName);
```

**Interface**
IPathConfigurationManager
### DeleteParameter Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Deletes the named parameter.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>DIQ.DeleteParameter(name)</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>DIQ</code></td>
<td>A DIQ Object</td>
</tr>
<tr>
<td><code>name</code></td>
<td>(String) Parameter name that was used when the parameter was created.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>diq.DeleteParameter &quot;myNewParam&quot;</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT DeleteParameter (BSTR name);</td>
</tr>
<tr>
<td>Interface</td>
<td>IDIQ</td>
</tr>
</tbody>
</table>
### DeleteSegment Method

**Description**  
Removes the specified number of segments from the scratch mixer starting at the index position.

**VB Syntax**  
`conv.DeleteSegment index,count`

**Variable** *(Type)* - Description
- **conv**  
  A Converter Object
- **index**  
  (Long integer) Position at which to start removing segments. Valid index range is between 1 and the current segment count. Use `SegmentCount Property` to read the current count in the Applied Mixer.
- **count**  
  (Long integer) Optional argument. Number of segments to remove. If unspecified, 1 segment is removed.

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`mxr.DeleteSegment 1,5 'Removes 5 segments beginning at the first position.`

See example program

**C++ Syntax**  
`HRESULT DeleteSegment(long index, long count);`

**Interface**  
IConverter5
# DeleteShortCut Method

**Description**  
Removes a macro from the list of macros in the analyzer. Does not remove the file.

**Note:** There are always 25 macro positions. They do not have to be sequential. For example, you can have number 7 but not numbers 1 to 6.

<table>
<thead>
<tr>
<th>VB Syntax</th>
<th>app.DeleteShortCut item</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Variable</strong></td>
<td></td>
</tr>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
<tr>
<td>item</td>
<td>(long integer) number of the macro to be deleted.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>app.DeleteShortCut 2</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT DeleteShortcut(long Number )</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>
### DisallowAllEvents Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets event filtering to monitor NO eventst.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.DisallowAllEvents</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>app.DisallowAllEvents</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT DisallowAllEvents()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>
### DiscardChanges Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Cancels changes that have been made to the Converter setup and reverts to the previously-saved setup. Same as the <strong>Cancel</strong> button on the mixer setup dialog box.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>conv.DiscardChanges</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>conv</code></td>
<td>A Converter Object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>conv.DiscardChanges</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT DiscardChanges();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IConverter</td>
</tr>
</tbody>
</table>
About Cal Window

DisplayNAWindowDuringCalAcquisition Method

**Description**
Set the 'show' state of the window to be displayed during a calibration.

When this command is sent, the specified window is 'flagged' to be shown during calibration. The flag is cleared when the window is closed. A Preset or Instrument State Recall also closes the window. If the same window number is reopened, this command must be sent again to show the window during a calibration. The flag is NOT saved with an instrument state.

Send this command for each additional window to show during a calibration.

**VB Syntax**
```
calMgr.DisplayNAWindowDuringCalAcquisition (winNum, State)
```

**Variable**
- **(Type)** - Description
  - *calMgr* (object) - A CalManager object
  - *winNum* (long) - Window number to show during a calibration. The calibration window will also be shown with this window.

  The window must already be created.

  Use `NaWindows.count` or `app.WindowNumber` to read existing window numbers.

  - **state** (Boolean) Window state. Choose from:
    - **True** - Show the specified window during calibration.
    - **False** - Do NOT show the specified window during calibration.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**
```
calMgr.DisplayNAWindowDuringCalAcquisition 2,True
```

See example using this command

**C++ Syntax**
```
HRESULT DisplayNAWindowDuringCalAcquisition( long WinNum, VARIANT_BOOL bVal);
```

**Interface**
ICalManager5
### DisplayOnlyCalWindowDuringCalAcquisition Method

**Description**
Clears the flags for windows to be shown during calibrations other than the Cal Window. To flag a window to be shown see `DisplayNAWindowDuringCalAcquisition`.

**VB Syntax**
```
calMgr.DisplayOnlyCalWindowDuringCalAcquisition
```

**Variable**
- **Type** - Description
- **calMgr** - A `CalManager` object

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**
```
calMgr.DisplayOnlyCalWindowDuringCalAcquisition
```

See example using this command

**C++ Syntax**
```
HRESULT DisplayOnlyCalWindowDuringCalAcquisition()
```

**Interface**
`ICalManager5`
**DoPrint Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Prints the screen to the default Printer.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.DoPrint</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>app.DoPrint</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT DoPrint()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IApplication</code></td>
</tr>
</tbody>
</table>
### DoECAL1PortEx Method

**Description**

This method replaces DoECAL1Port Method.

Does a 1-Port calibration using an ECAL module. You must first have a 1-port measurement active to perform the calibration.

The characterization within the ECAL module that will be used for the calibration is specified by ECALCharacterizationEx. The default value is 0.

**VB Syntax**

```vbnet
cal.DoECAL1PortEx [port][,module]
```

<table>
<thead>
<tr>
<th>Variable</th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>cal</td>
<td>object</td>
<td>A Calibrator</td>
</tr>
<tr>
<td>port</td>
<td>long integer</td>
<td>Optional argument - Port number to calibrate. Choose from:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - Calibrate port 1 (default if unspecified)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - Calibrate port 2</td>
</tr>
<tr>
<td>module</td>
<td>long integer</td>
<td>Optional argument. ECAL module.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Choose from modules 1 through 8</td>
</tr>
</tbody>
</table>

Use IsECALModuleFoundEx to determine the number of modules connected to the VNA.

Use GetECALModuleInfoEx to returns the model and serial number of each module.

**Return Type** None

**Default** Not Applicable

**Examples**

```vbnet
cal.DoECAL1PortEx,2,2
```

**C++ Syntax**

```cpp
HRESULT DoECAL1PortEx(long port, long moduleNumber = 1);
```

**Interface** ICalibrator4
**DoECAL2PortEx Method**

**Description**  
This method replaces DoECAL2Port Method.  

Does a 2-port calibration using an ECal module.  

2-port refers to the number of ports to calibrate; NOT to the number of ECal module ports.  

You must first have a measurement active to perform the calibration.  

The characterization within the ECal module that will be used for the calibration is specified by ECalCharacterizationEx. The default value is 0.

**VB Syntax**  
`cal.DoECAL2PortEx [portA],[portB],[module]`

**Variable** *(Type)* - Description  
`cal` A Calibrator *(object)*  

`portA` *(long integer)* Optional argument - Number of the receive port to calibrate.  
Choose from:  
1 - Calibrate port 1 (default, if unspecified)  
2 - Calibrate port 2  
3 - Calibrate port 3  

And so forth for all available VNA / test set ports.  

`portB` *(long integer)* Optional argument - Number of the source port to calibrate.  
Choose from:  
1 - Calibrate port 1  
2 - Calibrate port 2 (default, if unspecified)  
3 - Calibrate port 3  

And so forth for all available VNA / test set ports.  

`module` *(long integer)* Optional argument. ECal module.  
Choose from modules 1 through 8

Use IsECALModuleFoundEx to determine the number of modules connected to the VNA  

Use GetECALModuleInfoEx to returns the model and serial number of each module.
<table>
<thead>
<tr>
<th><strong>Return Type</strong></th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>cal.DoECAL2PortEx,1,2,3</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT DoECAL2PortEx( long portA = 1, long portB =2, long moduleNumber = 1);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalibrator4</td>
</tr>
</tbody>
</table>
## DoneCalConfidenceCheckECAL Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Concludes the Confidence Check and sets the ECal module back into the idle state.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>cal.DoneCalConfidenceCheckECAL</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>cal</code></td>
<td>A Calibrator (<strong>object</strong>)</td>
</tr>
<tr>
<td>Return Type</td>
<td>None</td>
</tr>
<tr>
<td>Default</td>
<td>None</td>
</tr>
<tr>
<td>Examples</td>
<td><code>cal.DoneCalConfidenceCheckECAL</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT DoneCalConfidenceCheckECAL( );</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ICalibrator</td>
</tr>
</tbody>
</table>
**DoReceiverPowerCal Method**

**Description**

*Note: This command replaces DataToDivisor, LogMagnitudeOffset, Normalization, InterpolateNormalization.*

Immediately performs a receiver power calibration. The connection to the receiver must be in place when this command is sent.

A Receiver Power Cal requires that the active measurement be an Unratioed power measurement.

**VB Syntax**

`cal.DoReceiverPowerCal(param, srcPort [,pwrOffset])`

**Variable**

- **cal** *(object)* — A Calibrator
- **param** *(string)* — Receiver to be calibrated. Choose any receiver in your VNA. See a block diagram of your VNA.
- **srcPort** *(long integer)* — Number of the port which will supply source power to the receiver during this cal.

Receivers can also be referred to using **logical receiver notation**. This notation makes it easy to refer to receivers with an external test set connected to the VNA. You do not need to know which physical receiver is used for each test port. Learn more.

- **pwrOffset** *(double)* — Optional argument. Offset value in dB. Adjusts a receiver power cal to account for components or adapters that are added between the source port and receiver while performing this cal. Specify loss as a negative number; and gain as a positive number.

**Return Type**

None

**Default**

Not Applicable

**Examples**

`cal.DoReceiverPowerCal "B", 1, -10`

**C++ Syntax**

`HRESULT DoReceiverPowerCal(BSTR parameter, long lSrcPort, double dPowerOffset);`
Interface  ICalibrator5
# DoResponseCal Method

**Description**
Performs and immediately applies a Response cal. Same as selecting **Normalize** from the Unguided Cal - Measure Standards page. Learn more.

**VB Syntax**
```
cal.DoResponseCal (measParam),(SourcePort)
```

**Variable**
- **Type** - Description
- **cal** A **Calibrator (object)**
- **measParam** (String) Measurement parameter to correct. It is NOT necessary for this measurement to be present.
- **SourcePort** (long integer) Source port number to calibrate. Optional for S-parameter measurements. Choose from:
  0 - N5264B Measurement Receiver (no source ports).
  1 - Calibrate port 1
  2 - Calibrate port 2 (default, if unspecified)
  3 - Calibrate port 3
  And so forth for all available VNA / test set ports.

**Return Type**
None

**Default**
Not Applicable

**Examples**
```
cal.DoResponseCal "A/R",1
```

**C++ Syntax**
```
HRESULT DoResponseCal(BSTR param, long SourcePort);
```

**Interface**
ICalibrator9
**Embed Method**

**Description**
Embeds a fixture (usually a matching network) into an existing Cal Set based on an S2P file. A new Cal Set is created with the effects of the matching network included in the correction data.

When the new Cal Set is applied to a channel, the effects of the fixture are included in the measurement data. Do NOT enable fixturing. The effects of the matching network are included when the new Cal Set is selected and correction is turned ON.

**VB Syntax**
\[ \text{calMgr.Embed } (cs1, cs2, s2p, port, compPwr, extrap) \]

Variable *(Type) - Description*

- **calMgr** *(object) - A CalManager object*
  - *cs1* *(String)* Name of an existing Cal Set which resides on the VNA.
  - *cs2* *(String)* Name of new Cal Set which contains updated error terms with fixture embedded.
  - *s2p* *(String)* Name of the S2P file which characterizes the adapter/fixture.
  - *port* *(Long Integer)* Port number from which fixture will be embedded.
  - *compPwr* *(Boolean)*
    - **True** - Increase the source power to compensate for the loss through the fixture. The result is that the specified power level will be correct at the DUT input.
    - **False** - Do not compensate for loss in source power through the matching network.
  - *extrap* *(Boolean)*
    - **True** - Applies a simple extrapolation when the S2P file has a narrower frequency range than the Cal Set. The values for the first and last data points are extended in either direction to cover the frequency range of the Cal Set.
    - **False** - Extrapolation is NOT performed (default setting).

**Return Type** Not Applicable
<table>
<thead>
<tr>
<th>Default</th>
<th>Not Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example</td>
<td><code>calMgr.Embed &quot;MyCalSet&quot;,&quot;MyNewCalSet&quot;,&quot;Fixture.s2p&quot;,1,True,True</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT Embed (BSTR srcSet, BSTR destSet,BSTR s2p, long port, BOOL compPwr, BOOL extrap);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ICalManager8</td>
</tr>
</tbody>
</table>
### ENREmbedAdapter Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Generate a new ENR file by embedding an adapter to an existing ENR file.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>calMgr.ENREmbedAdapter (inENR, s2p, outEnr)</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>calMgr</code> (object)</td>
<td>A CalManager object</td>
</tr>
<tr>
<td><code>inENR</code> (String)</td>
<td>Path and filename of an existing ENR file</td>
</tr>
<tr>
<td><code>s2p</code> (String)</td>
<td>Path and filename of an s2p file of adapter.</td>
</tr>
<tr>
<td><code>outENR</code> (String)</td>
<td>Path and filename of an ENR file to output</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>ICal Set Interface</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>calMgr.ENREmbedAdapter (D:\Original.enr&quot;,&quot;D:\adapter.s2p&quot;,&quot;D:\new.enr&quot;)</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT ENREmbedAdapter (BSTR inEnr, BSTR s2p, BSTR outEnr);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalManager11</td>
</tr>
</tbody>
</table>
## EnumerateCalSets Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns an array of Cal Set names being stored on the PNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>value = calMgr.EnumerateCalSets</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(variant) - Variable to store the returned Cal Set names</td>
</tr>
<tr>
<td>calMgr</td>
<td>(object) - A CalManager object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>VARIANT array</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>Dim pna set</td>
</tr>
<tr>
<td></td>
<td>pna=CreateObject(&quot;AgilentPNA835x.Application&quot;)</td>
</tr>
<tr>
<td></td>
<td>Dim catalog</td>
</tr>
<tr>
<td></td>
<td>catalog=pna.getcalmanager.EnumerateCalSets</td>
</tr>
<tr>
<td></td>
<td>for i=lboud(catalog) to Ubound(catalog)</td>
</tr>
<tr>
<td></td>
<td>wscript.echo catalog(i)</td>
</tr>
<tr>
<td></td>
<td>next</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT EnumerateCalSets(VARIANT* names);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalManager4</td>
</tr>
</tbody>
</table>
### EnumerateItems Method

**Description**
Returns a list of all name-value pairs (items) in the Cal Set.

**See Also**
- Item Property (Learn about Name-Value pairs.)
- RemoveItem Method

**VB Syntax**
```vbnet
names = CalSet.EnumerateItems
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>names</td>
<td>(Variant array) List of string names.</td>
</tr>
<tr>
<td>CalSet</td>
<td>(object) - A CalSet object</td>
</tr>
</tbody>
</table>

**Return Type**
Variant

**Default**
Not Applicable

**Examples**
See example

**C++ Syntax**
```cpp
HRESULT EnumerateItems (VARIANT* itemNames);
```

**Interface**
ICalSet6
**Execute Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Allows the use of COM to send a SCPI command.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This method can be used with :SYST:ERR? to convert scpi errors into text.</td>
</tr>
<tr>
<td></td>
<td><strong>See an example</strong> of how to return error information when using the Parse method.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>The SCPIStringParser Methods can NOT be used with SCPI Status Reporting. However, the *OPC? will work.</td>
</tr>
</tbody>
</table>

**VB Syntax**

Scpi.Execute *(SCPI_Command)*

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>scpi</em></td>
<td>A ScpiStringParser <em>(Object)</em></td>
</tr>
</tbody>
</table>

**SCPI_Command**

| (String) | Any valid SCPI command |

**Return Type**

String

**Default**

Not Applicable

**Examples**

```
Dim scpi As ScpiStringParser
Set scpi = app.ScpiStringParser
scpi.Execute("SYST:PRES");
ErrorString = scpi.Execute("SYST:ERROr");
```

**C++ Syntax**

Execute(BSTR SCPI_Command, BSTR * pQueryResponse);

**Interface**

IScpiStringParser2
## ExecuteShortcut Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Executes a Macro (shortcut) stored in the analyzer. Use <code>app.getShortcut</code> to list existing macros. Use <code>app.putShortcut</code> to associate the macro number with the file.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.ExecuteShortcut index</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><code>index</code></td>
<td>(long integer) - Number of the macro stored in the analyzer.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>app.ExecuteShortcut 1</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT <code>ExecuteShortcut</code> (long <code>index</code>)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IAApplication</td>
</tr>
</tbody>
</table>


## Exists Method

**Description**
Returns whether or not the specified Cal Set exists on the VNA.

**VB Syntax**
```
calsets.Exists (string)
```

**Variable**
- **calsets**: A Calsets (collection)
- **string**: (String) Name or GUID of the Cal Set enclosed in quotes.

**Return Type**
Boolean

- **True** - Cal Set exists
- **False** - Cal Set does NOT exist

**Default**
Not Applicable

**Examples**
```
dim check
check=calsets.Exists ("MyCalset")
or
check=calsets.Exists ("7C4EEA5E-40D2-4D70-A048-33BFFE704163")
```

**C++ Syntax**
```
HRESULT Exists(BSTR nameOrGuid, VARIANT_BOOL * exists)
```

**Interface**
ICalSets2
# ExportCSVfile Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Exports segment data to a CSV file.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>segs.ExportCSVfile(bstrFile)</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>segs</code></td>
<td>A Segments collection (object)</td>
</tr>
<tr>
<td><code>bstrFile</code></td>
<td>(string) - Full path, file name, and extension of the file.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>segs.ExportCSVfile(&quot;D:\MyFile.csv&quot;)</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT ExportCSVfile(BSTR bstrFile)</td>
</tr>
<tr>
<td>Interface</td>
<td>ISegments6</td>
</tr>
</tbody>
</table>

Find Method

Description
Returns a list of either VISA address strings or VISA address aliases.

VB Syntax
value = Vpassthru.Find (regex, findmode)

Variable (Type) - Description

value (Variant) - Variable to store the list of addresses or aliases.

Vpassthru (object) - A VISAPassthrough object

regex (String) VISA regular expressions are expressions defined by the user to find devices that have been set up on the VISA interface. The following are examples of VISA regular expressions:

<table>
<thead>
<tr>
<th>Interface</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPIB</td>
<td>GPIB[0-9]*::?*INSTR</td>
</tr>
<tr>
<td>PXI</td>
<td>PXI?*INSTR</td>
</tr>
<tr>
<td>VXI</td>
<td>VXI?*INSTR</td>
</tr>
<tr>
<td>GPIB-VXI</td>
<td>GPIB-VXI?*INSTR</td>
</tr>
<tr>
<td>GPIB and GPIB-VXI</td>
<td>GPIB?*INSTR</td>
</tr>
<tr>
<td>All VXI</td>
<td>?<em>VXI[0-9]</em>::?*INSTR</td>
</tr>
<tr>
<td>ASRL</td>
<td>ASRL[0-9]*::?*INSTR</td>
</tr>
<tr>
<td>All</td>
<td>?<em>INSTR or ?</em></td>
</tr>
</tbody>
</table>

Note that using "INSTR" in the VISA regular expression finds "instruments." To search all interfaces, use "?*".

findmode (enum as NAVISAFindMode) - Find mode determines the type of return values. Choose from:

0 - naAddresses: return VISA address names

1 - naAliases: return VISA address aliases

Note: The list of aliases may have lessor more entries than the list of addresses because not all addresses will have aliases, and one address can have more than one alias.

Return Type Variant
Default Address

Examples

C++ Syntax HRESULT Find(string reg_expression, enum NAVISAFindMode)

Interface IVISAPassthrough

value=Vpassthru.Find("?*",1) 'Finds list of aliases
## FrequencyAutoTune Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Auto tunes and zooms in on a signal within a SA sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>sa.FrequencyAutoTune</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>sa</code> <strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>sa.FrequencyAutoTune</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT FrequencyAutoTune();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISpectrumAnalyzer4</td>
</tr>
</tbody>
</table>
### FrequencySpanFullMethod

**Description**
Sets the frequency span to the entire frequency span of the analyzer.

**VB Syntax**
`chan.FrequencySpanFull`

**Variable**
`chan` *(object)* - A `Channel` object

**Return Type**
None

**Default**
Maximum frequency span of the analyzer

**Examples**
`chan.FrequencySpanFull`

**C++ Syntax**
`HRESULT FrequencySpanFull();`

**Interface**
`IChannel25`
**GenerateGlobalDeltaMatchSequence Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Initiates a global delta match calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Learn more about Delta match calibration.</td>
</tr>
<tr>
<td></td>
<td>See example of a complete Delta Match calibration.</td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
numSteps = guided.GenerateGlobalDeltaMatchSequence conn, cKit
```

**Variable**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numSteps</td>
<td>Long Integer</td>
<td>Variable to store the returned number of connection steps required by the Global Delta Match Cal.</td>
</tr>
<tr>
<td>guided</td>
<td>GuidedCalibration (object)</td>
<td></td>
</tr>
<tr>
<td>conn</td>
<td>String</td>
<td>Connector Type for port 1.</td>
</tr>
<tr>
<td>cKit</td>
<td>String</td>
<td>Cal Kit for all ports.</td>
</tr>
</tbody>
</table>

**Return Type**

Not Applicable

**Default**

Not Applicable

**Examples**

```vbnet
guided.GenerateGlobalDeltaMatchSequence "APC 3.5 female","85052B"
```

**C++ Syntax**

```cpp
HRESULT GenerateGlobalDeltaMatchSequence(BSTR port_1_conn, BSTR cal_kit, long *num_steps);
```

**Interface**

IGuidedCalibration2
GenerateErrorTerms Method

**Description**  
Generates the error terms for the specified calibration type, stores the error terms in a Cal Set, saves the Cal Set, and returns the Cal Set GUID.

If ALL the data for the cal type has NOT been acquired an error message is returned.

**Note:** The manner in which the calibration is assigned to a Cal Set (Cal Register or User Cal Set) is determined by the setting of `RemoteCalStoragePreference`.

**VB Syntax**  
```vbnet
value = obj.GenerateErrorTerms
```

**Variable**  
<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(String) - Variable to store the returned GUID or error message.</td>
</tr>
<tr>
<td>obj</td>
<td>Any of the following:</td>
</tr>
<tr>
<td></td>
<td>GuidedCalibration (object)</td>
</tr>
<tr>
<td></td>
<td>SMCType (object)</td>
</tr>
<tr>
<td></td>
<td>VMCType (object)</td>
</tr>
</tbody>
</table>

**Return Type**  
String

**Default**  
Not Applicable

**Examples**  
```vbnet
string = SMC.GenerateErrorTerms
```

**C++ Syntax**  
```c++
HRESULT GenerateErrorTerms(BSTR* calsetGUID);
```

**Interface**  
IGuidedCalibration

SMCType

VMCType
# GenerateSteps Method

**Description**

Returns the number of steps required to complete the calibration.

For an ECal User Characterization this generate steps for the ECal User Characterization process. The channel must already be calibrated using the same, or greater number of VNA ports as the ECal module. Also, the VNA ports must begin with Port 1 and use sequential port numbers.

After this command is executed, subsequent commands can be used to query the number of measurement steps, issue the acquisition commands, query the connection description strings, and subsequently complete a User Characterization calibration.

**VB Syntax**

```vbnet
value = obj.GenerateSteps
```

**Variable**

- **value** (long) - Variable to store the returned number of steps
- **obj** Any of the following:
  - `GuidedCalibration` (object)
  - `SMCType` (object)
  - `VMCType` (object)
  - `ECalUserCharacterizer` (object)

**Return Type**

Long

**Default**

Not Applicable

**Examples**

```vbnet
value = SMC.GenerateSteps
```

**C++ Syntax**

```cpp
HRESULT put_GenerateSteps(long* steps);
```

**Interface**

- `IGuidedCalibration`
- `SMCType`
- `VMCType`
- `IECalUserCharacterizer`
### GetAllSegments Method

**Description**  
Downloads a segment table from the VNA.

**VB Syntax**  
```vbnet
segdata = Segs.GetAllSegments
```

**Variable**  
- **(Type)** - Description  
- `segs` A `Segments (Collection)`  
- `segdata` (Variant) A 2-dimensional array of Segment data:
  
  - Dimension 0 is the number of elements in each segment.
  - Dimension 1 is the number of segments that will be used.

All elements in the returned array are Variant. The type inside each Variant will be as is listed below.

The returned array will contain values for all elements regardless of the settings of `IFBandwidthOption`, `SweepTimeOption`, `SourcePowerOption` and `CouplePorts` properties. Ignore the values for the properties that are set to false.

The following is a list of dimension 0 elements for each segment:

- 0 = Segment state (Boolean True or False)
- 1 = Number of Points in this segment (Integer)
- 2 = Start Freq (Double)
- 3 = Stop Freq (Double)
- 4 = IFBW (Double)
- 5 = Dwell Time (Double)
- 6 + N = Power (Double) where N is the number of source ports of the VNA. For example, with a 4-port, 1-source VNA, indices 6 through 9 correspond to the per-segment power levels for Ports 1 to 4. Use `SourcePortCount Property` and `SourcePortNames Property` to see the available source ports for the VNA.

**Return Type**  
Variant, containing an array.

**Default**  
Not Applicable

**Examples**  
See a VB example using this command
| **C++ Syntax** | HRESULT GetAllSegments (VARIANT *pSegments ); |
| **Interface**   | ISegments5 |
### GetAuxIO Method

**Description**  
This method returns the **IAuxIO** interface.

**VB Syntax**  
```vbnet
app.GetAuxIO
```

**Variable** *(Type) - Description*

- **app**  
  An **Application** *(object)*

**Return Type**  
**IHWAuxIO**

**Default**  
Not Applicable

**Example**
```vbnet
Dim app As AgilentPNA835x.Application
Dim aux As IHWAuxIO
Set aux = app.GetAuxIO
```

**C++ Syntax**  
```cpp
HRESULT GetAuxIO (IHWAuxIO **pAux);
```

**Interface**  
**IApplication**
## GetCalStandard Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a handle to a calibration standard for modifying its definitions. To select a standard for performing a calibration (use Calibrator.AquireCalStandard).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>calkit.GetCalStandard(index)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>calkit</code></td>
<td>A calKit <em>(object)</em></td>
</tr>
<tr>
<td><code>index</code></td>
<td><em>(long)</em> - Number of calibration standard. Choose 1 to 30; (there are 30 cal standards in every kit).</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>calStandard</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
| **Examples** | Dim short As CalStandard  
Set short = calKit.getCalStandard(1)  
short.label = "myShort" |
| **C++ Syntax** | HRESULT GetCalStandard(long standardNumber, ICalStandard **pCalStd) |
| **Interface** | ICalKit |
**GetCalKitTypeString Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns ECal module model number and serial number based on the index number of the attached ECal modules.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>[ECalID = cal.GetCalKitTypeString (module)]</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>ECalID</td>
<td>(string) - variable to store the returned ECal module ID information.</td>
</tr>
<tr>
<td>cal</td>
<td>A Calibrator (object)</td>
</tr>
<tr>
<td>module</td>
<td>(long integer) ECal module.</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>[info = cal.GetCalKitTypeString(2)]</td>
</tr>
</tbody>
</table>

**Example return string:**

"N4691-60003 ECal 01234"

**C++ Syntax**

```cpp
HRESULT GetCalKitTypeString(long moduleNumber, BSTR* info);
```

**Interface**

ICalibrator8
GetCompatibleCalKits Method

**Description**

**Note:** This command replaces CompatibleCalKits Property

Returns a comma-separated list of valid kits that use the specified connector type. This includes mechanical cal kits, applicable characterizations found within ECal modules currently connected to the VNA, and all user characterizations stored in VNA disk memory.

For ECal modules, the returned list includes the serial numbers.

See the ECalUserCharacterizer Object.

Use items in the list to select the kit to be used with the CalKitType Property

**VB Syntax**

```vbnet
value = guidCal.GetCompatibleCalKits (connectorType)
```

**Variable** (Type) - Description

- **value** (Variant) - Variable to store the returned list of cal kits. One-dimensional array of string values.
- **guidCal** A GuidedCalibration (object)
- **connectorType** (String) Connector type for which compatible cal kits will be returned.

Use ValidConnectorType to return a list of connector type strings.

Use ConnectorType to set the connector type for each port to be calibrated.

**Return Type**

Variant – Containing one-dimensional array of strings.

**Default**

Not Applicable

**Examples**

```vbnet
Dim kits As Variant
kits = guidedCal.GetCompatibleCalKits "Type N (50) male"
```

**C++ Syntax**

```c
HRESULT GetCompatibleCalKits(BSTR connector, VARIANT* Kits);
```

**Interface**

IGuidedCalibration5
### GetCalManager Method

<table>
<thead>
<tr>
<th>Description</th>
<th>This method returns the <strong>ICalManager</strong> interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>app.GetCalManager()</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>ICalManager*</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Example</td>
<td><code>dim app as AgilentPNA835x.Application</code>&lt;br&gt;   <code>dim mgr as CalManager</code>&lt;br&gt;   <code>set mgr = app.GetCalManager()</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT GetCalManager( ICalManager **mgr);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IApplication</td>
</tr>
</tbody>
</table>
## Get CalSetByGUID Method

**Description**
Requests a Cal Set by GUID. Returns an ICal Set interface.

**VB Syntax**
`calMgr.GetCalSetByGUID (GUID)`

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>calMgr</code></td>
<td>(object) - A CalManager object</td>
</tr>
<tr>
<td><code>GUID</code></td>
<td>(string) - GUID of the Cal Set being requested.</td>
</tr>
</tbody>
</table>

**Return Type**
Interface object

**Default**
Not Applicable

**Example**
`calMgr.GetCalSetByGUID (2B893E7A-971A-11d5-8D6C-00108334AE96)`

**C++ Syntax**
```cpp
HRESULT GetCalSetByGUID( BSTR* strGUID, ICal Set* pCalSet);
```

**Interface**
ICalManager
### GetCalSetCatalog Method - Superseded

**Description**
This method is replaced with **EnumerateCalSets**

Returns a string containing a list of comma-separated GUIDs in the following format:

- `{FD6F863E-9719-11d5-8D6C-00108334AE96},
- `{1B03B2CE-971A-11d5-8D6C-00108334AE96},
- `{2B893E7A-971A-11d5-8D6C-00108334AE96}

**VB Syntax**

```
value = calMgr.GetCalSetCatalog
```

**Variable (Type) - Description**

- `value` *(string)* - Variable to store the returned GUID list
- `calMgr` *(object)* - A CalManager object

**Return Type** String

**Default** Not Applicable

**Example**

```
value = calMgr.GetCalSetCatalog
```

**C++ Syntax**

```
HRESULT GetCalSetCatalog( BSTR);  
```

**Interface** ICalManager
### GetCalSetUsageInfo Method

**Description**

Returns a string identifying the Cal Set currently in use by the specified channel.

This method identifies the Cal Set being used by returning its GUID.

This method also identifies the "Error Term set" within the Cal Set.

Error term sets are identified by integers, with set 0 belonging to the original (non-interpolated) terms. As stimulus values for a channel are changed causing interpolation to be required, a new Error Term set is constructed within the Cal Set to hold the interpolated Error Terms. The sets are sequentially numbered 1, 2, 3, and so forth. These Error Term sets are destroyed when they are no longer being used.

If there is no Cal Set in use for the given channel, the <GUID> argument is set to the empty string.

**VB Syntax**

```vbnet
calMgr.GetCalSetUsageInfo chan, GUID, setNumber
```

**Variable (Type) - Description**

- **calMgr**: `object` - A CalManager object
- **chan**: `long` - channel of the Cal Set being requested
- **GUID**: `string` - variable to store the GUID of the Cal Set being requested. If there is no Cal Set in use for the given channel, the <GUID> argument is set to the empty string.
- **setNumber**: `long` - variable to store the error term ID being requested. If the returned argument is greater than 0, the set is being interpolated.

**Return Type**

String, Long Integer

**Default**

Not Applicable

**Example**

```vbnet
calMgr.GetCalSetUsageInfo 1,GUID,EtermID
```

**C++ Syntax**

```cpp
HRESULT GetCalSetUsageInfo (long lChannel, BSTR* CalSetGUID, long* etermSetID);
```

**Interface**

ICalManager
**GetCalTypes Method**

**Description**
Returns a list of available calibration types known to the VNA. The Standard CalTypes are the same on all VNA's, but the Custom CalTypes are not necessarily the same. They are dependent on the custom measurement in the VNA. Learn more about applying Cal Types.

See also CalibrationTypeID to apply a Cal Type containing in a Cal Set.

**VB Syntax**

```vbnet
v = mgr.GetCalTypes
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mgr</td>
<td>A CalManager (Object)</td>
</tr>
<tr>
<td>v</td>
<td>Name/GuidPair that contains the calibration type name and associated GUID for each cal type known to the VNA.</td>
</tr>
</tbody>
</table>

**Return Type** (variant) Two dimensional array.

**Default** Not Applicable

**Examples**

```vbnet
v = CalManager.GetCalTypes
```

**C++ Syntax**

```c++
HRESULT GetCalTypes( VARIANT * NameGuidPair )
```

**Interface** ICalManager2
GetComplex Method

Description
Retrieves complex data from the specified location.

See also getNAComplex, getData, and getPairedData Methods

VB Syntax
measData.getComplex location, numPts, real(), imag()

Variable (Type) - Description
measData An IArrayTransfer interface which supports the Measurement object

location (enum NADataStore - IArrayTransfer) - Where the data you want is residing. Choose from:

0 - naRawData
1 - naCorrectedData
2 - naMeasResult
3 - naRawMemory
4 - naMemoryResult
5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

numPts (long integer) - Number of data points requested
[out] - specifies number of data elements returned
[in] - specifies the data being requested or the capacity of the arrays

real (single) - Array to store the real values
imag (single) - Array to store the imaginary values

Return Type Single

Default Not Applicable

Examples
Dim real(201) As Single
Dim imag(201) As Single
Dim pts as Integer
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement
measData.getComplex naCorrectedData, pts, real(0), imag(0)

C++ Syntax
IArrayTransfer - HRESULT getComplex(tagNADataStore DataStore, long* pNumValues, float* pReal, float* pImag)
Interface IArrayTransfer
### GetConnectedPhaseReferences Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Reads the ID strings of the phase references that are currently connected to the VNA USB.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>refs = phaseRef.GetConnectedPhaseReferences</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><em>phaseRef</em></td>
<td>A PhaseReferenceCalibration (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>ref = phaseRef.GetConnectedPhaseReferences()</code></td>
</tr>
<tr>
<td><strong>See example program</strong></td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT GetConnectedPhaseReferences()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPhaseReferenceCalibration</td>
</tr>
</tbody>
</table>
### GetConverter Method

**Description**
This method returns a handle to a `Converter` object.

**VB Syntax**
`chan.GetConverter()`

**Variable**
(Type) - Description
- `chan` - A Channel (object)

**Return Type**
IConverter

**Default**
Not Applicable

**Example**
```vbnet
dim app as AgilentPNA835x.Application
set app = createobject("AgilentPNA835x.Application")
dim chan as IChannel
set chan = app.ActiveChannel
dim convert
set convert = chan.GetConverter()
```

**C++ Syntax**
`HRESULT GetConverter( IConverter **obj);`

**Interface**
IChannel17
**GetDataBuffer Method**

**Description**
Retrieves trace data (Y data) from the modulation distortion measurement.

**VB Syntax**
```
data = meas.GetDataBuffer DataBufferName
```

**Variable**
- `data` *(variant)* - Array to store the data.
- `meas` *(object)* - A `Measurement` object

**DataBufferName** *(string)* – Name of the buffer to be read.

**Note:** The following names assume a 2-port VNA. Also, by default, the parameter names assume that VNA Port 1 is connected to the DUT input and VNA Port 2 is connected to the DUT output. If the RF Path settings are changed from the default, the names will change accordingly. For example, if the VNA has 4 ports and Port 4 is connected to the DUT output, the names will change to "PIn1", "POut4", "S11", "S41", etc.

Choose from:

"POut2" - Power Out

"PIn1" - Power In

"MSig2" - Modulation Signal Out

"MDist2" - Modulation Distortion Out

"MGain21" - Modulation Gain

"MComp21" - Modulation Compression

"PGain21" - Power Gain

"S11" - Linear Input Match

"S21" - Linear Gain

"LPin1" - Linear Power In

"LPOut1" - Linear Reflected Power In
<table>
<thead>
<tr>
<th><strong>Return Type</strong></th>
<th>Variant array</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meas.GetDataBuffer &quot;S21&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT GetDataBuffer(BSTR DataBufferName, VARIANT* Data );</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement19</td>
</tr>
</tbody>
</table>
GetDataBufferCompact Method

**Description**
Retrieves compact signal trace data (Y data) from the modulation distortion measurement.

**VB Syntax**
\[ data = meas.GetDataBufferCompact DataBufferName \]

**Variable**
(Type) - Description

- **data** (variant) - Array to store the data.
- **meas** (object) - A Measurement object

**DataBufferName** (string) – Name of the buffer to be read.

**Note:** The following names assume a 2-port VNA. Also, by default, the parameter names assume that VNA Port 1 is connected to the DUT input and VNA Port 2 is connected to the DUT output. If the RF Path settings are changed from the default, the names will change accordingly. For example, if the VNA has 4 ports and Port 4 is connected to the DUT output, the names will change to "PIn1", "POut4", "S11", "S41", etc.

Choose from:

- "POut2" - Power Out
- "PIn1" - Power In
- "MSig2" - Modulation Signal Out
- "MDist2" - Modulation Distortion Out
- "MGain21" - Modulation Gain
- "MComp21" - Modulation Compression
- "PGain21" - Power Gain
- "S11" - Linear Input Match
- "S21" - Linear Gain
- "LPin1" - Linear Power In
"LPOut1" - Linear Reflected Power In

"LPOut2" - Linear Power Out

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Variant array</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>meas.GetDataBufferCompact &quot;S21&quot;</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT GetDataBufferCompact(BSTR DataBufferName, VARIANT* Data);</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement19</td>
</tr>
</tbody>
</table>
getDataByString Method

**Description**
Retrieves variant data from the specified location in your choice of formats.

The VNA returns complex trace data which is ratioed if required by the measurement parameter, such as S11 or A/B. Otherwise it is raw receiver data, such as A or B.

**Equation Editor Notes:**

- When equation editor is active on a trace in a standard S-parameter channel, GetData returns the data from the parameter on the trace that was measured last. For example, for the equation "S22 + S33 + S11", then S33 is the last measured parameter because it uses source port 3.

- In applications, if equation editor is active and the original parameter for the trace is not requested anywhere in the channel, then zeros are returned. If the original parameter is being measured within the channel, then data for the original parameter is returned.

- In general, if an equation contains no measurement parameters, then data for the original parameter is returned.

**VB Syntax**

```vbnet
data = meas.getDataByString location, format```

**Variable**

- **data** (variant) - Array to store the data.
- **meas** (object) - A Measurement object
- **location** (string) – Name of the buffer to be read. Choose from:
  - "naRawData"
  - "naCorrectedData"
  - "naMeasResult"
  - "naRawMemory"
  - "naMemoryResult"
  - "naDivisor" - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

See Data Access Map
format (enum NADataFormat) - Format in which you would like the data. It does not have to be the displayed format. Choose from:

0 - naDataFormat_LinMag
1 - naDataFormat_LogMag
2 - naDataFormat_Phase
3 - naDataFormat_Polar
4 - naDataFormat_Smith
5 - naDataFormat_Delay
6 - naDataFormat_Real
7 - naDataFormat_Imaginary
8 - naDataFormat_SWR
9 - naDataFormat_PhaseUnwrapped
10 - naDataFormat_InverseSmith
11 - naDataFormat_Kelvin
12 - naDataFormat_Fahrenheit
13 - naDataFormat_Centigrade

Learn more about Data Format.

* Specify Smith or Polar formats to obtain complex data pairs, which require a two-dimensional array varData (numpts, 2) to accommodate both real and imaginary data.

All scalar formats return a single dimension varData(numpts).

Return Type Variant array
Default Not Applicable
Examples meas.getDataByString "naMeasResult", naDataFormat_Phase
C++ Syntax HRESULT getDataByString( BSTR location, tagDataFormat dataFormat, VARIANT * pData );
Interface IMeasurement
**GetData Method**

**Description**
Retrieves variant data from the specified location in your choice of formats. To get smoothed data from any of the specified locations, the format must be the same as the displayed format.

The VNA returns complex trace data which is ratioed if required by the measurement parameter, such as S11 or A/B. Otherwise it is raw receiver data, such as A or B.

This method returns a variant which is less efficient than methods available on the IArrayTransfer interface.

If you plan to **Put** this data back into analyzer, **putDataComplex** (variant data) method requires complex, two-dimensional data. Therefore, request the data in **Polar** format.

**Equation Editor Notes:**

- When equation editor is active on a trace in a standard S-parameter channel, **GetData** returns the data from the parameter on the trace that was measured last. For example, for the equation \( S22 + S33 + S11 \), then S33 is the last measured parameter because it uses source port 3.

- In **applications**, if equation editor is active and the original parameter for the trace is not requested anywhere in the channel, then zeros are returned. If the original parameter is being measured within the channel, then data for the original parameter is returned.

- In general, if an equation contains no measurement parameters, then data for the original parameter is returned.

**VB Syntax**

```vbnet
data = meas.GetData(location, format)
```

**Variable** *(Type) - Description*

- **data**: Variant array to store the data.
- **meas**: A Measurement *(object)*
- **location**: *(enum NADataStore)* - Where the data you want is residing. See Data Access Map. Choose from:
  - 0 - naRawData
  - 1 - naCorrectedData
5 - naDivisor When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

*format (enum NADataFormat) - Format in which you would like the data. It does not have to be the displayed format. Choose from:

0 - naDataFormat_LinMag
1 - naDataFormat_LogMag
2 - naDataFormat_Phase
3 - naDataFormat_Polar*
4 - naDataFormat_Smith*
5 - naDataFormat_Delay
6 - naDataFormat_Real
7 - naDataFormat_Imaginary
8 - naDataFormat_SWR
9 - naDataFormat_PhaseUnwrapped
10 - naDataFormat_InverseSmith
11 - naDataFormat_Kelvin
12 - naDataFormat_Fahrenheit
13 - naDataFormat_Centigrade

Learn more about Data Format.

* Specify Smith or Polar formats to obtain complex data pairs, which require a two-dimensional array varData (numpts, 2) to accommodate both real and imaginary data.

All scalar formats return a single dimension varData(numpts).
**Example Data Format**

**naDataFormat_Phase** and **naDataFormat_PhaseUnwrapped** return degrees. However, **putDataScalar** method accepts data in radians (not degrees) and displays in degrees.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Variant array - automatically dimensioned to the size of the data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**Examples**

```vbs
Dim varData As Variant
varData = meas.GetData(naMeasResult, naDataFormat_Phase)
'Print Data
For i = 0 to chan.NumberOfPoints-1
    Print varData(i)
Next i
```

See a C# example.

**C++ Syntax**

```cpp
HRESULT getData(tagNADataStore DataStore, tagDataFormat DataFormat, VARIANT *pData)
```

**Interface**

IMeasurement
GetECALModuleInfoEx Method

Description
This property replaces GetECALModuleInfo Method.

Returns the following information about the connected ECAL module: model number, serial number, connector type, calibration date, min and max frequency.

The characterization within the ECAl module that this information will be read from is specified by ECALCharacterizationEx. The default value is 0.

VB Syntax
moduleInfo = cal.GetECALModuleInfoEx(module)

Variable
moduleInfo (string) - variable to store the module information

cal A Calibrator (object)

module (long integer) ECal module.

Choose from modules 1 through 8

Use IsECALModuleFoundEx to determine the number of modules connected to the VNA

Return Type
String

Default
Not Applicable

Examples
info = cal.GetECALModuleInfoEx(2)

Example return string:

ModelNumber: 85092-60007, SerialNumber: 01386, ConnectorType: N5FN5F, PortAConnector: Type N (50) female, PortBConnector: Type N (50) female, MinFreq: 30000, MaxFreq: 9100000000, NumberOfPoints: 250, Calibrated: July 4 2002

C++ Syntax
HRESULT GetECALModuleInfoEx(long moduleNumber, BSTR* info);

Interface
ICalibrator4
### GetEcalUserCharacterizer Method

**Description**  
This method returns a handle to an ECalUserCharacterizer object.

**VB Syntax**  
`calMgr.GetEcalUserCharacterizer()`

**Variable**  
*(Type) - Description*

- `calMgr`  
  A CalManager Object *(object)*

**Return Type**  
IEcalUserCharacterizer

**Default**  
Not Applicable

**Example**  
```vbnet
Dim mgr as ICalManager
Set mgr = app.GetCalManager
Dim ecalCharacterizer
Set ecalCharacterizer = mgr.GetEcalUserCharacterizer()
```

**C++ Syntax**  
```c++
HRESULT GetEcalUserCharacterizer( IECalUserCharacterizer **obj);
```

**Interface**  
ICalManager6
### GetENRData Method

**Description**
Read the ENR calibration data from VNA memory.

**VB Syntax**

```vbnet
vData = enr.GetENRData()
```

**Variable**

- **vData**
  Variable to store the returned ENR data. Frequency value in Hz, followed by corresponding ENR value in dB.

- **enr**
  An ENRFile (object)

**Return Type**
Variant Array

**Default**
Not Applicable

**Examples**
See example program

**C++ Syntax**

```c++
HRESULT GetENRData(VARIANT vdata);
```

**Interface**
IENRFFile
### GetErrorCorrection Method

**Description**

Reads the error correction state for the channel.

Use **ErrorCorrection Property** to set this value.

When this command returns true, some measurements on the channel MAY not have error correction ON. This is because the Cal Set currently in place may not contain the appropriate calibration data. To read the error correction state for a measurement, use **Error Correction Property**.

**VB Syntax**

```vbnet
bool = chan.GetErrorCorrection
```

**Variable**

*(Type)* - Description

- *boolean*

  *(boolean)* Variable to store the returned value.

  **False** - Error correction has been set OFF

  **True** - Error correction has been set ON

**chan**

A **Channel (object)**

**Return Type**

Boolean

**Default**

About Error Correction

**Examples**

```vbnet
value = chan.GetErrorCorrection
```

**C++ Syntax**

```cpp
HRESULT GetErrorCorrection (VARIANT_BOOL *bState)
```

**Interface**

IChannel8
### GetErrorTerm Method - Superseded

**Description**

Note: This command is replaced by Get ErrorTermByString Method

Retrieves error term data that is used for error correction. The data is complex pairs. Memory for the returned Variant is allocated by the server. The server returns a variant containing a two-dimensional safe Array.

This method returns a variant which is less efficient than getErrorTermComplex on the ICalData interface.

**Note:** When performing an ECal, send SetCalInfoEx Method BEFORE calling GetErrorTerm method.

Learn about reading and writing Calibration data.

#### VB Syntax

```vbnet
data = cal.getErrorTerm term, rcv src
```

#### Variable (Type) - Description

- **data** Variant array to store the data.
- **cal** A Calibrator (object)
- **term** (enum As NaErrorTerm). Choose from:
  - naErrorTerm_Directivity_Isolation
  - naErrorTerm_Match
  - naErrorTerm_Tracking
- **rcv** (long integer) - Receiver Port
- **src** (long integer) - Source Port
<table>
<thead>
<tr>
<th>To get this</th>
<th>Specify these parameters:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Error Term</strong></td>
<td><strong>term</strong></td>
</tr>
<tr>
<td>Fwd Directivity</td>
<td>naET_Directivity Isolation</td>
</tr>
<tr>
<td>Rev Directivity</td>
<td>naET_Directivity Isolation</td>
</tr>
<tr>
<td>Fwd Isolation</td>
<td>naET_Directivity Isolation</td>
</tr>
<tr>
<td>Rev Isolation</td>
<td>naET_Directivity Isolation</td>
</tr>
<tr>
<td>Fwd Source Match</td>
<td>naErrorTerm_Match</td>
</tr>
<tr>
<td>Rev Source Match</td>
<td>naErrorTerm_Match</td>
</tr>
<tr>
<td>Fwd Load Match</td>
<td>naErrorTerm_Match</td>
</tr>
<tr>
<td>Rev Load Match</td>
<td>naErrorTerm_Match</td>
</tr>
<tr>
<td>Fwd Reflection Tracking</td>
<td>naErrorTerm_Tracking</td>
</tr>
<tr>
<td>Rev Reflection Tracking</td>
<td>naErrorTerm_Tracking</td>
</tr>
<tr>
<td>Fwd Trans Tracking</td>
<td>naErrorTerm_Tracking</td>
</tr>
<tr>
<td>Rev Trans Tracking</td>
<td>naErrorTerm_Tracking</td>
</tr>
</tbody>
</table>

**Return Type** Variant

**Default** Not Applicable

**Examples**

```vbnet
Dim varError As Variant
varError = cal.getErrorTerm(naErrorTerm_Tracking, 2, 1)
```

**C++ Syntax**

```c
HRESULT getErrorTerm(tagNAErrorTerm ETerm, long ReceivePort, long SourcePort, VARIANT* pData)
```

**Interface** ICalibrator
GetErrorTerm Method  **Superseded**

**Description**  This command has been replaced with **Get ErrorTermByString**

Returns error term data from the Cal Set. The returned data is complex pairs.

Learn more about **Reading and Writing Cal Data**

See examples of **Reading and Writing Cal Set Data**

**VB Syntax**  
```
data = calSet.getErrorTerm(setNumber, term, rcv, src)
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>data</strong></td>
<td>(Variant) Two-dimensional safe array to store the returned data. Memory for the returned Variant is allocated by the VNA and must be released by client.</td>
</tr>
<tr>
<td><strong>calSet</strong></td>
<td>A Cal Set (object)</td>
</tr>
<tr>
<td><strong>setNumber</strong></td>
<td>(Long) There can be more than one set of error terms in a Cal Set.</td>
</tr>
<tr>
<td><strong>term</strong></td>
<td>(enum As NaErrorTerm2). Choose from:</td>
</tr>
<tr>
<td>0</td>
<td>naET_Directivity  ( rcv = src )</td>
</tr>
<tr>
<td>1</td>
<td>naET_SourceMatch  ( rcv = src )</td>
</tr>
<tr>
<td>2</td>
<td>naET_ReflectionTracking  ( rcv = src )</td>
</tr>
<tr>
<td>3</td>
<td>naET_TransmissionTracking  ( rcv \neq src )</td>
</tr>
<tr>
<td>4</td>
<td>naET_LoadMatch  ( rcv \neq src )</td>
</tr>
<tr>
<td>5</td>
<td>naET_Isolation  ( rcv \neq src )</td>
</tr>
<tr>
<td><strong>rcv</strong></td>
<td>(Long) - Receiver Port</td>
</tr>
<tr>
<td><strong>src</strong></td>
<td><strong>(Long)</strong> - Source Port</td>
</tr>
<tr>
<td>----------</td>
<td>-------------------------</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
| **Examples** | `Dim varError As Variant`  
`varError = CalSet.getErrorTerm(0, naET_TransmissionTracking, 2, 1)` |
| **C++ Syntax** | `HRESULT getErrorTerm(long setID, tagNAErrorTerm2 ETerm, long ReceivePort, long SourcePort, VARIANT* pData)` |
| **Interface** | ICalSet |
GetErrorTermByString Method

Description
Returns error term data from the Cal Set by specifying the string name of the error term.

- Learn more about Reading and Writing Cal Data
- See examples of Reading and Writing Cal Set Data
- See GetCalSetUsageInfo to determine the setNumber.

VB Syntax
```
pdata = calset.GetErrorTermByString(setNumber, errorTerm)
```

Variable (Type) - Description

pdata (Variant) Two-dimensional safe array to store the returned data. Memory for the returned Variant is allocated by the VNA and must be released by client.

**Note:** See also getErrorTermComplexByString on the ICalData3 interface to avoid using the variant data type.

calset A Cal Set (object)

setNumber (Long) Set number of the required Cal Set data.

See GetCalSetUsageInfo to determine the setNumber.

- SetNumber 0 contains the original "primary" set of error terms for a Cal Set.
- SetNumbers > 0 refers to the VNA channel number that contains the error terms. When retrieving channel error terms, Correction must be ON.

The channel error term data contains interpolation, fixturing, and port extension data if each is ON.

- For Balanced Measurements, interpolation, fixturing, and port extensions can be ON independently.
- For Standard S-parameters, to get port extension data, both fixturing and port extensions must be ON.

errorTerm (String) The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)". To determine the string names of error terms, see GetErrorTermList2.
Return Type: Variant
Default: Not Applicable
Examples: See an Example
C++ Syntax: HRESULT GetErrorTermByString (long SetNumber, BSTR bufferName, VARIANT* pdata);
Interface: ICalSet2
GetErrorTermComplex Method  **Superseded**

**Description**

This command has been replaced by GetErrorTermComplexByString

Retrieves error term data from the error correction buffer. The data is in complex pairs.

**Note:** When performing an ECal, send SetCalInfoEx Method BEFORE calling GetErrorTermComplex method.

Learn more about reading and writing Cal Data using COM.

This method exists on a non-default interface. If you cannot access this method, use the GetErrorTerm Method on ICalibrator.

**VB Syntax**

```vbnet
eData.GetErrorTermComplex term, rcv, src, numPts, real(), imag()
```

**Variable**

(Type) - Description

- `eData`  
  An ICalData pointer to the Calibrator object

- `term`  
  (enum NAErrorTerm) - The error term to be retrieved. Choose from:

  - `naErrorTerm_Directivity_Isolation`
  - `naErrorTerm_Match`
  - `naErrorTerm_Tracking`

- `rcv`  
  (long integer) - Receiver Port

- `src`  
  (long integer) - Source Port

- `numPts`  
  (long integer) - on input, max number of data points to return; on output: indicates the actual number of data points returned.

- `real()`  
  (single) - array to accept the **real** part of the error-term. One-dimensional for the number of data points.

- `imag()`  
  (single) - array to accept the **imaginary** part of the error-term. One-dimensional for the number of data points.
<table>
<thead>
<tr>
<th>Error Term</th>
<th>Specify these parameters:</th>
<th>$term$</th>
<th>$rcv$</th>
<th>$src$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fwd Directivity</td>
<td>naET_Directivity Isolation</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Directivity</td>
<td>naET_Directivity Isolation</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Isolation</td>
<td>naET_Directivity Isolation</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Isolation</td>
<td>naET_Directivity Isolation</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Source Match</td>
<td>naErrorTerm_Match</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Source Match</td>
<td>naErrorTerm_Match</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Load Match</td>
<td>naErrorTerm_Match</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Load Match</td>
<td>naErrorTerm_Match</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Reflection Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Reflection Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Trans Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Trans Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

Return Type: Single
Default: Not Applicable
Examples:
```vbnet
ReDim rel(numpts)
ReDim img(numpts)
Dim eData As ICalData
Set eData = chan.Calibrator
eData.getErrorTermComplex naErrorTerm_Directivity_Isolation, 1, 1, 201, rel(0), img(0)
```

C++ Syntax:
```cpp
HRESULT raw_getErrorTermComplex(tagNAErrorTerm ETerm, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pImag)
```

Interface: ICalData
GetErrorTermComplex Method  **Superseded**

Description

This command is replaced with Get ErrorTermComplexByString

Returns error term data from the Cal Set. The data is in complex pairs.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

**Note:** This method exists on a non-default interface. If you cannot access this method, use the GetErrorTerm Method on ICal Set.

VB Syntax

```vbnet
iCalData2.GetErrorTermComplex setNumber, term, rcv, src, numPts, real(), imag()
```

Variable

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>iCalData2</code></td>
<td>An ICalData2 pointer to the Cal Set object</td>
</tr>
<tr>
<td><code>setNumber</code></td>
<td>(Long) There can be more than one set of error terms in a Cal Set.</td>
</tr>
</tbody>
</table>

- `setNumber 0` contains the original set of error terms for a Cal Set.
- `setNumbers > 0` contain Interpolated error terms. Interpolated error terms are generated when interpolation is required and destroyed when no longer used. Learn about Interpolation.
- To determine the setNumber in use by a channel, see GetCalSetUsageInfo

`term` (enum NAEErrorTerm2) - The error term to be retrieved. Choose from:

- 0 - naET_Directivity
- 1 - naET_SourceMatch
- 2 - naET_ReflectionTracking
- 3 - naET_TransmissionTracking
- 4 - naET_LoadMatch
- 5 - naET_Isolation

`rcv` (Long) - Receiver Port

`src` (Long) - Source Port
**numPts** *(Long)* An In/Out parameter.

On the way **in**, you specify the **max** number of values being requested.

On the way **out**, the VNA returns number of values actually returned.

**real()** *(single)* - array to accept the **real** part of the error-term. One-dimensional for the number of data points.

**imag()** *(single)* - array to accept the **imaginary** part of the error-term. One-dimensional for the number of data points.

**Return Type** Single

**Default** Not Applicable

**Examples**
```
dim numpts as long
numpts = ActiveChannel.NumberOfPoints
ReDim r(numpts) ' real part
ReDim i(numpts) ' imaginary part
Dim CalSet as CalSet
set CalSet = pna.GetCalManager.GetCalSetByGUID( txtGUID )
Dim eData As ICalData2
Set eData = CalSet
eData.getErrorTermComplex 0, naET_LoadMatch, 1, 2, numpts, r(0), i(0)
```

**C++ Syntax**
```
HRESULT getErrorTermComplex(long setID, tagNAErrorTerm2 ETerm, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pImag)
```

**Interface** ICalData2
### GetErrorTermComplexByString Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns error term data from the Cal Set by specifying the string name.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Learn more about <a href="#">Reading and Writing Cal Data</a></td>
</tr>
<tr>
<td></td>
<td>See examples of <a href="#">Reading and Writing Cal Set Data</a></td>
</tr>
<tr>
<td>Note:</td>
<td>This method exists on a non-default interface. If you cannot access this method, use GetErrorTermByString</td>
</tr>
</tbody>
</table>

#### VB Syntax

```vbnet
ICalData3.GetErrorTermComplexByString setNumber, errorTerm, numPoints, real(0), imag(0)
```

#### Variable (Type) - Description

- **ICalData3** (Object)
- **setNumber** (Long)
  - There can be more than one set of error terms in a Cal Set.
  - setNumber 0 contains the original set of error terms for a Cal Set.
  - setNumbers > 0 contain Interpolated error terms. Interpolated error terms are generated when interpolation is required and destroyed when no longer used. [Learn about Interpolation](#).
  - To determine the setNumber in use by a channel, see [GetCalSetUsageInfo](#)
- **errorTerm** (String)
  - The string name of error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)".
  - For a list error term string names, use [Get ErrorTermList2](#)
- **numPoints** (Long)
  - An In/Out parameter.
  - On the way in, you specify the max number of values being requested.
  - On the way out, the VNA returns number of values actually returned.
- **real** (Single)
  - The real component of the complex data.
- **imag** (Single)
  - The imaginary component of the complex data.

### Return Type

- Single

### Default

- Not Applicable

### Examples

See example
C++ Syntax

HRESULT GetErrorTermComplexByString(long etermSetID, BSTR bufferName, long* lnumPoints, single* real, single* imag);

Interface

ICalData3
GetErrorTermList Method  **Superseded**

**Description**  
*Note:* This command is replaced by `CalSet.getErrorTermList2`

Returns the list of Error Terms contained in this Cal Set for the CalType specified in the `OpenCal Set` method. Learn more about reading and writing Cal Data using COM.

The list is a comma separated, textual representation of the error terms with the term name followed by the port path in parentheses:

- Term (n, n),
- Term (m,n)

Before calling this method you must open the Cal Set with `OpenCal Set`. If the Cal set is not open, this method returns E_NA_Cal Set_ACCESS_DENIED.

Use `StringToNAErrorTerm2` to convert the list entries to values that can be used with `GetErrorTerm` and `PutErrorTerm`.

*Note:* The port path designation (m n) indicates the ports that contribute to the error being compensated. Directivity, source match and reflection tracking are single port characteristics, designated in this list by (n n) where n equals the port being characterized.

Other terms characterize the interaction between ports. For example, the load match term is describing the match at port (m) while looking into port (n). Thus the notation (m n) indicates the two ports that contribute to the loadmatch error.

**VB Syntax**

```vbnet
CalSet.GetErrorTermList (SetID, count, strList)
```

**Variable (Type) - Description**

- *CalSet* (object) - A Cal Set object
- *SetID* (long) - specifies the error term set to query. Use 0 for the primary set.
- *count* (long) - the number of error terms in the returned list
- *strList* (string) - comma separated list of error terms found in Cal Set

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**

```vbnet
dim count as Integer
dim list as string
OpenCalSet (naCalType_TwoPortSOLT 1, 2)
GetErrorTermList( 0, count, list)
```
CloseCalSet()  

Assuming the cal set contained the full set of error terms for this two-port Cal, the returned list would be:

"Directivity(1 1),SourceMatch(1 1),ReflectionTracking(1 1),TransmissionTracking(2 1),LoadMatch(2 1),Isolation(2 1),Directivity(2 2),SourceMatch(2 2),ReflectionTracking(2 2),TransmissionTracking(1 2),LoadMatch(1 2),Isolation(1 2)"

**C++ Syntax**  
HRESULT GetErrorTermList (long etermSetID, long* count, BSTR* strList);

**Interface**  
ICalSet
GetErrorTermList2 Method

Description
Returns a list of error terms found in the Cal Set containing the specified text filter.

Learn more about Reading and Writing Cal Data
See examples of Reading and Writing Cal Set Data

VB Syntax
`list = CalSet.GetErrorTermList2(setNumber, filter)`

Variable (Type) - Description

`list` (Variant) Variant containing a string array of error term names.

`CalSet` (object) - A CalSet object

`setNumber` (Long) There can be more than one set of error terms in a Cal Set.

- `setNumber 0` contains the original set of error terms for a Cal Set.
- `setNumbers > 0` contain Interpolated error terms. Interpolated error terms are generated when interpolation is required and destroyed when no longer used. Learn about Interpolation.
- To determine the `setNumber` in use by a channel, see `GetCalSetUsageInfo`

`filter` (String) This string is used as a filter so that only the error term names of interest are returned. If the filter is empty, all terms are returned. The string is case-insensitive. Here are some examples:

- "" (empty string)- returns all error term names for the identified Cal Set and `setNumber`
- "TRACKING" - returns all error term names that include the substring "tracking," such as ResponseTracking(S21) and TransmissionTracking(S21)
- "(s21)" - returns all error term names that end with (S21)

Return Type Variant
Default Not Applicable
Examples See an Example
**C++ Syntax**

HRESULT GetErrorTermList2 (long SetNumber, BSTR filter, VARIANT* list)

**Interface**

ICalSet2
### GetErrorTermStimulus Method

**Description**  
Returns the stimulus values over which the specific error term was acquired. For example, with mixer channels, you may get a different set of values for Directivity at the input port versus Directivity at the output port.

- Learn more about [Reading and Writing Cal Data](#)  
- See examples of [Reading and Writing Cal Set Data](#)  
- See [GetCalSetUsageInfo](#) to determine the setNumber.  
- See [PutErrorTermStimulus Method](#).

**VB Syntax**  
`pdata = calset.GetErrorTermStimulus(setNumber, bufferName)`

**Variable**  
<table>
<thead>
<tr>
<th>(Type) - Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pdata</code> (Variant)</td>
<td>The VNA will allocate the memory for the returned variant. The data is returned as a SafeArray of Variant elements. Each element is of VarType double.</td>
</tr>
<tr>
<td><code>calset</code> A Cal Set (object)</td>
<td></td>
</tr>
<tr>
<td><code>setNumber</code> (Long)</td>
<td>Set number of the required Cal Set data. See <a href="#">GetCalSetUsageInfo</a> to determine the setNumber.</td>
</tr>
<tr>
<td><code>bufferName</code> (String)</td>
<td>The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be &quot;Directivity(3,3)&quot;. To determine the string names of error terms, see <a href="#">GetErrorTermList2</a>.</td>
</tr>
</tbody>
</table>

**SetNumber**  
- SetNumber 0 contains the original "primary" set of error terms for a Cal Set.  
- SetNumbers > 0 refers to the VNA channel number that contains the error terms. When retrieving channel error terms, Correction must be ON. The channel error term data may be interpolated and may also be compensating for a fixture if that feature is on.  
  - For Balanced Measurements, interpolation, fixturing, and port extensions can be ON independently.  
  - For Standard S-parameters, to get port extension data, both fixturing and port extensions must be ON.

**bufferName** (String)  
The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)". To determine the string names of error terms, see [GetErrorTermList2](#).

**Return Type**  
Variant
The sequence is:

```csharp
complexData = calset1.GetErrorTermByString(0, BufferName)

frequencyData = calset1.GetErrorTermStimulus(0, BufferName)

// manipulate complex data here

Calset2.PutErrorTermByString(BufferName, manipulatedComplexData)

Calset2.PutErrorTermStimulus(BufferName, frequencyData);
```

See an Example

**C++ Syntax**

```c++
HRESULT GetErrorTermStimulus (long SetNumber, BSTR bufferName,
VARIANT* pdata);
```

**Interface**

ICalSet7
### GetExtendedCalInterface Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns an interface that exposes the properties of Noise Calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>Cal2.GetExtendedCalInterface (interface)</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>Cal2</code></td>
<td>An ICalibrate2 (object)</td>
</tr>
<tr>
<td><code>interface</code></td>
<td>(object) Returns a handle to the specified interface. Choose from '</td>
</tr>
<tr>
<td></td>
<td>&quot;NoiseCal&quot;</td>
</tr>
</tbody>
</table>

**Return Type**

**Default**

**Example**

```vbnet
dim noiseCal
dim noiseCalExtensions
set noiseCal= Get Calmanager?.CreateCustomCalEx("NoiseCal")
set noiseCalExtensions =
noiseCal.GetExtendedCalInterface("INoiseCal")
```

**C++ Syntax**

`HRESULT GetExtendedCalInterface();`

**Interface**

ICalibrate2
### Get ExternalTestSetIO Method

<table>
<thead>
<tr>
<th>Description</th>
<th>This method returns the IExternalTestSetIO interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>app.GetExternalTestSetIO</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>IHWExternalTestSetIO</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Example</td>
<td><code>Dim app As AgilentPNA835x.Application</code></td>
</tr>
<tr>
<td></td>
<td><code>Dim ets As HWExternalTestSetIO</code></td>
</tr>
<tr>
<td></td>
<td><code>Set ets = app.GetExternalTestSetIO</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT GetExternalTestSetIO (IHWExternalTestSetIO</code></td>
</tr>
<tr>
<td></td>
<td><code>**ptestset);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IApplication</td>
</tr>
</tbody>
</table>
GetFilterStatistics Method

Description
Returns all four Filter Statistics resulting from a SearchFilterBandwidth. These statistics are useful for determining the bandwidth, center frequency, loss, and Q factor (Center Frequency / Bandwidth) of resonators. To retrieve individual filter statistics, use meas.FilterCF, meas.FilterBW, meas.FilterLoss, meas.FilterQ properties.

VB Syntax
meas.GetFilterStatistics cf,bw,loss,q

Variable
(Type) - Description
meas A Measurement (object)
cf,bw,loss,q Dimensioned variables to store the returned values

Return Type
(double) cf
(single) bw,loss,q

Default Not Applicable

Examples
' Dimension variables
Dim cf as Double
Dim bw as Single
Dim loss as Single
Dim q as Single

meas.GetFilterStatistics cf,bw,loss,q

C++ Syntax
HRESULT GetFilterStatistics(double* centerFreq, float* bw, float* loss, float* quality)

Interface IMeasurement
GetGuid Method

Description
Returns a string containing the GUID identifying this Cal Set. Each Cal Set is assigned a GUID (global unique ID). GUIDs are used to retrieve and select Cal Sets on the VNA. Learn more about reading and writing Cal Data using COM.

VB Syntax

```
value = CalSet.GetGuid
```

Variable (Type) - Description

- **value** (string) - Variable to store the returned GUID
- **CalSet** (object) - A Cal Set object

Return Type
String

Default
Not Applicable

Examples

```
guid = CalSet.GetGuid 'Read
```

C++ Syntax

```
HRESULT GetGUID( BSTR* pGUIDString);
```

Interface
ICalSet
# InputVoltageEX Property

This command replaces `get InputVoltage` Method

Reads the ADC voltage from the specified location.

## VB Syntax

```vbnet
volts = AuxIO.InputVoltageEX loc
```

## Variable

- **volts** *(double)* - variable to store the return value
- **AuxIO** *(object)* - A Hardware Auxiliary Input / Output object
- **loc** *(Long)* - Location from which to read data.

## For PNA-X models:

- 1 Reads voltage on Analog In 1 port (pin 7).
- 2 Reads voltage on Analog In 2 port (pin 8).
- 3 Reads voltage on GndSens (pin 6).
- 4 Reads voltage on Analog Out 1 port (pin 3).
- 5 Reads voltage on Analog Out 2 port (pin 4).

## For all other VNA models:

- 1 Reads voltage on the Analog IN (pin 14) of the AUX IO connector.
- 4 Reads voltage on Analog Out 1 port (pin 3).
- 5 Reads voltage on Analog Out 2 port (pin 2).

## Return Type

Double

## Default

Not Applicable

## Examples

```vbnet
Dim aux as HWAuxIO
Set aux = PNA.getAuxIO
volts = aux.InputVoltageEX 1
'for PNA-X, read voltage on PowerI/O pin 7
'for all other models, reads voltage on Aux I/O Analog In (pin 14)
```

## C++ Syntax

```c_cpp
HRESULT get_InputVoltageEX (long muxLoc, double* vtVoltage);
```
Interface HWAuxIO2
### Description
Reads a hardware latch that captures high to low transitions on Input1 of the Material Handler IO. Reading the latch causes it to reset and is ready for the next transition. The hardware latch is only capable of capturing one transition per query. Additional transitions are ignored until after the next query.

Momentarily driving Input1 high, then low, causes a transition to be detected and latched.

### VB Syntax
```vbnet
inp1 = handlerIo.get_Input1
```

### Variable (Type) - Description
- **inp1** *(variant)* - A variable to store the return value
- **handlerIo** *(object)* - A HandlerIO object

### Return Type
- **Variant**
  - **0** - a high to low transition occurred at Input1 since the last time it was queried.
  - **1** - no high to low transition occurred.

### Default
Not Applicable

### Examples
```vbnet
input1 = handlerIo.get_Input1 'Read
```

### C++ Syntax
```cpp
HRESULT get_Input1 (VARIANT* Data);
```

### Interface
IHWMaterialHandlerIO
GetIPConfigurationStruct Method

**Description**
Returns an NA_IPConfiguration data structure which contains information about the current status of the VNA’s computer networking configuration. This is the same set of information that is returned in a single string by the LANConfiguration property.

**VB Syntax**
```vbnet
value = app.GetIPConfigurationStruct
```

**Variable**

<table>
<thead>
<tr>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td>(NA_IPConfiguration) Variable to receive the VNA IP (LAN) configuration information.</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
</tbody>
</table>

**Return Type**
NA_IPConfiguration

**Default**
Not Applicable

**Examples**
```vbnet
Dim networkConfigInfo As NA_IPConfiguration
networkConfigInfo = app.GetIPConfigurationStruct()

MsgBox “Host name = ” & networkConfigInfoHostException
MsgBox “Domain name = ” & networkConfigInfo.DomainName
MsgBox “IP address = ” & networkConfigInfo.IPAddress

If Not networkConfigInfo.DHCPEnabled Then
    MsgBox “IP address is static”
Else
    MsgBox “IP address is dynamic”
End If

MsgBox “Subnet mask = ” & networkConfigInfo.SubNet
MsgBox “Gateway = ” & networkConfigInfo.DefaultGateway
MsgBox “Primary DNS server = ” & networkConfigInfo.DNSServer1
MsgBox “Secondary DNS server = ” & networkConfigInfo.DNSServer2
MsgBox “First suffix in DNS suffix search order = ” & networkConfigInfo.DNSSuffix1
MsgBox “Second suffix in DNS suffix search order = ” & networkConfigInfo.DNSSuffix2
MsgBox “Primary WINS server = ” &
```
networkConfigInfo.PrimaryWINSServer

MsgBox "Secondary WINS server = " & networkConfigInfo.SecondaryWINSServer

MsgBox "Network adapter device ID = " & networkConfigInfo.DeviceID

MsgBox "Description of network adapter = " & networkConfigInfo.Description

MsgBox "MAC address = " & networkConfigInfo.MacAddress

C++ Syntax  HRESULT GetIPConfigurationStruct (tagNA_IPConfiguration * pIPConfig);

Interface  IApplication14
**GetIsolationPaths Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Gets the list of paths (port pairings) for which isolation standards will be measured during calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = obj.GetIsolationPaths</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Variant) - Variable to store the returned port paths in pairs. One-dimensional array of Long Integers.</td>
</tr>
<tr>
<td><code>obj</code></td>
<td>Any of the following:</td>
</tr>
</tbody>
</table>
| &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&n...
# GetLibraryFunctions Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the functions in an imported (loaded) DLL.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>functions = equation.GetLibraryFunctions location</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>functions</code></td>
<td>(variant) - Array to store the returned functions.</td>
</tr>
<tr>
<td><code>equation</code></td>
<td>A MeasurementEquation object</td>
</tr>
<tr>
<td><code>location</code></td>
<td>(string) – Full path and filename of the *.dll to be read.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant array</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>functions=equation.GetLibraryFunctions &quot;C:\Program Files(x86)\Keysight\Network Analyzer\UserFunctions\Expansion.dll&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT GetLibraryFunctions( BSTR filename, BSTR* functionList);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurementEquation</td>
</tr>
</tbody>
</table>
GetLicenses Method

Description
Returns the list of licenses. See a list of common licenses.

VB Syntax
app.Licenses(type)

Variable (Type) - Description
app An Application (object)
type (Enum as NALicenseSet) Choose the type of licenses to be recalled:

0 - naValidLicenses - Return a list of licenses which have enabled VNA software features.

1 - naAllLicenses - Return a list of all installed licenses including the ones not related to the VNA software.

2 - nalIgnoredLicenses - Return a list of VNA software licenses which are either invalid or ignored. This can occur when a transportable license is transported to an instrument that does not support the license feature. In addition, this can occur when multiple licenses for the same base feature are installed and only the least restrictive license is used (the more restrictive licenses are ignored). For example, when transporting multiple Spectrum Analyzer licenses to the same instrument, the license with the greatest frequency range is used and the other licenses are ignored.

Note: Licenses not related to the VNA software but installed on the instrument are not reported as ignored when using nalIgnoredLicenses.

Return Type String
Default Not Applicable

Examples
app.Licenses(naAllLicenses)
"N5242B-423,N5242B-020,N5242B-021,N5242B-022,S93029A/B-1FP"

C++ Syntax
HRESULT Licenses(tagNALicenseSet type, BSTR* LicenseString)

Interface IApplication23
### Get MaterialHandlerIO Method

<table>
<thead>
<tr>
<th>Description</th>
<th>This method returns the MaterialHandlerIO interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.GetMaterialHandlerIO</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application (<code>object</code>)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>IHWMaterialHandlerIO</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
| **Example** | `Dim app As AgilentPNA835x.Application`  
`Dim hand As HWMaterialHandlerIO`  
`Set hand = app.GetMaterialHandlerIO` |
| **C++ Syntax** | `HRESULT GetMaterialHandlerIO (IHWMaterialHandlerIO **phand);` |
| **Interface** | IApplication |
GetNAComplex Method

Description
Retrieves complex data from the specified location. See also getComplex and getData Method.

VB Syntax
`measData.getNAComplex location, numPts, data`

Variable (Type) - Description
`measData`  An IArrayTransfer interface which supports the Measurement object
`location`  (enum NADataStore) - Where the data you want is residing. Choose from:
- 0 - naRawData
- 1 - naCorrectedData
- 2 - naMeasResult
- 3 - naRawMemory
- 4 - naMemoryResult
- 5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

`numPts`  (long integer) - Number of data points requested
[out] - specifies number of data elements returned
[in] - specifies the data being requested or the capacity of the dComplex array

`data`  (NAComplex) - A one-dimensional array of NaComplex to store the data.

Return Type
NAComplex

Default
Not Applicable

Examples
Dim dComplex(201) As NaComplex
Dim measData As IArrayTransfer
Dim pts as Long
Set measData = app.ActiveMeasurement
measData.getNAComplex naCorrectedData, pts, dComplex(0)

Notes
The data is stored as Real and Imaginary (Re and Im) members of the NaComplex user defined type. You can access each number individually by iterating through the array.

For i = 0 to NumPts-1
    dReal (i) = dcomplex (i).Re
    dImg (i) = dcomplex (i).Im
C++ Syntax
HRESULT getNAComplex(tagNADataStore DataStore, long* pNumValues, TsComplex* pComplex)

Interface
IArrayTransfer
### GetNumberOfGroups Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of groups a channel has yet to acquire. To set the number of groups for a channel, use <strong>Number Of Groups Method</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = chan.GetNumberOfGroups</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long Integer)</em> - Number of groups</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>Channel <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td><em>(Long Integer)</em></td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>groups = chan.GetNumberOfGroups</code>  <code>Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT GetNumberOfGroups(long* numberOfGroups);</td>
</tr>
<tr>
<td>Interface</td>
<td>IChannel3</td>
</tr>
</tbody>
</table>

---

**About Trigger**

Read-only

2776
## get_Output Method

**Description**  
Type 1 and Type2 configurations: Returns the last value written to the selected output pin.

Type3 configuration: Returns the current state of the selected output pin. If an Input1 trigger occurs, the state may not be the same value as was written.

All configurations: Data is written using put_Output Method.

### VB Syntax

```vb
data = handlerIo.get_Output(pin)
```

### Variable (Type) - Description

- **data** *(variant)*  - A variable to store the return value. The returned value will be one of the following:
  - 0 - TTL Low
  - 1 - TTL High

- **handlerIo** *(object)*  - A HandlerIO object

- **pin** *(enum as NAMatHandlerOutput)*  - output to read. Choose from:
  - naOutput1 (0)
  - naOutput1User (1)
  - naOutput2 (2)
  - naOutput2User (3)

**Learn about User Output**

### Return Type

Variant

### Default

Not Applicable

### Examples

```vb
data = handlerIo.get_Output(naOutput1)
```

### C++ Syntax

```c++
HRESULT get_Output ( tagNAMatHandlerOutput Output, VARIANT* Data );
```

### Interface

IHWMaterialHandlerIO
**get_OutputVoltage Method**

**Description**

**E836x and PNA-L:** Reads voltages on the DAC/Analog Output 1|2 of the Auxiliary IO connector.

**PNA-X:** Reads voltage on the Power I/O connector AnalogOut1|2.

**VB Syntax**

```vbnet
globals = AuxIO
volts = globals.get_OutputVoltage (output)
```

**Variable**

- **volts** *(double)* - variable to store the return value
- **AuxIO** *(object)* - A Hardware Auxiliary Input / Output object
- **output** *(variant)* - Number of the output DAC from which to read voltage. Choose from:
  1. Output 1 (Aux I/O pin 3) and (Power I/O pin 3)
  2. Output 2 (Aux I/O pin 2) and (Power I/O pin 4)

**Return Type**

Double

**Default**

Not Applicable

**Examples**

```vbnet
Dim aux as HWAuxIO
Set aux = PNA.getAuxIO
volts = aux.get_OutputVoltage(1)
' read voltage from Analog Out 1 (Aux I/O pin3) or (Power I/O pin 3)
```

**C++ Syntax**

```c++
HRESULT get_OutputVoltage(VARIANT Output, double* Voltage);
```

**Interface**

IHWAuxIO
get OutputVoltageMode Method

**Description**
This command returns the mode of the selected "Analog Out" line on the Power I/O connector. The modes give the user the option to have the requested voltage applied immediately or not until the sweep is done. To set the mode, use put_OutputVoltageMode Method.

**VB Syntax**
```
mode = auxIo.get_OutputVoltageMode (output)
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode</td>
<td>(enum NAOutputVoltageMode)</td>
<td>variable to store the returned mode.</td>
</tr>
<tr>
<td></td>
<td>naWaitEOS</td>
<td>While in this mode any voltage changes sent to the selected analog out will only get applied to the output between sweeps.</td>
</tr>
<tr>
<td></td>
<td>naNoWait</td>
<td>While in this mode any voltage changes sent to the selected analog out will occur right away without waiting until the end of a sweep, the voltage gets applied immediately.</td>
</tr>
</tbody>
</table>

| auxIo      | (object)       | A Hardware Auxiliary Input / Output object                                   |
| output     | (double)       | Analog Output. Choose from 1 or 2                                            |

**Return Type**
enum as NAOutputVoltageMode

**Default**
naWaitEOS

**Examples**
```
vOutMode = auxIo.get_OutputVoltageMode (1)
```

**C++ Syntax**
```
HRESULT get_OutputVoltageMode(VARIANT Output, 
tagNAOutputVoltageMode* pMode);
```

**Interface**
IHWAuxIO
GetPairedData Method

Description
Retrieves pairs of data from the specified location.

Note: This method exists on a non-default interface. If you cannot access this method, use the Get Data Method on IMeasurement.

VB Syntax
measData.getPairedData location, format, numPts, d1, d2

Variable
(Type) - Description
measData - An IArrayTransfer interface which supports the Measurement object
location (enum NADataStore) - Where the data you want is residing. Choose from:

0 - naRawData
1 - naCorrectedData
2 - naMeasResult
3 - naRawMemory
4 - naMemoryResult
5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

See Data Access Map

format (enum NApairedDataFormat) - Format in which you would like the Paired data. Choose from:

0 - naLogMagPhase - Log magnitude and phase
1 - naLinMagPhase - Linear magnitude and phase
2 - naRealImaginary - Real and Imaginary

Note: Selecting naRealImaginary format is the same as using the getComplex method

numPts (long integer) - Number of data points requested
[out] - specifies number of data elements returned
[in] - specifies the data being requested or the capacity of the dPaired array

d1 (single) - Array to store the magnitude / real values
(single) - Array to store the phase / imaginary values

**Return Type**
Two Single arrays

**Default**
Not Applicable

**Examples**
```vba
Dim logm() As Single
Dim phase() As Single
Public measData As IArrayTransfer
Set measData = app.ActiveMeasurement
Dim numpts As Long
numPoints = app.ActiveChannel.NumberOfPoints
ReDim logm(numPoints)
ReDim phase(numPoints)
measData.getPairedData naCorrectedData, naLogMagPhase,
numPoints, logm(0), phase(0)

Print values(0), values(1)
```

**C++ Syntax**
```c
HRESULT getPairedData(tagNADataStore DataStore,
tagNAPairedDataFormat PairFormat, long* pNumValues, float* pReal, float* pImag)
```

**Interface**
IArrayTransfer
**get_Port Method**

**Description**
Returns the value from the specified "readable" port.

**VB Syntax**
```vbnet
data = handlerIo.get_Port(port)
```

**Variable**

<table>
<thead>
<tr>
<th><strong>data</strong></th>
<th>(variant) - A variable to store the return value. The following table shows what the returned data represents:</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Port</th>
<th>MSB...........................................LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>C3...C0</td>
</tr>
<tr>
<td>D</td>
<td>D3...D0</td>
</tr>
<tr>
<td>E</td>
<td>D3...D0 + C3...C0</td>
</tr>
</tbody>
</table>

**handlerIo** (object) - A HandlerIO object

**port** (enum as NAMatHandlerPort) - port to get data from. Choose from:

- **naPortC** - (2)
- **naPortD** - (3)
- **naPortE** - (4)

**Note**: Reading data from the Write-only ports (A,B,F,G,H) will return an error. Ports C and D must be put in Read mode before reading from C, D, or E using PortMode Property.

**Return Type**
Variant

**Default**
0

**Examples**
```vbnet
data = handlerIo.get_Port(naPortC)
```

**C++ Syntax**
```cpp
HRESULT get_Port ( tagNAMatHandlerPort Port, VARIANT* Data );
```

**Interface**
IHWMaterialHandlerIO
Read-only

**get PortCData Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Reads a 4-bit value from Port C of the Aux I/O connector (pins 22-25) and the Material Handler IO (pins 21-24 Anritsu) - (pins 22-25 Avantest).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>These lines are connected to both the Handler IO and Aux IO in the VNA.</td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td>value = AuxIO.get PortCData</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(variant) - Variable to store the returned data</td>
</tr>
<tr>
<td>AuxIO</td>
<td>(object) - A Hardware Auxiliary Input / Output object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>value = auxIo.get PortCData 'Reading a value of 15 when in Positive Logic indicates Port C lines C0, C1, C2, C3 are High. If in Negative Logic they are Low.</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_PortCData( VARIANT* Data );</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IHWAuxIO</td>
</tr>
</tbody>
</table>
GetPortNumber Method

**Description**

Returns the port number that is associated with the specified port name. These numbers are used with several commands to specify a VNA port.

To learn more, see [Remotely Specifying a Source Port](#).

**VB Syntax**

```vbnet
value = object.GetPortNumber (portName)
```

**Variable (Type) - Description**

- `value` *(Long)* - Variable to store the returned Port Number integer value.
- `object` - A Channel *(object)* - always more complete than capabilities object.
- `portName` *(String)* - Name of the VNA port.

- Use [SourcePortNames Property](#) to return a list of VNA port (string) names.
- If an external source is selected, specify the external source name that is used in the [Select an External Source dialog](#).

**Return Type**

Long Integer

**Default**

Not Applicable

**Examples**

```vbnet
value = chan.GetPortNumber ("Src2 Out1") 'Read
```

**C++ Syntax**

```c++
HRESULT GetPortNumber(BSTR name, long *number);
```

**Interface**

IChannel13

ICapabilities4
GetRaw2DData Method

Description

Returns raw data at all frequency and power data points for any GCA sweep. Previously 2D sweep only.

- When using SMART sweep, ALL data is returned including ALL background iteration sweeps. Use TotalIterations to determine the number of iteration sweeps. The number of data points that are returned is always the number of frequency points times the number of iteration sweeps.

- When using 2D sweeps, ALL data is returned. The number of data points returned / freq may vary. Learn more.

Use the standard "get data" commands to return just the displayed data results (not the background sweeps).

A compression parameter must be present. Learn more.

VB Syntax

```vbnet
data = gca.GetRaw2DData location, format, param
```

Variable (Type) - Description

- `data` Variant array in which to store returned measurement data.
- `gca` A GainCompression (object)
- `location` (enum NADataStore) - Where the data you want is residing. Choose from:
  - 0 - naRawData
  - 1 - naCorrectedData
- `format` (enum NADataFormat) - Format in which you would like the data. It does not have to be the displayed format. Choose from:
  - 0 - naDataFormat_LinMag
  - 1 - naDataFormat_LogMag
  - 2 - naDataFormat_Phase
  - 3 - naDataFormat_Polar*
  - 4 - naDataFormat_Smith*
  - 5 - naDataFormat_Delay -- Not valid for this command.
  - 6 - naDataFormat_Real
7 - naDataFormat_Imaginary
8 - naDataFormat_SWR
9 - naDataFormat_PhaseUnwrapped
10 - naDataFormat_InverseSmith

Learn more about Data Format.

* Specify Smith or Polar formats to obtain complex data pairs, which require a two-dimensional array \textit{varData} (numpts, 2) to accommodate both real and imaginary data.

All scalar formats return a single dimension \textit{varData(numpts)}.

\textit{naDataFormat\_Phase} and \textit{naDataFormat\_PhaseUnwrapped} returns degrees.

\textit{param} \textbf{(String)} Parameter of data to return. Not case-sensitive. The specified parameter need NOT be displayed. However, a compression parameter must be present. Learn more.

Choose from:

- "pin" - (Compln21) Input power at the compression point.
- "pout" - (CompOut21) Output power at the compression point.
- "gain" - (CompGain21) Device gain (S21) at the compression point.
- "inputmatch" - (CompS11) Input match at the compression point.
- "DeltaGain" - (DeltaGain21) Measured Gain (watts) / Ref Gain (watts). Learn more.
- "AI1" and "AI2" - ADC measurements at the specified compression level. Learn more.

\textbf{Return Type} Variant Array
\textbf{Default} Not Applicable
\textbf{Examples} \texttt{data = gca.GetRaw2DData naRawData, naDataFormat\_Real, "pin"}
\textbf{C++ Syntax} HRESULT GetRaw2DData(tagNADataStore location, tagNADataFormat format, BSTR data\_name, VARIANT* pData);

\textbf{Interface} IGainCompression
## GetDataIm Method

**Description**
For a specified data point, returns the imaginary part of the specified Gain Compression data. If correction is on, corrected data are returned. Otherwise, raw data are returned. Can be used with Smart and 2D sweeps.

- For SMART sweep, the number of data points that are returned is always going to be the number of iteration sweeps. Use `naIterationSweeps` to determine the number of iteration sweeps.
- For 2D sweeps, the number of data points returned / freq may vary. Learn more.

**VB Syntax**
```
data = gca.GetDataIm stim, dPoint, param```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>data</code></td>
<td>Variant array in which to store returned measurement data.</td>
</tr>
<tr>
<td><code>gca</code></td>
<td>A GainCompression (object)</td>
</tr>
<tr>
<td><code>stim</code></td>
<td>(NAGCAIndexSelect)</td>
</tr>
<tr>
<td><code>dPoint</code></td>
<td>Data point (Frequency or Power) for which data is returned.</td>
</tr>
<tr>
<td><code>param</code></td>
<td>Parameter of data to return. Not case-sensitive. Choose from:</td>
</tr>
<tr>
<td></td>
<td>- &quot;pin&quot; - input power at each data point.</td>
</tr>
<tr>
<td></td>
<td>- &quot;pout&quot; - output power at each data point.</td>
</tr>
<tr>
<td></td>
<td>- &quot;gain&quot; - device gain (S21) at each data point.</td>
</tr>
<tr>
<td></td>
<td>- &quot;inputmatch&quot; - input match (S11) at each data point.</td>
</tr>
<tr>
<td></td>
<td>- &quot;DeltaGain&quot; - Measured Gain (watts) / Ref Gain (watts). Learn more.</td>
</tr>
<tr>
<td></td>
<td>- &quot;AI1&quot; and &quot;AI2&quot; - ADC measurements at the specified compression level. Learn more.</td>
</tr>
</tbody>
</table>

**Return Type**
Variant Array

**Default**
Not Applicable
Examples
For the fifth frequency data point, returns 'Power Output' imaginary (phase) data from all power stimulus values. If there are 30 power sweep points, 30 values are returned.

```c
data = gca.GetDataIm naFrequencySelect, 5, "pout"
```

For the 30th stimulus power data point, returns 'Power Output' imaginary (phase) data from all frequency stimulus values. If there are 201 frequency sweep points, 201 values are returned.

```c
data = gca.GetDataIm naPowerSelect, 30, "pout"
```

Note: For 2D sweeps, the number of data points returned / freq may vary.

Learn more.

C++ Syntax
HRESULT GetDataIm(tagNAGCAIndexSelect index_select, int index,BSTR data_name, VARIANT* pData);

Interface
IGainCompression
GetDataRe Method

Description
Reads the REAL part of the data acquired from any GCA sweep. Previously 2D sweep only.

**Note:** For 2D sweeps, the number of data points returned / freq may vary. Learn more.

VB Syntax
`data = gca.GetDataRe stim, dPoint, param`

Variable (Type) - Description

- **data** Variant array in which to store returned measurement data.
- **gca** A GainCompression object
- **stim** (NAGCAIndexSelect)
  - **naFrequencySelect** - for the specified frequency data point, returns all of the measured data for each power stimulus.
  - **naPowerSelect** - for the specified power data point, returns all of the measured data for each frequency stimulus.
- **dPoint** Data point (Frequency or Power) for which data is returned.
- **param** Parameter of data to return. Not case-sensitive. Choose from:
  - "pin" - input power at each data point.
  - "pout" - output power at each data point.
  - "gain" - device gain (S21) at each data point.
  - "inputmatch" - input match (S11) at each data point.
  - "DeltaGain" - Measured Gain (watts) / Ref Gain (watts). Learn more.
  - "AI1" and "AI2" - ADC measurements at the specified compression level. Learn more.

Return Type
Variant Array

Default
Not Applicable

Examples
For the fifth frequency data point, returns 'Power Output' REAL data from all power stimulus values. If there are 30 power sweep points, 30 values are returned.

`data = gca.GetDataRe naFrequencySelect, 5, "pout"`
For the 30th stimulus power data point, returns 'Power Output' REAL data from all frequency stimulus values. If there are 201 frequency sweep points, 201 values are returned.

```csharp
data = gca.GetDataRe naPowerSelect, 30, "pout"
```

**Note:** For 2D sweeps, the number of data points returned / freq may vary. Learn more.

**C++ Syntax**

```c
HRESULT GetDataRe(tagNAGCAIndexSelect index_select, int index, BSTR data_name, VARIANT* pData);
```

**Interface**

IGainCompression
GetRxLevelingConfiguration Method

Description  This method returns a handle to a RxLevelingConfiguration object.

VB Syntax  

```
chan.GetRxLevelingConfiguration()  
```

Variable  (Type) - Description

```
chan  A Channel (object)  
```

Return Type  IRxLevelingConfiguration

Default  Not Applicable

Example  

```
dim app as AgilentPNA835x.Application
dim mgr as RxLevelingConfiguration
set mgr = app.GetRxLevelingConfiguration()  
```

C++ Syntax  

```
HRESULT GetRxLevelingConfiguration( IRxLevelingConfiguration **mgr);  
```

Interface  IChannel17
## GetSourceByRole Method

**Description**
Returns the name of a source that is assigned to the specified role.

*Note:* For non-converter channels, use `chan.RoleDevice`

**VB Syntax**
```vbnet
source = conv.GetSourceByRole (role)
```

**Variable**
- **source** (String) Source name, from Source Configuration dialog, that is assigned to the specified role.
- **conv** A Converter Object
- **role** (String) Role for which the source name will be returned. Use `GetSourceRoles` for a list of valid roles.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vbnet
RF2Source = conv.GetSourceByRole ("RF2")
```

**C++ Syntax**
```cpp
HRESULT GetSourceByRole(BSTR roleID, BSTR deviceName);
```

**Interface**
IConverter
<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the defined role names (&quot;RF2&quot;, &quot;LO1&quot;).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>For non-converter channels, use <code>chan.RoleDevice</code></td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>roles = conv.GetSourceRoles()</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>roles</code></td>
<td>(Variant array) Variable to store returned list of valid roles.</td>
</tr>
<tr>
<td><code>conv</code></td>
<td>A Converter Object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>roles = conv.GetSourceRoles()</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT GetSourceRoles(VARIANT* roles);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IConverter</td>
</tr>
</tbody>
</table>
### GetReferenceMarker Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a handle to the reference marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>meas.GetReferenceMarker</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Object</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>meas.GetReferenceMarker</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT GetReferenceMarker(IMarker** refMarker)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
Write-only

GetRequiredEtermNames

Description

Returns an array of strings specifying the error terms required by the caltype's correction algorithm in order to correct the specified parameter.

This function interrogates a specific caltype (caltypeGUID) for the list of error terms it would need in order to correct the specified parameter. All the standard S Parameter calibration types embed port specifiers in the error term name. The specific port information is gleaned from the passed parameter.

For example, to query the error term requirements specific to a two port cal on ports 1 and 3, issue this with a parameter of S13 or S31. The buffer names returned will be formatted in this way:

Full 1 Port SOLT(1,3):TransmissionTracking(3,1)

VB Syntax

EtermNames = GetRequiredEtermNames(CalTypeGUID As String, Parameter As String)

Variable (Type) - Description

caltypeGUID: [in] the GUID of the desired calibration type

parameter [in] string specifying the parameter to be corrected

EtermNames [out] array of strings containing the error term names.

Note: In C++ Allocated by server. Must be freed by caller using SysFreeString.

Return Type

Not Applicable

Default

Not Applicable

Examples

enames = GetRequiredEtermNames(ctGUID, Parm)

C++ Syntax

HRESULT GetRequiredEtermNames( BSTR caltypeGUID, BSTR parameter, VARIANT* EtermNames )

Interface

ICalManager2
GetScalar Method

Description
Retrieves scalar data (ONE number per data point) from the specified location.

**Note:** This method exists on a non-default interface. If you cannot access this method, use the Get Data Method on IMeasurement.

**VB Syntax**
```vbnet
measData.getScalar location, format, numPts, data
```

**Variable**

- **measData** (Variable) - An IArrayTransfer interface which supports the Measurement object
- **location** (Type: enum NADATAStore) - Where the data you want is residing. Choose from:
  - 0 - naRawData
  - 1 - naCorrectedData
  - 2 - naMeasResult
  - 3 - naRawMemory
  - 4 - naMemoryResult
  - 5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.
- **format** (Type: enum NADATFormat) - Format in which you would like the data. Choose from:
  - 0 - naDataFormat_LinMag
  - 1 - naDataFormat_LogMag
  - 2 - naDataFormat_Phase
  - 3 - naDataFormat_Polar
  - 4 - naDataFormat_Smith
  - 5 - naDataFormat_Delay
  - 6 - naDataFormat_Real
  - 7 - naDataFormat_Imaginary
8 - naDataFormat_SWR
9 - naDataFormat_PhaseUnwrapped
10 - naDataFormat_InverseSmith
11 - naDataFormat_Kelvin
12 - naDataFormat_Fahrenheit
13 - naDataFormat_Centigrade

**Note:** Polar, Smith, and Inverse Smith are invalid formats for this command. See Get Complex Method.

Learn more about Data Format.

\[
numPts \quad (\text{long integer}) \quad - \quad \text{Number of data points requested}
\]

[out] - specifies number of data elements returned

[in] - specifies the data being requested or the capacity of the \textit{dScalar} array

\[
data \quad (\text{single}) \quad - \quad \text{Array to store the scalar data.}
\]

**Return Type** Single

**Default** Not Applicable

**Examples**

```vbnet
Dim dScalar() As Single
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement
Dim numpts as Long
numpts = app.ActiveChannel.NumberOfPoints
ReDim dScalar(numPoints)
measData.getScalar naCorrectedData, naDataFormat_LogMag, numpts, dScalar(0)
Print dScalar(0), dScalar(1)
```

**C++ Syntax**

```c++
HRESULT getScalar(tagNADataStore DataStore, tagNADataFormat DataFormat, long* pNumValues, float* pVals)
```

**Interface** IArrayTransfer
## GetShortcut Method

### Description
From the index (line number in the user interface) returns the Title, Path, and optional argument strings, of the specified Macro (shortcut). Use this method to list the titles and paths of macros in the analyzer.

### VB Syntax
```vbnet
app.GetShortcut index, title, path, arguments
```

### Variable
**(Type)** - Description

- **app** - An Application **(object)**
- **index** - **(long)** - Number of the macro. Use a number between 1 and 25.
- **title** - **(string)** - Title of the specified macro. (Appears in the softkey label)
- **path** - **(string)** - Pathname of the specified macro.
- **arguments** - **(string)** - Arguments for the specified macro

### Return Type
String

### Default
Not Applicable

### Example
```vbnet
Dim t As String
Dim p As String
Dim arg As String
Dim i As Integer
For i = 1 to 25
    app.GetShortcut i, t, p, arg
    Print t, p
Next
```

### C++ Syntax
```cpp
HRESULT GetShortcut(long Number, BSTR* title, BSTR* pathname, BSTR* arguments)
```

### Interface
IApplication

### Remarks
Shortcuts can also be defined and accessed using the macro key on the front panel. However, the benefit of this feature is primarily for the interactive user.
GetSnPData Method  **Superseded**

**Description**

**Note:** this command is replaced by Get SnpDataWithSpecifiedPorts Method.

Reads SnP data from the selected measurement. Learn more about SnP that is returned from the VNA.

**VB Syntax**

```vbnet
data = meas.GetSnPData type
```

**Variable**

*data* Variant array to store the data.

*meas* A Measurement *(object)*

*type* *(string)* - Type of SnP data to return. If unspecified, <n> is set to 2. Choose from:

- "S1P" returns 1-Port data for the active measurement if the active measurement is a reflection parameter such as S11 or S22. The behavior is UNDEFINED if the active measurement is a transmission parameter such as an S21.

- "S2P" returns data for the current 2-port measurement (4 S-parameters).

- "S3P" returns data for the current 3 port measurement (9 S-parameters). Valid only on instruments with 3 ports or more.

- "S4P" returns data for the current 4 port measurement (16 S-parameters). Valid only on instruments with 4 ports or more.

SnP data can be output using several data formatting options. See SnPFormat Property

**Return Type**

Variant - 3 dimensional array.

- First dimension size is number of parameters returned.

- Second dimension size is number of points in the channel

- Third dimension size is 2 (real,imaginary)

For example:

Data(0,5,1) returns the imaginary value of the fifth data point of S11 (if the s2p
request includes port #1)

<table>
<thead>
<tr>
<th>Default</th>
<th>Not Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td><code>snp = meas.GetSnPData(&quot;slp&quot;)</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT GetSnPData( BSTR snptype, VARIANT * response)</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement3</td>
</tr>
</tbody>
</table>
### Description

**Note:** This command replaces Get SnPData. This command is more explicit regarding the data to be returned, and works for VNAs with multiport test sets.

Reads SnP data for the measurement by specifying the VNA port numbers. Learn more about SnP that is returned from the VNA.

### VB Syntax

```vbnet
data = meas.GetSnPDataWithSpecifiedPorts ports
```

### Variable **(Type) - Description**

- **data** *(Variant)* array to store the data.
- **meas** A Measurement *(object)*
- **ports** *(Variant Array)* One-dimensional array containing the list of port numbers for which data is required.

### Return Type

Variant - 3 dimensional array.

- First dimension size is number of parameters returned.
- Second dimension size is number of points in the channel
- Third dimension size is 2; format of the data is specified with SnPFormat Property.

For example:

Data(0,5,1) returns the imaginary part of the fifth data point of S11 (if the s2p request includes port #1)

### Default

Not Applicable

### Example

'This VBScript example can be pasted into a notepad file and run on the VNA as a macro. Learn how.

```vbnet
Dim pna
Dim meas
Dim param
Dim point
Dim snp
Dim ports
'List the port numbers for required data
```
ports = Array(3,4)
Set pna = CreateObject("AgilentPnA835x.application")
Set meas = pna.ActiveMeasurement
' limit amount of data to display
set chan=pna.ActiveChannel
chan.NumberOfPoints=2
snp = meas.GetSnPDataWithSpecifiedPorts (ports)
' returns a 3 dimensional array
' snp(param,point,data pair)
'-------------------------------------
' show me the data
For param = LBound(snp, 1) To UBound(snp, 1)
  MsgBox ("Parameter: ", (param + 1))
  For point = LBound(snp, 2) To UBound(snp, 2)
    MsgBox "Point:" , (point + 1) , " ", snp(param, point, 0) , " ", , & , snp(param, point, 1)
  Next
Next

---

**C++ Syntax**

```c++
HRESULT GetSnpDataWithSpecifiedPorts(VARIANT portsToMeasure, VARIANT* response);
```

**Interface**

```c++
IMeasurement7
```
**getSourcePowerCalDataEx Method**

**Description**

*Note:* This method replaces `getSourcePowerCalData Method`

Retrieves (as variant data type) source power calibration data, if it exists, from the channel.

If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the `Get X-Axis Values` command to return the X-axis values in the displayed order.

*Note:* This method returns a variant which is less efficient than methods available on the ISourcePowerCalData interface

**VB Syntax**

```vbnet
data = chan.getSourcePowerCalDataEx (buffer, srcPort)
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>(variant) – Array to store the data.</td>
</tr>
<tr>
<td>chan</td>
<td>(object) – A Channel object</td>
</tr>
<tr>
<td>buffer</td>
<td>(enum NASourcePowerCalBuffer) - The requested source power cal data buffer.</td>
</tr>
</tbody>
</table>

0 - **naCorrectionValues**  Last iteration of Cal data

1 - **naPriorIterationCorrectionValues**  Prior iteration of Cal data. This argument can be used to determine the final power reading at each point of the power cal, for a cal that did not pass tolerance limits.

The following formula can be used to determine the power reading (in dB):

\[
\text{Power reading} = \text{Target power at the source port} + \text{specified power cal offset value} + \text{‘prior’ iteration corr value} – \text{actual power corr value.}
\]

The "actual" value in this equation is returned with **naCorrectionValues**.

**srcPort**  (long integer) – The source port for which calibration data is being requested.

*Note:* If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source VNA-x model, then you must use `chan.getPortNumber` to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

**Return Type**

Variant array – automatically dimensioned to the size of the data.
Examples

Dim varData As Variant
Const port1 As Long = 1
varData = chan.getSourcePowerCalDataEx (naCorrectionValues, port1)
'Print the data
For i = 0 to chan.NumberOfPoints - 1
 Print varData(i)
Next i

C++ Syntax

HRESULTgetSourcePowerCalDataEx(tagNASourcePowerCalBuffer
bufSelect, long sourcePort, VARIANT *pData);
getDescriptionPowerCalDataScalarEx Method

**Description**

Note: This method replaces `getDescriptionPowerCalDataScalar` Method

Retrieves (as scalar values) source power calibration data, if it exists, from this channel.

If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the Get X-Axis Values2 command to return the X-axis values in the displayed order.

Note: This method exists on a non-default interface. If you cannot access this method, use the `getDescriptionPowerCalDataEx` Method on IChannel4.

**VB Syntax**

```vbnet
chanData.getDescriptionPowerCalDataScalarEx buffer, srcPort, numValues, data
```

**Variable (Type) - Description**

- **chanData** (interface) – An ISourcePowerCalData2 interface on the Channel object.
- **buffer** (enum NASourcePowerCalBuffer) - The requested source power cal data buffer.
  - 0 - `naCorrectionValues` Last iteration of Cal data.
  - 1 - `naPriorIterationCorrectionValues` Prior iteration of Cal data. This argument can be used to determine the final power reading at each point of the power cal, for a cal that did not pass tolerance limits.

The following formula can be used to determine the power reading (in dB):

Power reading = Target power at the source port + specified power cal offset value + ‘prior’ iteration corr value – actual power corr value.

The "actual" value in this equation is returned with `naCorrectionValues`.

- **srcPort** (long integer) – The source port for which calibration data is being requested.

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source VNA-x model, then you must use `chan.getPortNumber` to translate the string into a port number. To learn more see Remotely Specifying a Source Port.

- **numValues** (long integer) – Number of data values.
[out] – specifies number of data values returned.

[in] – specifies number of values being requested (this must not be larger than the capacity of the data array).

**data** (single) – Array to store the data.

**Return Type** Single

**Default** Not Applicable

**Examples**

```vba
Dim numValues As Long
Dim scalarCalValues() As Single
Dim chanData As ISourcePowerCalData2
Const port1 As Long = 1
numValues = app.ActiveChannel.NumberOfPoints
ReDim scalarCalValues(numValues)
Set chanData = app.ActiveChannel

chanData.getSourcePowerCalDataScalarEx naCorrectionValues, port1, numValues, scalarCalValues(0)

'Print the data
For i = 0 to numValues - 1
Print scalarCalValues(i)
Next I
```

**C++ Syntax**

```c++
HRESULT getSourcePowerCalDataScalarEx(tagNASourcePowerCalBuffer bufSelect, long sourcePort, long *pNumValues, float *pData);
```

**Interface** ISourcePowerCalData2
GetStandard Method **Superseded**

**Description**
This command has been replaced with Get StandardByString

Returns standard acquisition data from the Cal Set. The returned data is complex pairs.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

**VB Syntax**

```
data = CalSet.getStandard (standard, rcv, src)
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>data</em></td>
<td>(Variant) Two-dimensional safe array to store the returned data. Memory for the returned Variant is allocated by the VNA and must be released by client.</td>
</tr>
</tbody>
</table>

**Note:** See also getStandardComplex on the ICalData2 interface to avoid using the variant data type.

**CalSet**

A Cal Set (object)

**standard**

(enum NACalClass) Standard data to be read. Choose from:

1 - naClassA  
2 - naClassB  
3 - naClassC  
4 - naClassD  
5 - naClassE  
6 - naReferenceRatioLine  
7 - naReferenceRatioThru

**SOLT Standards**

1 - naSOLT_Open  
2 - naSOLT_Short  
3 - naSOLT_Load  
4 - naSOLT_Thru  
5 - naSOLT_Isolation

**TRL Standards**

1 - naTRL_Reflection  
2 - naTRL_Line_Reflection
3 - naTRL_Line_Tracking
4 - naTRL_Thru
5 - naTRL_Isolation

\[ rcv \ (long) \ - \text{Receiver Port} \]
\[ src \ (long) \ - \text{Source Port} \]

Return Type \textbf{(variant)}

Default Not Applicable

Examples
\begin{verbatim}
Dim varStd As Variant
Dim varStd2 As Variant
Cal Set.OpenCalSet( naCalType_TwoPortSOLT, 1, 2)
varStd = CalSet.getStandard(naSOLT_Thru,2,1)
varStd2 = CalSet.getStandard(naSOLT_Thru,1,2)
Cal Set.CloseCalSet( )
\end{verbatim}

C++ Syntax
\begin{verbatim}
HRESULT getStandard(tagNACalClass stdclass, long ReceivePort, long SourcePort, VARIANT* pData)
\end{verbatim}

Interface ICalSet
GetStandardByString Method

Description
Returns standard acquisition data from the Cal Set. The returned data is complex pairs.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

VB Syntax
\[
data = \text{calSet}.\text{GetStandardByString}(\text{stdName})
\]

Variable (Type) - Description

\[
data \quad \text{(Variant)} \quad \text{Two-dimensional safe array to store the returned data. Memory for the returned Variant is allocated by the VNA and must be released by client.}
\]

\[\text{Note: See also Get StandardComplexByString on the ICalData2 interface to avoid using the variant data type.}\]

\[\text{calSet} \quad \text{A CalSet (Object)}\]

\[\text{stdName} \quad \text{(String)} \quad \text{The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be "S11C(3,3)".}\]

Return Type
Variant

Default
Not Applicable

Examples
See an example

C++ Syntax
\[
\text{HRESULT GetStandardByString( BSTR bufferName, VARIANT* pData)}
\]

Interface
ICalSet2
**GetStandardComplex Method Superseded**

**Description**  
This command is replaced with Get StandardComplexByString

Returns standard acquisition data from the Cal Set. The returned data is complex pairs.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

**Note:** This method exists on a non-default interface. If you cannot access this method, use the GetStandard Method on ICal Set

### VB Syntax

```
ICalData2.getStandardComplex class, rcv, src, numPts, real(), imag()
```

### Variable

**Type** - Description

- **ICalData2** An ICalData2 pointer to the Cal Set object
- **class** (*enum NACalClass*) Standard data to be read. Choose from:
  1. naClassA
  2. naClassB
  3. naClassC
  4. naClassD
  5. naClassE
  6. naReferenceRatioLine
  7. naReferenceRatioThru

**SOLT Standards**

1. naSOLT_Open
2. naSOLT_Short
3. naSOLT_Load
4. naSOLT_Thru
5. naSOLT_Isolation

**TRL Standards**

1. naTRL_Reflection
2. naTRL_Line_Reflection
3. naTRL_Line_Tracking
4. naTRL_Thru
5 - naTRL_Isolation

recv (long) - Receiver Port

csrc (long) - Source Port

numPts (Long) An In/Out parameter.

On the way in, you specify the max number of values being requested.

On the way out, the VNA returns number of values actually returned.

real() (single) - array to accept the real part of the calibration data. One-dimensional for the number of data points.

imag() (single) - array to accept the imaginary part of the calibration data. One-dimensional for the number of data points.

Return Type (single)

Default Not Applicable

Examples

Dim numpts as long
numpts = ActiveChannel.NumberOfPoints
ReDim r(numpts) ' real part
ReDim i(numpts) ' imaginary part
Dim Cal Set as Cal Set
set Cal Set = pna.GetCalManager.GetCal SetByGUID( txtGUID )
Dim sData As ICalData2
Set sData = Cal Set
sdata.getStandardComplex naSOLT_Open, 1, 1, numpts, r(0), i(0)

C++ Syntax

HRESULT getStandardComplex(tagNACalClass stdclass, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pImag)

Interface ICalData2
GetStandardComplexByString Method

Description
Returns standard acquisition data from the Cal Set.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

VB Syntax
ICalData3.GetStandardComplexByString stdName, lnumPoints, real(0), imag(0)

Variable (Type) - Description

ICalData3 An ICalData3 pointer to a CalSet (Object)

stdName (String) The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be "S11C(3,3)".

lnumPoints (Long) An In/Out parameter.

On the way in, you specify the max number of values being requested.

On the way out, the VNA returns number of values actually returned.

real (Single) The real component of the complex data.

imag (Single) The imaginary component of the complex data.

Return Value
Single

Default Not Applicable

Examples See example

C++ Syntax
HRESULT GetStandardComplexByString( BSTR bufferName, long* lnumPoints, float* real, float* imag);

Interface ICalData3
About Cal Sets

GetStandardsList Method  Superseded

Description

**Note:** This command is replaced by `CalSet.getStandardList2`.

Returns the list of Standards contained in this Cal Set for the CalType specified in the `OpenCal Set` method. Learn more about reading and writing Cal Data using COM.

The list is a comma separated, textual representation of the error terms with the term name followed by the port path in parentheses.

- Standard (n, n),
- Standard (m, n)

Before calling this method you must open the Cal Set with `OpenCal Set`. If the Cal Set is not open, this method returns `E_NA_Cal Set_ACCESS_DENIED`.

Use `StringToNA CalClass` to convert the list entrees to values that can be used with `GetStandard` and `PutStandard`.

**Note:** The port path designation (m n) indicates the receive and source ports for the measurement. Shorts, opens and loads are single port devices, designated in this list by (n n) where n equals the port to which the device is connected. These devices are all characterized by reflection measurements. The dual port thru device is characterized by both transmission and reflection measurements in order to compensate for load match and tracking terms. The notation (n n) indicates the reflection measurement for this device. The notation (m n) indicates the transmission measurement, where the source and receive ports are different.
**VB Syntax**  
*CalSet.GetStandardsList (count, list)*

**Variable (Type) - Description**

- **CalSet (object) -** A Cal Set object
- **count (long [out]) -** indicates the number of items returned in the list
- **list (string) -** Variable to store the returned Comma separated list of items.

**Return Type**  
String

**Default**  
Not Applicable

**Examples**

```vbnet
Dim count As Integer
Dim list As String
OpenCalSet (naCalType_TwoPortSOLT, 1, 2)
GetStandardsList( count, list)
CloseCalSet( )
```

Assuming the Cal Set contained the full set of standards for this two port cal,  
the returned list would be:

"Open(1 1),  
Short(1 1),  
Load(1 1),  
Thru(1 1),  
Isolation(2 1),  
Open(2 2),  
Short(2 2),  
Load(2 2),  
Thru(2 2),  
Isolation(1 2)  
Thru(2 1),  
Thru(1 2)"

**C++ Syntax**  
`HRESULT GetStandardsList( long* count, BSTR* list);`

**Interface**  
ICalSet
## GetStandardList2 Method

**Description**
Returns a list of standards names found in the Cal Set object for the specified text filter.

**Note:** The “Standards data” container in the calset is intended for internal use only. External access is provided for use in diagnosing calibration problems. Users should not form any expectations as to the presence of the data or the naming conventions used.

### VB Syntax
```vbnet
list = calset.GetStandardList2(standard filter)
```

### Variable (Type) - Description
- **calset** (object) - A CalSet object
- **list** (Variant) Variant containing a string array of standards for the specified filter.
- **standard filter** (String) This string is used as a filter so that only the standards of interest are returned. If the filter is empty, all standards are returned. The string is case-insensitive. Here are some examples:
  - “” (empty string) - returns all standard names for the identified Cal Set
  - “THR” - returns all standard names that include the substring “thr,” such as Thru(1,1) and Thru(1,2), etc.
  - “(1,2)” - returns all standards that contain “(1,2),” such as “Isolation(1,2)” and “Thru(1,2).”

### Return Type
Variant

### Default
Not Applicable

### Examples
See an example

### C++ Syntax
```cpp
HRESULT GetStandardList2 (BSTR filter, VARIANT* list)
```

### Interface
ICalSet2
**GetStandardsForClass Method**

**Description**  
Get the calibration standard numbers for a specified calibration class. To set the calibration number use `SetStandardsForClass` Method.

**VB Syntax**  
`calkit.GetStandardsForClass(calclassorder, std1, std2, std3, std4, std5, std6, std7)`

**Variable**  
*(Type)* - Description

- **calKit** A CalKit *(object)*
- **calclassorder** *(enum NACalClassOrder)* Choose from:
  
  0 - naRefl_1_S11  
  1 - naRefl_2_S11  
  2 - naRefl_3_S11  
  3 - naTran_1_S21  
  4 - naRefl_1_S22  
  5 - naRefl_2_S22  
  6 - naRefl_3_S22  
  7 - naTran_1_S12  
  8 - naRefl_1_S33  
  9 - naRefl_2_S33  
  10 - naRefl_3_S33  
  11 - naTran_1_S32  
  12 - naTran_1_S23  
  13 - naTran_1_S31  
  14 - naTran_1_S13  
  15 - naTRL_T  
  16 - naTRL_R  
  17 - naTRL_L

- **std1…std7** *(long)* Calibration Standard Number. Nominal values from *1* through *30*. *0* indicates that a standard number has not been selected.
<table>
<thead>
<tr>
<th>Return Type</th>
<th>Not applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>calkit.GetStandardsForClass naRefl_3_S11, std1, std2, std3, std4, std5, std6, std7</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT GetStandardsForClass(NACalClassOrder calclassorder, long std1, long std2, long std3, long std4, long std5, long std6, long std7)</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ICalKit</td>
</tr>
</tbody>
</table>
**GetStepDescription Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the description of the specified step in the calibration process. For an ECal User Characterization this returns the description of the specified step in the ECal User Characterization process.</th>
</tr>
</thead>
</table>

**VB Syntax**

```vbnet
value = obj.GetStepDescription(n)
```

<table>
<thead>
<tr>
<th>Variable (Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>value</strong> (string) - Variable to store the returned number of steps.</td>
</tr>
<tr>
<td><strong>obj</strong> Any of the following:</td>
</tr>
<tr>
<td>GuidedCalibration (object)</td>
</tr>
<tr>
<td>SMCType (object)</td>
</tr>
<tr>
<td>VMCType (object)</td>
</tr>
<tr>
<td>ECalUserCharacterizer (object)</td>
</tr>
<tr>
<td><strong>n</strong> (Long) Step in the process.</td>
</tr>
</tbody>
</table>

Use `GenerateSteps` to determine the total number of steps.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value = SMC.GetStepDescription(5)</code></td>
</tr>
</tbody>
</table>

**C++ Syntax**

```c++
HRESULT get_GetStepDescription(long step, BSTR* str);
```

**Interface**

IGuidedCalibration  
SMCType  
VMCType  
IECalUserCharacterizer
### GetSupportedALCModes Method

**Description**
Returns the valid ALC Modes for the VNA.

See `ALCLevelingMode` for a list of supported ALC Modes.

**VB Syntax**

```vbnet
value = chan.GetSupportedALCModes(sourcePort)
```

**Variable (Type) - Description**

- `value` (Variant Array) - variable to store the returned valid ALC Modes.
- `chan` (object) - A Channel object
- `sourcePort` (long integer) - Source port.

**Note:** If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source VNA-x model, then you must use `chan.getPortNumber` to translate the string into a port number. To learn more see `Remotely Specifying a Source Port`.

**Return Type**
Variant array

**Default**
Not Applicable

**Examples**
```
modes = chan.GetSupportedALCModes(4) 'Read
```

**C++ Syntax**

```c++
HRESULT GetSupportedALCModes(long port, VARIANT * ALCModes);
```

**Interface**
IChannel9
GetTestResult Method

Description
Returns the result of limit line testing. There are three ways to use this command:

- If neither optional parameter is specified, limit results for ALL data is returned.
- If one parameter is specified (`start`), the limit result for that data point is returned.
- If both parameters are specified, limit results are returned beginning with `start`, and ending with `(start+size)-1`

**Note:** In 'strongly-typed' languages such as C#, all parameters must be specified.

**VB Syntax**
```
testRes = limts.GetTestResult [start,size]
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>testRes</code></td>
<td>(<code>enum NALimitTestResult</code>) - A dimensioned variable to store test results. If a limit line is not tested, a PASS is returned.</td>
</tr>
<tr>
<td><code>limts</code></td>
<td>A LimitTest (<code>object</code>)</td>
</tr>
<tr>
<td><code>start</code></td>
<td>(<code>long</code>) - Optional argument. A start data point number to return limit test results.</td>
</tr>
<tr>
<td><code>size</code></td>
<td>(<code>long</code>) - Optional argument. Number of data points from <code>start</code> to return limit test results.</td>
</tr>
</tbody>
</table>

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**
```
Dim testRes As NALimitTestResult
testRes = limts.GetTestResult
Select Case testRes
    Case 1
        Print "Fails"
    Case 2
        Print "Pass"
End Select
```
**C++ Syntax**  
HRESULT GetTestResult(long lStart, long lSize, tagNALimitTestResult *pVal)

**Interface**  
ILimitTest
GetTraceStatistics Method

Description
Returns all four Trace Statistics. To retrieve individual Trace statistics, use Mean, PeakToPeak, StandardDeviation properties. Use ShowStatistics to display the statistics of the screen.

VB Syntax
meas.GetTraceStatistics pp, mean, stdev

Variable
meas (Type) - Description
A Measurement (object)

pp, mean, stdev (double) - Dimensioned variables to store the returned values

Return Type
Double

Default
Not Applicable

Examples
'Dimension variables
Dim pp As Double
Dim mean As Double
Dim stdv As Double
meas.GetTraceStatistics pp, mean, stdv

C++ Syntax
HRESULT GetTraceStatistics(double* pp, double* mean, double* stdDeviation)

Interface
IMeasurement
GetXAxisValues2 Method

Description
Returns the channel's X-axis values into a dimensioned Typed array. GetXAxisValues2 is a convenient method for determining the frequency of each point when the points are not linearly spaced - as in segment sweep.

Note: This method will fail if called using a scripting client such as VBScript or Keysight Vee, (see remarks)

Note: In Segment Sweep, chan.NumberofPoints will return the total number of data points for the combined segments.

VB Syntax
chan.GetXAxisValues2 numPts, data

Variable (Type) - Description
chan (object) - A Channel object
numPts (long integer) - Number of data points in the channel
data (double) Single dimensioned array of data matching the number of points in the channel.

Return Type double
Default Not applicable

Examples
Dim App As Application
Set App = New Application
Dim numPoints As Long
Dim values() As Double
numPoints = App.ActiveChannel.NumberOfPoints
ReDim values(numPoints)
App.ActiveChannel.GetXAxisValues2 numPoints, values(0)
Print values(0), values(1)

C++ Syntax
HRESULT GetXAxisValues2(long* pNumValues, double* stimulus)

Interface IChannel

Remarks:
This method will fail if called using a scripting client such as VBScript or Keysight Vee. Use the GetXAxisValues method as a replacement for these COM environments.

This method also cannot be called using late-bound typing in Visual Basic. For instance, if, in the example above, the first line were replaced with "Dim App as Object", then this method would fail.
GetXAxisValues Method

Description
Returns the stimulus values for the measurement. To understand how this property is useful, see IMeasurement2 Interface.

VB Syntax
\[
data = meas.GetXAxisValues
\]

Variable
(Type) - Description

*data* (Variant) Array to store the data.

*meas* A Measurement (object)

Return Type
Variant

Default
Not Applicable

Examples
```vbnet
Dim varData As Variant
Dim i As Integer
varData = meas.GetXAxisValues
'Print Data
For i = 0 To meas.NumberOfPoints - 1
Print varData(i)
Next i
```

See C++ example

C++ Syntax
```cpp
HRESULT GetXAxisValues(VARIANT* xData);
```

Interface
IMeasurement2
GetXAxisValues Method

**Description**
Returns the channel's X-axis values. GetXAxisValues is a convenient method for determining the frequency of each point when the points are not linearly spaced - as in segment sweep.

See the Measurement2 Interface to learn how this method differs from meas.GetXAxisValues.

**Note:** This method returns a variant which is less efficient than GetXAxisValues2.

**Note:** In Segment Sweep, chan.NumberofPoints will return the total number of data points for the combined segments.

**VB Syntax**
```
data = chan.GetXAxisValues
```

**Variable**
- **(Type) - Description**
  - `data`: Variant array to store the data.
  - `chan`: A Channel (object)

**Return Type**
Variant

**Default**
Not Applicable

**Examples**
```
Dim varData As Variant
Dim i As Integer
varData = chan.GetXAxisValues
'Print Data
For i = 0 To chan.NumberOfPoints - 1
    Print varData(i)
Next i
```

**C++ Syntax**
```
HRESULT GetXAxisValues (VARIANT* xData)
```

**Interface**
IChannel
GetXDataBuffer Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Retrieves frequency tone data from the modulation distortion measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>data = meas.GetXDataBuffer DataBufferName</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>data</code></td>
<td>(variant) - Array to store the data.</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>(object) - A <code>Measurement</code> object</td>
</tr>
<tr>
<td><code>DataBufferName</code></td>
<td>(string) – Name of the buffer to be read.</td>
</tr>
</tbody>
</table>

**Note:** The following names assume a 2-port VNA. Also, by default, the parameter names assume that VNA Port 1 is connected to the DUT input and VNA Port 2 is connected to the DUT output. If the RF Path settings are changed from the default, the names will change accordingly. For example, if the VNA has 4 ports and Port 4 is connected to the DUT output, the names will change to "PIn1", "POut4", "S11", "S41", etc.

Choose from:

"POut2" - Power Out

"PIn1" - Power In

"MSig2" - Modulation Signal Out

"MDist2" - Modulation Distortion Out

"MGain21" - Modulation Gain

"MComp21" - Modulation Compression

"PGain21" - Power Gain

"S11" - Linear Input Match

"S21" - Linear Gain

"LPin1" - Linear Power In

"LPOut1" - Linear Reflected Power In
<table>
<thead>
<tr>
<th><strong>Return Type</strong></th>
<th>Variant array</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meas.GetXDataBuffer &quot;S21&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT GetXDataBuffer(BSTR DataBufferName, VARIANT* Data );</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement19</td>
</tr>
</tbody>
</table>
GetXDataBufferCompact Method

Description
Retrieves compact signal frequency tone data from the modulation distortion measurement.

VB Syntax
\[ \text{data} = \text{meas}.\text{GetXDataBufferCompact} \ \text{DataBufferName} \]

Variable
\begin{itemize}
  \item \textit{data} \ (\text{variant}) - Array to store the data.
  \item \textit{meas} \ (\text{object}) - A Measurement object
  \item \textit{DataBufferName} \ (\text{string}) – Name of the buffer to be read.
\end{itemize}

Note: The following names assume a 2-port VNA. Also, by default, the parameter names assume that VNA Port 1 is connected to the DUT input and VNA Port 2 is connected to the DUT output. If the RF Path settings are changed from the default, the names will change accordingly. For example, if the VNA has 4 ports and Port 4 is connected to the DUT output, the names will change to "PIn1", "POut4", "S11", "S41", etc.

Choose from:

"POut2" - Power Out

"PIn1" - Power In

"MSig2" - Modulation Signal Out

"MDist2" - Modulation Distortion Out

"MGain21" - Modulation Gain

"MComp21" - Modulation Compression

"PGain21" - Power Gain

"S11" - Linear Input Match

"S21" - Linear Gain

"LPin1" - Linear Power In
"**LPOut1**" - Linear Reflected Power In

"**LPOut2**" - Linear Power Out

<table>
<thead>
<tr>
<th><strong>Return Type</strong></th>
<th>Variant array</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meas.GetXDataBufferCompact &quot;S21&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT GetXDataBufferCompact(BSTR DataBufferName, VARIANT* Data);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement19</td>
</tr>
</tbody>
</table>
### GetVISATimeout Method

**Description**
Returns the timeout value (in milliseconds) for pass-through commands for the specified VISA session.

**VB Syntax**
```vbnet
value = Vpassthru.GetVISATimeout(visaID)
```

**Variable (Type) - Description**
- **value** *(long)* - Variable to store the returned value of the timeout.
- **Vpassthru** *(object)* - A VISAPassthrough object
- **visaID** *(long)* - VISA session number (see Open method).

**Return Type**
Long

**Default**
Not Applicable

**Examples**
```vbnet
value = Vpassthru.GetVISATimeout(2)
```

**C++ Syntax**
```c
HRESULT GetVISATimeout(long session_num)
```

**Interface**
IVISAPassthrough
HasCalType Method

Description
Verifies that the Cal Set object contains the error terms required to perform the specified correction (CalType) to an appropriate measurement.

The argument list includes specifiers for up to 3 ports. The number of arguments required depends on the CalType specified. The value for each port is set to 0 if not specified.

VB Syntax
`check = CalSet.HasCalType(calType, p1, p2, p3)`

Variable (Type) - Description
- **check** (boolean) - variable to store the returned value
  - **TRUE (1)** - Cal Set has all of the error terms necessary to apply the specified correction CalType.
  - **FALSE(0)** - Cal Set DOES NOT have all of the error terms necessary to apply the specified CalType.
- **CalSet** (object) - A Cal Set object
- **calType** (enum as naCalType) - type of correction to be applied. Choose from:
  
<table>
<thead>
<tr>
<th>Caltype</th>
<th>p arguments required</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - naCalType_Response_Open</td>
<td>p1</td>
</tr>
<tr>
<td>1 - naCalType_Response_Short</td>
<td>p1</td>
</tr>
<tr>
<td>2 - *naCalType_Response_Thru</td>
<td>p1 (rcv), p2 (src)</td>
</tr>
<tr>
<td>3 - *naCalType_Response_Thru_And_Isol</td>
<td>p1 (rcv), p2 (src)</td>
</tr>
<tr>
<td>4 - naCalType_OnePort</td>
<td>p1</td>
</tr>
<tr>
<td>5 - naCalType_TwoPort_SOLT</td>
<td>p1, p2</td>
</tr>
<tr>
<td>6 - naCalType_TwoPort_TRCL</td>
<td>p1, p2</td>
</tr>
<tr>
<td>7 - naCalType_None</td>
<td>N/A</td>
</tr>
<tr>
<td>8 - naCalType_ThreePort_SOLT</td>
<td>p1, p2, p3</td>
</tr>
<tr>
<td>9 - Custom</td>
<td>N/A</td>
</tr>
<tr>
<td>10 - naCalType_FourPort_SOLT</td>
<td>p1, p2, p3</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
  * order of port arguments is significant for these CalTypes

- **p1** (long) - required. This argument must be specified.

This specifies either:
- the one significant port for an open/short response cal or a 1 port cal.

- or one of the ports involved in a 2, 3, or 4 port cal

- or the **receive** port for a thru response / thru-isolation cal.

\[ p^2 \text{ (long)} \] - required for any CalType involving more than one port

This specifies either:

- one of the ports involved in a 2, 3, or 4 port cal (order independent)

- or the **source** port for a thru response / thru-isolation cal

\[ p^3 \text{ (long)} \] - required for 3 and 4-port cal

This specifies one of the ports involved in a 3 or 4 port cal (order independent)

<table>
<thead>
<tr>
<th>Return Type</th>
<th>VARIANT_BOOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>\texttt{value = CalSet.HasCalType(nacCalType_TwoPort_TRL, 1, 2)}</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>\texttt{HRESULT HasCalType( tagNACalType, long port1, long port2, long port3, BOOL *pVal);}</td>
</tr>
<tr>
<td>Interface</td>
<td>ICalSet</td>
</tr>
</tbody>
</table>
# Hold Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Puts the channel in Hold - not sweeping. See <code>chans.Hold</code> to put ALL channels in hold.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan.Hold [sync]</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>chan</code></td>
<td>A Channel (object)</td>
</tr>
<tr>
<td><code>[sync]</code></td>
<td>The <code>[sync]</code> argument is ignored.</td>
</tr>
<tr>
<td></td>
<td>Program control ALWAYS waits until the channel is in the Hold state.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chan.Hold</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Hold</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel</td>
</tr>
</tbody>
</table>
### Hold (channels) Method

**Description**  
Places ALL channels in hold mode.

To resume all channels sweeping, use `chans.Resume`. (Must be the same instance of `chans`).

To place a single channel in hold mode, use `channel.Hold` Method.

**VB Syntax**  
`chans.Hold`

**Variable**  
`chans`  
A Channel collection *(object)*

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`chans.Hold`

**C++ Syntax**  
`HRESULT Hold();`

**Interface**  
IChannels2
### ImportCSVfile Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Imports segment data to a CSV file.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>segs.ImportCSVfile(bstrFile)</code></td>
</tr>
</tbody>
</table>
| **Variable** | *(Type) - Description*
| `segs` | A `Segments` collection *(object)* |
| `bstrFile` | *(string) - Full path, file name, and extension of the file.* |
| **Return Type** | Not Applicable |
| **Default** | Not Applicable |
| **Examples** | `segs.ImportCSVfile("D:\MyFile.csv")` |
| **C++ Syntax** | `HRESULT ImportCSVfile(BSTR bstrFile)` |
| **Interface** | ISegments6 |
### ImportDataSet Method

**Description**  
Imports the Guided Power Cal Set (from an existing SMC Cal) or Phase Reference Cal into the current SMC calibrations.

**For the Guided Power Cal:**

The port of the mixer input must have the same source attenuator setting between the SMC channel and the Guided Power Cal Set. The frequencies of the Guided Power Cal must include all the mixer frequencies. Interpolation will be applied to the Guided Power Cal frequencies if they do not exactly match.

**For the Phase Reference Cal:**

The port of the mixer input must have the same source attenuator setting as used in the phase reference cal. The phase reference cal must include all the mixer frequencies. Interpolation will be applied to the phase reference cal frequencies if they do not exactly match. Learn more about Phase Reference Cal.

The following error message may appear (it is not written to the VNA Error Log):

**Interpolation target is out of range. Cannot interpolate** when incompatible frequency ranges occur.

**VB Syntax**  
```
smc.ImportDataSet (calset, dataName)
```

**Variable (Type) - Description**

- **smc**  
  SMCType (object)

- **calset**  
  (String) Name of existing SMC Cal Set from which cal data is imported.

- **dataName**  
  (String) Name of the data set. Choose from:

  - "POWER_STEP" - Import Guided Power Cal data.
  
  - "POWER_AND_PHASE" - Import the Phase Reference + power cal data. When this command is sent, the SMC Cal Method is automatically set to Use Phase Reference Cal Set. Learn more. There is no other command to set this.

**Return Type**  
Not Applicable

**Default**  
Not Applicable
<table>
<thead>
<tr>
<th>Examples</th>
<th><code>smc.ImportDataSet(&quot;MySMCCal&quot;,&quot;POWER_STEP&quot;)</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>See example program</td>
<td></td>
</tr>
</tbody>
</table>

| **C++ Syntax** | HRESULT ImportDataSet(BSTR csName, BSTR dataset); |
| Interface      | SMCType4                                       |
# ImportLibrary Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Imports an Equation Editor DLL.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>equation.ImportLibrary location</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>equation</code></td>
<td>A <code>MeasurementEquation</code> object</td>
</tr>
<tr>
<td><code>location</code></td>
<td>(string) – Full path and filename of the *.dll to be imported.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>equation.ImportLibrary &quot;C:\Program Files(x86)\Keysight\Network Analyzer\UserFunctions\Expansion.dll&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT ImportLibrary( BSTR filename);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IMeasurementEquation</code></td>
</tr>
</tbody>
</table>
### Initialize Method

**Description**
Begins a calibration.

**Note:** `chan` must be the active channel.

For ECal User Characterization, use Initialize (ECal).

**VB Syntax**

```vbnet
obj.Initialize(chan, useCalStorPref)
```

**Variable** (Type) - Description

- `obj` Any of the following:
  - `GuidedCalibration` (object)
  - `SMCType` (object)
  - `VMCType` (object)

- `chan` (Long) Channel number to calibrate.

- `useCalStorPref` (boolean)
  - True or 1 - Assignment of Cal Set will be based on the setting of the `RemoteCalStoragePreference` COM property.
  - False or 0 – If the channel currently has a selected Cal Set, the calibration will be stored to that Cal Set. Otherwise, the assignment of Cal Set is based upon the setting of the `RemoteCalStoragePreference` COM property.

**Return Type** Not Applicable

**Default** Not Applicable

**Examples**

```vbnet
smc.Initialize(2,True)
```

**C++ Syntax**

```cpp
HRESULT put_Initialize(long channelnumber, VARIANT_BOOL bCalPref);
```

**Interface**

- `IGuidedCalibration`
- `SMCType`
- `VMCType`
About ECal User Characterization

InitializeEx Method

**Description**: This property replaces Initialize (ECal) Method.

Initiates a User Characterization of an ECal module. The specified channel number must be an S-parameter measurement channel. The channel must already be calibrated using the same, or greater number of VNA ports as the ECal module. Also, the calibrated VNA ports must begin with Port 1 and use sequential port numbers.

For characterizations that are to be saved to the ECal module, the User Characterization number must already be set before issuing this command using CharacterizationNumber Property. For characterizations that are to be saved to the VNA disk memory, setting the User Characterization number is not necessary.

After this command is executed, subsequent commands can be used to query the number of measurement steps, issue the acquisition commands, query the connection description strings, and subsequently complete an Ecal User characterization.

**VB Syntax**: `userChar.InitializeEx(chan,bool)`

**Variable (Type) - Description**

- **userChar**: ECalUserCharacterizer (object)
- **chan**: (Long) Channel number of a calibrated S-parameter channel.
- **bool**: (Boolean) Choose from:
  - **True**: Check ECal memory to ensure that a new characterization with the channel’s current number of points will fit in the module memory. Select for User Characterizations to be stored in internal ECal memory.
  - **False**: Skip the check. Select for User Characterizations is to be stored to VNA disk memory.

**Return Type**: Not Applicable

**Default**: Not Applicable

**Examples**: `userchar.InitializeEx(2,True)`

**C++ Syntax**: `HRESULT put_InitializeEx(long chanNum,VARIANT_BOOL bCheck);`

**Interface**: IECalUserCharacterizer2
### Initialize Method Superseded

**Description**

Note: This command is replaced with InitializeEx Method

Initiates a User Characterization of an ECal module. The specified channel number must be an S-parameter measurement channel. The User characterization number must already be set before issuing this command using CharacterizationNumber Property.

**VB Syntax**

```
userChar.Initialize(chan)
```

**Variable (Type) - Description**

- `userChar`  ECalUserCharacterizer (object)
- `chan` (Long) Channel number of a calibrated S-parameter channel.

**Return Type**

Not Applicable

**Default**

Not Applicable

**Examples**

```
userchar.Initialize(2)
```

**C++ Syntax**

```
HRESULT put_Inititalize(long chanNum);
```

**Interface**

IECalUserCharacterizer
## IsLibraryImported Method

**Description**  
Returns whether a DLL has been imported into the VNA.

**VB Syntax**  

```vbnet
flag = equation.IsLibraryImported location
```

**Variable**  
(Types) - Description

- `flag` *(Boolean)*
  - **True** - DLL has been imported.
  - **False** - DLL has NOT been imported.

- `equation` *(MeasurementEquation object)*
- `location` *(string)* – Full path and filename of the *.dll.

**Return Type**  
Variant boolean

**Default**  
Not Applicable

**Examples**

```vbnet
flag = equation.IsLibraryImported "C:\Program Files(x86)\Keysight\Network Analyzer\UserFunctions\Expansion.dll"
```

**C++ Syntax**  

```c++
HRESULT IsLibraryImported( BSTR filename, VARIANT_BOOL* pImported);
```

**Interface**  
IMeasurementEquation
**Item Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns an object from the collection of objects.</th>
</tr>
</thead>
</table>

**Notes**

- The order of objects within a collection cannot be assumed.
- Most, but not all, VNA Collections are ‘1-based’

**VB Syntax**

```vbnet
Object[.Item](n)
```

**Variable**

*(Type) - Description*

**Object**

Any of the following *(collections)*:

- Cables Collection
- CalFactorSegments collection
- CalFactorSegmentsPMAR Collection
- Cal Sets collection
- Channels collection
- E5091Testset collection
- ECalModules Collection
- ExternalDevices Collection
- ExternalTestsets collection
- FOM Collection
- GuidedCalibrationPowerSensors Collection
- LimitTest collection
- Measurements collection
- NaWindows collection
- Ports Collection
- PowerLossSegments collection
- PowerLossSegmentsPMAR_Collection
- PowerSensors collection
- Segments collection
- Traces collection
- PowerMeterInterfaces Collection

Learn more about collections in the VNA

**.Item** Optional - Item is the default property of a collections object and therefore can be called implicitly. For example, the following two commands are equivalent:

```
Channels.Item(3).Averaging = 1
Channels(3).Averaging = 1
```

\( n \) (variant) - Number of the item in the collection.

In addition, the following collections allow you to specify the name of the item as a string:

- The Measurements, Traces, and FOM collections. For example:
  ```
  measCollection("CH_S11_1").InterpolateMarkers
  ```

- The Cal Sets collection. For example:
  ```
  Calsets("MyCalSet").Description = "New Description"
  ```

- The ExternalDevices Collection. For example:
  ```
  Set extDev = externalDevices.Item("NewPMAR")
  ```

- The GuidedCalibrationPowerSensors Collection. For example:
  ```
  Set PowerSensor = GuidedCalibrationPowerSensors.Item.Name = "26GHzPowerSensor"
  ```

**Return Type** *(Object)*

**Default** Not Applicable

**Examples**

```
For i = 1 to Traces.Count 1
    Traces.Item(i).YScale = .5dB
Next i
```

**C++ Syntax**

```c++
HRESULT Item(VARIANT index, <interface>** pItem)
```

**Interfaces**

All listed above.
### Item (Independent Power Calibration Port) Method

<table>
<thead>
<tr>
<th>Description</th>
<th>This method returns a handle to the <code>IndependentPowerCalibrationPort</code> object.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>IndependentPwrrCal.Item</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>IndependentPwrrCal</code></td>
<td>A <code>IndependentPowerCalibration</code> (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td><code>IIndependentPowerCalibrationPort</code></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>CalibrateAllChannels.IndependentPowerCalibration.Item</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT Item( IIndependentPowerCalibrationPort** value);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IIndependentPowerCalibration</code></td>
</tr>
</tbody>
</table>
LANConfigurationInitialize Method

Description: Performs an initialization (reset) of the VNA’s LAN configuration, as dictated by Section 8.14 of the LAN eXtensions for Instrumentation (LXI) standard (Version 1.1). This performs the same operation as pressing the LAN Reset button on the VNA LAN Status dialog.

VB Syntax: app.LANConfigurationInitialize

Variable: (Type) - Description

app An Application (object)

Return Type: Not Applicable

Default: Not Applicable

Examples: app.LANConfigurationInitialize

C++ Syntax: HRESULT LANConfigurationInitialize();

Interface: IApplication13
# LaunchCalWizard Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Launches the S-parameter Cal Wizard on the VNA and does not return until the Cal Wizard is dismissed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>To launch the Cal Wizard for a VNA Application, use the <strong>LaunchDialog Method</strong>.</td>
<td></td>
</tr>
<tr>
<td><strong>Note:</strong> The Cal Wizard operates on the active measurement. Therefore, activate the measurement to be calibrated before launching the Cal Wizard.</td>
<td></td>
</tr>
</tbody>
</table>

## VB Syntax

```vb
success = app.LaunchCalWizard(newCS)
```

## Variable *(Type)* - Description

<table>
<thead>
<tr>
<th>success</th>
<th>(boolean) - variable to store the returned value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>True</strong></td>
<td>The Cal was completed</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>The Cal was canceled without completing the calibration.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>app</th>
<th>An Application <em>(object)</em></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>newCS</th>
<th>(boolean)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>True</strong></td>
<td>Cal will be performed on a new Cal Set.</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>Cal will be performed using the existing Cal Set assigned to the channel. If no Cal Set is found, a new Cal Set will be created.</td>
</tr>
</tbody>
</table>

## Return Type

Boolean

## Default

Not Applicable

## Example

```vb
dim bSuccess as boolean
dim bNewCalset as boolean
bNewCalSet = false
bSuccess = app.LaunchCalWizard( bNewCalSet)
```

## C++ Syntax

```c++
HRESULT LaunchCalWizard(VARIANT_BOOL bCalsuccess)
```

## Interface

IAplication
LaunchDialog Method

Description

Launches the specified dialog box.

The Calibration Wizard dialog that appears depends on the active channel. For example, if a Gain Compression channel is active, then the GCA Cal Wizard appears. Use meas.Activate to activate a measurement and channel.

Remote operation returns after the dialog is dismissed.

To invoke the Cal Wizard and have it return immediately, then use Syst:Corr:Wiz with the SCPI Parser object.

VB Syntax

```
app.LaunchDialog dialog, [data]
```

Variable (Type) - Description

- `app` An Application (object)
- `dialog` (String) Dialog box to launch. Choose from:
  - "SourcePowerCal" See this dialog.
  - "PowerMeterSettings" See this dialog.
  - "PathConfiguration" See this dialog.
  - "CalibrationWizard" Depends on the channel
  - "CalibrationSelection" See this dialog.
  - "CalibrateAll" launches the Cal All wizard dialog.
  - "ConfigurePowerSensors" Starts the external device dialog for configuring power sensors for use with PMAR

[data] (Optional argument) Reserved for future use.

Return Type

Not Applicable

Default

Not Applicable

Examples

```
app.LaunchDialog "SourcePowerCal"
```

C++ Syntax

```
LaunchDialog( BSTR dialog, [defaultvalue(0)] VARIANT dialogData)
```

Interface

IAplication11

See the VNA Object Model
LaunchPowerMeterSettingsDialog Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Launches the Power Meter Settings dialog on the VNA. Changing certain values from that dialog will change values of the corresponding properties on this COM object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>pwrCal.LaunchPowerMeterSettingsDialog</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>pwrCal</code></td>
<td>A SourcePowerCalibrator (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>None</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>powerCalibrator.LaunchPowerMeterSettingsDialog</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT put_LaunchPowerMeterSettingsDialog();</code></td>
</tr>
<tr>
<td>Interface</td>
<td>ISourcePowerCalibrator2</td>
</tr>
</tbody>
</table>
LoadConfiguration Method

Description

Loads the named configuration onto the specified channel.

**Note:** Loading a stored configuration will over-write MANY RF and IF path configuration settings. Make your measurement settings AFTER recalling a stored configuration, NOT before.

Use Configurations Method to return the configuration names that are stored on the VNA.

VB Syntax

`pathMgr.LoadConfiguration ch, name`

Variable (Type) - Description

- `pathMgr` PathConfigurationManager (object)
- `ch` (Long) Channel number of the configuration to be saved.
- `name` (String) Configuration name. "Default" is the default factory configuration.

Return Type

Not Applicable

Default

Not Applicable

Examples

`path.LoadConfiguration 2, "myMixer"`

C++ Syntax

`HRESULT LoadConfiguration (long channelNum, BSTR configName );`

Interface

IPathConfigurationManager
### LoadENRFile Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Loads an ENR file from disk into VNA memory. This file is typically provided by the manufacturer of the noise source.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>enr.LoadENRFile (filename)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>enr</code></td>
<td>An ENRFile (object)</td>
</tr>
<tr>
<td><code>filename</code></td>
<td>(String) - Absolute path and filename of the ENR file.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>See example program</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT LoadENRFile(BSTR filename);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IENRFile</td>
</tr>
</tbody>
</table>
**LoadFile Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Recalls an external device configuration file from the VNA hard drive. Currently, only DC Supply and DC Meter configuration files are supported. See more DC Device commands.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>extDev.LoadFile(filename)</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>extDev</code></td>
<td>An ExternalDevice Object</td>
</tr>
<tr>
<td><code>filename</code></td>
<td>(String) File name and .xml extension of the external device configuration file. Currently, only DC Supply and DC Meter configuration files are supported. See more DC Device commands.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>extDev.LoadFile &quot;MyDCSupply&quot;</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT LoadFile(BSTR newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IExternalDevice</td>
</tr>
</tbody>
</table>
LoadFile Method

Description
Loads a previously-configured mixer attributes file (.mxr)

**VB Syntax**  
`obj.LoadFile(filename)`

**Variable (Type) - Description**

- `obj`  
  A Mixer Interface pointer to the Measurement (object)

  Or

  A Converter Object

- `filename`  
  (String) Full path, file name, and .mxr extension of the mixer attributes file.

  Files are typically stored in "C:\Program Files(x86)\Keysight\Network Analyzer\Documents".

**Return Type**  
String

**Default**  
Not Applicable

**Examples**
```
mixer.LoadFile("D:\myMixer.mxr")
```

**C++ Syntax**  
`HRESULT LoadFile(BSTR newVal)`

**Interface**  
IMixer

IConverter
# LoadTheme Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Load a color theme from a disc file.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>colors.LoadTheme (filename)</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>colors</code></td>
<td>A ComColors (object)</td>
</tr>
<tr>
<td><code>filename</code></td>
<td>(String) - Path and filename of the theme to load.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>colors.LoadTheme = (&quot;c:\Program Files(x86)\Keysight\Network Analyzer\Colors\Theme1.colors&quot;)</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT LoadTheme(BSTR filename);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IColors</td>
</tr>
</tbody>
</table>


# ManualTrigger Method

**Description**  
Triggers the analyzer when `TriggerSetup.Source = naTriggerManual`.  

*Note:* An SMC Fixed Output measurement cannot be triggered using this command. For more information, see the example program.

**VB Syntax**  
```vbnet
app.ManualTrigger [sync],[timeout]
```

**Variable**  
<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><code>[sync]</code></td>
<td>(boolean) - Optional argument. A variable set to either True or False.</td>
</tr>
</tbody>
</table>

**timeout**  
(long) - Optional argument.  
If `sync` is true, `timeout` sets the amount of time the VNA will wait until continuing program execution. Units are milliseconds. A value of -1 (the default setting) causes the VNA to wait indefinitely.

If `sync` is False, the timeout setting is ignored.

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
'After Manual trigger is executed, the VNA will wait 1 second to continue program execution'  
Dim `wait` as Boolean  
`wait = True`  
`app.ManualTrigger wait, 1000`

**C++ Syntax**  
```cpp
HRESULT ManualTrigger(VARIANT_BOOL bSynchronize, long timeout)
```

**Interface**  
IApplication
## Move Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Moves a trace from one window to another.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>trace.Move (winTo)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>trace</code></td>
<td>Trace <strong>(object)</strong></td>
</tr>
<tr>
<td><code>winTo</code></td>
<td>(Long integer) Window number to move the trace to. If the window number does not exist, it will be created.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>trace.Move 2 'Moves the trace to window 2</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT Move()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ITrace3</td>
</tr>
</tbody>
</table>
# NetworkPortMap Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Set the port mapping for a 4-port SNP file to be embedded. To read port mapping, use <code>NetworkPortMapA</code>, <code>NetworkPortMapB</code>, <code>NetworkPortMapC</code>, <code>NetworkPortMapD</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>fixture.NetworkPortMap network, inA, inB, outA, outB</code></td>
</tr>
</tbody>
</table>
| **Variable** | **(Type)** - **Description**  
  - `fixture` A Fixturing (object)  
  - `network` Network position. Choose from 1 or 2.  
  - `inA, inB, outA, outB` Port Mapping. Use four port numbers in any order. |
| **Return Type** | Not Applicable. |
| **Default** | 1,2,3,4 |
| **Examples** | `fixture.NetworkPortMap 1,1,3,2,4` |
| **Interface** | `IFixturing6` |
# NextIFBandwidth Method

**Description**  
A function that returns the Next higher IF Bandwidth value. Use to retrieve the list of available IFBandwidth settings.

**VB Syntax**  
`chan.Next_IFBandwidth bw`

**Variable**  
*(Type)* - Description  
- `chan`: A Channel *(object)*  
- `bw`: *(double)* - The argument that you use to send an IFBandwidth. The function uses this argument to return the Next higher IFbandwidth.

**Return Type**  
Double

**Default**  
Not Applicable

**Examples**  
Public `pnbw` As Double 'declare variable outside of procedure  

```
    pnBW = chan.IFBandwidth 'put the current IFBW in pnBW  
    chan.Next_IFBandwidth pnBW 'function returns the Next higher IFBandwidth.  
    chan.IFBandwidth = pnBW 'set IFBW to the Next value
```

**C++ Syntax**  
`HRESULT Next_IFBandwidth (double *pVal)`

**Interface**  
IChannel
**NumberofGroups Method**

**Description**
Sets the number of trigger signals the channel will receive. After the channel has received that number of trigger signals, the channel switches to Hold mode.

**VB Syntax**
`chan.NumberOfGroups num, sync`

**Variable (Type) - Description**
- `chan` A Channel (object)
- `num` (long integer) Number of trigger signals the channel will receive. Choose any number greater than 0.
- `sync` (boolean)

Variable set to either:

- **True** - subsequent commands are not processed until the groups are complete. Do not use with manual trigger.
- **False** - subsequent commands are processed immediately.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
`chan.NumberOfGroups 5,False`

**C++ Syntax**
`HRESULT NumberOfGroups(long count, VARIANT_BOOL bWait)`

**Interface**
IChannel
This command is no longer necessary. The CalSet.get... and put... commands that required this command have been replaced.

Open the Cal Set to read/write a particular CalType. Learn more about reading and writing Cal Data using COM.

This method is a prerequisite to several other Cal Set methods.

A Cal Set can contain more than one CalType. This method opens the Cal Set and allows access to a particular set of terms. Subsequent commands like getErrorTerm use this information to access the correct error terms in the Cal Set. For example:

```vbnet
cset.OpenCalSet (naCalType_TwoPortSOLT, 3, 2)
cset.PutErrorTerm (naDirectivity, 1, 1, Buffer)
```

The directivity error term for port 1 could belong to any number of caltypes: Full1Port (S11), Full2Port (12), Full2Port (13) or Full3Port (123). The CalType and port specifiers in OpenCalSet directs the uploaded directivity term to the correct set of error terms.

To close the Cal Set, see CloseCalSet.

**VB Syntax**

```vbnet
calset.OpenCalSet (CalType, p1, p2, p3)
```

**Variable (Type) - Description**

- `calset` (object) - A Cal Set object
- `CalType` (enum as naCalType) - type of correction to be applied. Choose from:
Caltype

0 - naCalType_Response_Open
1 - naCalType_Response_Short
2 - *naCalType_Response_Thru
3 - *naCalType_Response_Thru_And_Isol
4 - naCalType_OnePort
5 - naCalType_TwoPort_SOLT
6 - naCalType_TwoPortTRL
7 - naCalType_None
8 - naCalType_ThreePort_SOLT
9 - Custom
10 - naCalType_FourPort_SOLT

* order of port arguments is significant for these CalTypes

p1 (long) - required. This argument must be specified.

This specifies either:
- the one significant port for an open/short response cal or a 1 port cal.
- or one of the ports involved in a 2 or 3 port cal
- or the receive port for a thru response / thru-isolation cal.

p2 (long) - required for any caltype involving more than one port

This specifies either:
- one of the ports involved in a 2 or 3 port cal (order independent)
- or the source port for a thru response / thru-isolation cal

p3 (long) - required only for 3 port cal

This specifies either:
- one of the ports involved in a 3 port cal (order independent)

Return Type
None

Default
Not Applicable

Examples
CalSet.OpenCalSet naCalType_ThreePort_SOLT, 3,2,1
C++ Syntax
HRESULT OpenCalSet ( naCalType, port1, [optional] port2, [optional] port3);

Interface
ICalSet
Open Method

Description
Initiates a VISA pass-through session for a device, and returns a unique session ID to be used whenever communicating with that device. Pass-through sessions can be closed by using the Close method, the Reset method, or by properly shutting down the instrument or the analyzer application. Presetting the instrument will not close existing pass-through sessions.

**Note:** When opening a socket session (addresses of type: “TCPIP[board]::host address::port::SOCKET”), you must use the appropriate VISA Address for the identifier argument. Using an alias to open a socket session is not currently supported. Aliases are allowed for all other types of supported sessions.

**VB Syntax**

```vbnet
value = Vpassthru.Open(identifier, timeoutVal)
```

**Variable (Type) - Description**

- `value` (long) - Variable to store the returned session ID.
- `Vpassthru` (object) - A VISAPassthrough object
- `identifier` (string) - VISA address, or VISA alias of the device to be controlled.
- `timeoutVal` (long) - The amount of time (in milliseconds) to wait for a response from the remote device after sending a command. A "timeout" error is displayed after this time has passed without a response.

**Return Type**
Long

**Default**
Not Applicable

**Examples**

```vbnet
value=Vpassthru.Open("TCPIP0::A-N5242A-10096::hislip1::INSTR",1000)
```

```vbnet
value=Vpassthru.Open("MyAlias",5000)
```

**C++ Syntax**

```c
HRESULT Open(BSTR visaIdentifier, long timeOut, long* sessionID)
```

**Interface**
IVISAPassthrough
## OutputSNPFromECal Method

**Description**
Read S parameter of ECal Thru from the ECal memory and save it as s2p file.

**VB Syntax**
```vbnet
ecalMdls.OutputSNPFromECal (kit, ecalstate, snpfile)
```

**Variable (Type) - Description**
- `ecalMdls` (object) - A ECalModules correction
  - `kit` (String) ECal Model, User char name + " ECal", and serial number of the ECal module used for the characterization, separated by spaces. See examples below.
- `ecalstate` (String) ECal transmission path. Choose from AB, AC, AD, BA, BC, BD, CA, CB, CD, DA, DB or DC.
- `snpfile` (String) Path and filename of the output s2p file name.

**Return Type**
IECalModules Set Interface

**Default**
Not Applicable

**Example**
```vbnet
ecalMdls.OutputSNPFromECal ("N4433A ECal 00001", "BC", "D:\ecalthru.s2p")
```

**C++ Syntax**
```cpp
HRESULT OutputSNPFromECal (BSTRkit, BSTR ecalstate, BSTR snpfile);
```

**Interface**
IECalModules2
## Parse Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Allows the use of COM to send a SCPI command. See a C++ example of how to return error information when using this command.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>The SCPIStringParser Methods can NOT be used with SCPI Status Reporting. However, the *OPC? will work.</td>
</tr>
</tbody>
</table>

### VB Syntax

```
scpi.Parse("SCPI command")
```

### Variable

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>scpi</code></td>
<td>A ScpiStringParser (object)</td>
</tr>
<tr>
<td><code>SCPI command</code></td>
<td>(string) - Any valid SCPI command</td>
</tr>
</tbody>
</table>

### Return Type

String

### Default

Not Applicable

### Examples

```vbnet
Dim scpi As ScpiStringParser
Set scpi = app.ScpiStringParser
Dim startfreq As Double
startfreq = 100e6
's
scpi.Parse "Sens:Freq:Start " & startfreq'Write

Dim str As String
str = scpi.Parse ("Sens:Freq:Start?")'Read
```

### C++ Syntax

```
HRESULT Parse(BSTR SCPI_Command, BSTR *pQueryResponse)
```

### Interface

IScpiStringParser
### Preset Method

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Application Object:</strong> Deletes all traces and windows. In addition, resets the analyzer to factory defined default settings and creates an S11 measurement named &quot;CH1_S11_1&quot; in window 1.</td>
<td></td>
</tr>
<tr>
<td><strong>Channel Object:</strong> Resets the channel (object) to factory defined default settings. Does NOT delete the current measurements or add a new measurement.</td>
<td></td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
app.Preset
chan.Preset
```

**Variable**

- **app** An Application (object)
- **chan** A Channel (object)

**Return Type** Not Applicable

**Default** Not Applicable

**Examples**

```vbnet
app.Preset
```

**C++ Syntax**

```c++
HRESULT Preset()
```

**Interface**

IApplication
IChannel
### PreviousIFBandwidth Method

**Description**
A function that returns the previous IF Bandwidth value. Use to retrieve the list of available IFBandwidth settings.

**VB Syntax**
```
chan.Previous_IFBandwidth bw
```

**Variable**
- **(Type)** - Description
  - `chan` A Channel (object)
  - `bw` (double) - The argument that you use to send an IFBandwidth. The function uses this argument to return the previous IFBandwidth.

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```
Public pnbw As Double 'declare variable outside of procedure
PreBW = chan.IFBandwidth 'put the current IFBW in PreBW
chan.Previous_IFBandwidth PreBW 'function returns the Previous IFBandwidth of the current one.
chan.IFBandwidth = PreBW 'set IFBW to the previous value
```

**C++ Syntax**
```
HRESULT Previous_IFBandwidth (double *pVal)
```

**Interface**
IChannel
## PrintToFile Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves the screen image to a bitmap file.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.PrintToFile filename</code></td>
</tr>
</tbody>
</table>

### Variable (Type) - Description

- **app** An Application *(object)*
- **filename (string)** Full path, file name, and extension of the screen image file.

Files are typically stored in "D:\".

Use one of the following extensions:

- .bmp - not recommended due to large file size
- .jpg - not recommended due to poor quality
- .png - recommended

### Return Type

Not Applicable

### Default

Not Applicable

### Examples

```vbnet
app.PrintToFile "D:\myfile.png"
```

### C++ Syntax

`HRESULT PrintToFile(BSTR bstrFile)`

### Interface

IAplication
### PutComplex Method

**Description**

Puts real and imaginary data into the specified location. This method forces the channel into Hold mode to prevent the input data from being overwritten. Learn more about [reading and writing Cal Data using COM](#).

Data put in the raw data store will be **re-processed** whenever a change is made to the measurement attributes such as format or correction.

Data put in the measurement results store will be **overwritten** by any measurement attribute changes.

See also [putNAComplex](#)

**VB Syntax**

```
measData.putComplex location, numPts, real(), imag(), [format]
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>measData</td>
<td>An IArrayTransfer interface which supports the Measurement object</td>
</tr>
<tr>
<td>location</td>
<td>(enum NADataStore) Where the Data will be put. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naRawData</td>
</tr>
<tr>
<td></td>
<td>1 - naCorrectedData</td>
</tr>
<tr>
<td></td>
<td>2 - naMeasResult</td>
</tr>
<tr>
<td></td>
<td>3 - naRawMemory</td>
</tr>
<tr>
<td></td>
<td>4 - naMemoryResult</td>
</tr>
<tr>
<td></td>
<td>5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.</td>
</tr>
<tr>
<td>numPts</td>
<td>(long integer) - Number of data points in the channel</td>
</tr>
<tr>
<td>real()</td>
<td>(single) - Array containing real data values</td>
</tr>
<tr>
<td>imag()</td>
<td>(single) - Array containing imaginary data values</td>
</tr>
<tr>
<td>format</td>
<td>(enum NADataFormat) optional argument - display format of the real and imaginary data. Only used if destination is naMeasResult or naMemoryResult buffer. If unspecified, data is assumed to be in naDataFormat_Polar</td>
</tr>
<tr>
<td></td>
<td>0 - naDataFormat_LinMag</td>
</tr>
<tr>
<td></td>
<td>1 - naDataFormat_LogMag</td>
</tr>
</tbody>
</table>
Learn more about Data Format.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement
measData.putComplex naMemoryResult, 201, 
real(0), imag(0), naDataFormat_SWR

**C++ Syntax**
HRESULT putComplex( tagNADataStore DataStore, long lNumValues, float* pReal, float* pImag, tagDataFormat displayFormat)

**Interface**
IArrayTransfer
PutDataComplex Method

Description
Puts complex data into the specified location. This method forces the channel into Hold mode to prevent the input data from being overwritten.

VB Syntax
meas.putDataComplex location, data

Variable (Type) - Description
meas A measurement (object)
location (enum NADataStore) Where the Data will be put. Choose from:

0 - naRawData
1 - naCorrectedData
2 - naMeasResult - Valid ONLY when the display format is either Polar or Smith Chart.
3 - naRawMemory - See note below.
4 - naMemoryResult
5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

Note: When putting data into 3 - naRawMemory:

1. Put the analyzer in hold mode
2. Call DataToMemory to initialize a memory buffer
3. Call putDataComplex(naRawMemory, data)

This ensures that the memory buffer is appropriately initialized before receiving new data.

data (variant) - A two-dimensional variant array.

Note: All buffers except naMeasResult and naMemoryResult require Complex data

Return Type Not Applicable
Examples

' Put 201 points worth of raw (complex) data into the measurement
' Note that an array of complex numbers is represented by a 2-D array where the first rank is the number of points, and the 2nd rank is always size 2 (max index 1) representing the Real and Imag parts of the complex number.

' complex array of data (2nd dimension of size 2 represents Re/Im
Dim data(200,1) 
For i = 0 to 200
' Set Real part of data point i
data(i,0) = i/200;
' Set Imag part of data point i
data(i,1) = i/200;
Next
app.ActiveMeasurement.putDataComplex naRawData, data

C++ Syntax
HRESULT putDataComplex(tagNADataStore DataStore, VARIANT complexData)

Interface
IMeasurement
### PutENRData Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Write ENR calibration data to VNA memory. All of the frequency and ENR data must be sent at the same time.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>enr.PutENRData (vData)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>enr</code></td>
<td>An ENRFile <em>(object)</em></td>
</tr>
<tr>
<td><code>vData</code></td>
<td>(Variant array) - ENR data. Frequency value in Hz, followed by corresponding ENR value in dB. Enter as many data pairs as necessary.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>See example program</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT PutENRData(VARIANT vdata);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IENRFile</td>
</tr>
</tbody>
</table>
PutErrorTerm Method - Superseded

Description

Note: This command is replaced by PutErrorTermByString

Puts variant error term data into the error-correction buffer. If this command is being used to modify a calset that is currently in use by the channel, you must send the following commands to see the effects of the change:

Calset::Save
Channel::ErrorCorrection = false
Channel::ErrorCorrection = true

Learn about reading and writing Calibration data.

VB Syntax

```
cal.putErrorTerm(term,rcv, src, data)
```

Variable (Type) - Description

- `cal` A Calibrator (object)
- `term` (enum As NaErrorTerm)
- `rcv` (long integer) - Receiver Port
- `src` (long integer) - Source Port
- `data` (variant) Error term data in a two-dimensional array (0:1, 0:numpts-1).
<table>
<thead>
<tr>
<th>Error Term</th>
<th>Specify these parameters:</th>
<th>term</th>
<th>rcv</th>
<th>src</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fwd Directivity</td>
<td>naET_Directivity Isolation</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Directivity</td>
<td>naET_Directivity Isolation</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Isolation</td>
<td>naET_Directivity Isolation</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Isolation</td>
<td>naET_Directivity Isolation</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Source Match</td>
<td>naErrorTerm_Match</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Source Match</td>
<td>naErrorTerm_Match</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Load Match</td>
<td>naErrorTerm_Match</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Load Match</td>
<td>naErrorTerm_Match</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Reflection Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Reflection Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Trans Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Trans Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Trans Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**: Not Applicable

**Default**: Not Applicable

**Examples**

```vbnet
Dim varError As Variant
varError = cal.putErrorTerm(naErrorTerm_Tracking, 2, 1, VarData)
```

**C++ Syntax**

```cpp
HRESULT putErrorTerm(tagNAErrorTerm ETerm, long ReceivePort, long SourcePort, VARIANT varData)
```

**Interface**

ICalibrator
### PutErrorTerm Method **Superseded**

**Description**

This command is replaced with **PutErrorTermByString**

Puts error term data into the Cal Set.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

**VB Syntax**

```
CalSet.putErrorTerm (term, rcv, src, data)
```

**Variable (Type) - Description**

- **CalSet** (Object) A CalSet Object
- **term** (enum As NaErrorTerm2) Error Term. Choose from:
  0 - naET_Directivity  (src = rcv)
  1 - naET_SourceMatch  (src = rcv)
  2 - naET_ReflectionTracking (src = rcv)
  3 - naET_TransmissionTracking (src ≠ rcv)
  4 - naET_LoadMatch (src ≠ rcv)
  5 - naET_Isolation (src ≠ rcv)
- **rcv** (long integer) - Receiver Port
- **src** (long integer) - Source Port
- **data** (variant) Error term data in a two-dimensional array (0:1, 0:numpts-1). The data must be complex pairs.

**Note:** See also **PutErrorTermComplex** on the ICalData2 interface to avoid using the variant data type.

**Return Type**

Not Applicable

**Default**

Not Applicable

**Examples**

See an Example

**C++ Syntax**

```
HRESULT putErrorTerm(tagNAErrorTerm2 ETerm, long ReceivePort, long SourcePort, VARIANT varData)
```

**Interface**

ICalSet
PutErrorTermByString Method

Description
Puts error term data into the Cal Set. If this command is being used to modify a calset that is currently in use by the channel, you must send the following commands to see the effects of the change:

Calset::Save
Channel::ErrorCorrection = false
Channel::ErrorCorrection = true

Learn more about Reading and Writing Cal Data
See examples of Reading and Writing Cal Set Data

VB Syntax
`calSet.PutErrorTermByString(errorName, vdata)`

Variable (Type) - Description
- `calSet` (Object) A CalSet Object
- `errorName` (String) The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)". To determine the string names of error terms, see GetErrorTermList2.
- `vdata` (Variant) This data array is usually two dimensional. Each element is a type single. The two elements represent the real and imaginary parts of a complex pair.

Note: This structure is compatible with scripting clients who can only use variants. For alternative methods that use typed arrays, see ICalData3.

Return Type
Not Applicable

Default
Not Applicable

Examples
See an Example

C++ Syntax
`HRESULT PutErrorTermByString(BSTR errorName, VARIANT vdata)`

Interface
ICalSet2
**PutErrorTermComplex Method  Superseded**

**Description**

*Note: This command is replaced by PutErrorTermComplexByString*

Puts error term data into the error-correction data buffer. If this command is being used to modify a calset that is currently in use by the channel, you must send the following commands to see the effects of the change:

```
Calset::Save
Channel::ErrorCorrection = false
Channel::ErrorCorrection = true
```

Learn more about reading and writing Cal data using COM

**VB Syntax**

```
data.putErrorTermComplex term, rcv, src, numPts, real(), imag()
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>An ICalData pointer to the Calibrator object</td>
</tr>
<tr>
<td>term</td>
<td>(enum NAErrorTerm) - The error term to be retrieved. Choose from:</td>
</tr>
<tr>
<td></td>
<td>* naErrorTerm_Directivity_Isolation</td>
</tr>
<tr>
<td></td>
<td>* naErrorTerm_Match</td>
</tr>
<tr>
<td></td>
<td>* naErrorTerm_Tracking</td>
</tr>
<tr>
<td>rcv</td>
<td>(long integer) - Receiver Port</td>
</tr>
<tr>
<td>src</td>
<td>(long integer) - Source Port</td>
</tr>
<tr>
<td>numPts</td>
<td>(long integer) - number of data points in the array</td>
</tr>
<tr>
<td>real()</td>
<td>(single) - array containing the <strong>real</strong> part of the calibration data. One-dimensional: the number of data points.</td>
</tr>
<tr>
<td>imag()</td>
<td>(single) - array containing the <strong>imaginary</strong> part of the calibration data. One-dimensional: the number of data points.</td>
</tr>
<tr>
<td>To get this</td>
<td>Specify these parameters:</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Error Term</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Return Type: Not Applicable
Default: Not Applicable

Examples:
```vbnet
Dim eData As ICalData
Set eData = chan.Calibrator
eData.putErrorTermComplex naErrorTerm_Directivity_Isolation, 1, 1, 201, rel(0), img(0)
```

C++ Syntax:
```csharp
HRESULT putErrorTermComplex(tagNAErrorTerm ETerm, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pImag)
```

Interface: ICalData
PutErrorTermComplex Method  **Superseded**

Description  This command is replaced with `PutErrorTermComplexByString`  

Puts error term data into the Cal Set.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

VB Syntax  

```
data.putErrorTermComplex term, rcv, src, numPts, real(), imag()
```

Variable  

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>data</code></td>
<td>An <code>ICalData2</code> pointer to a Cal Set object</td>
</tr>
<tr>
<td><code>term</code></td>
<td>(enum <code>NAErrorTerm2</code>) - The error term to be written. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naET_Directivity</td>
</tr>
<tr>
<td></td>
<td>1 - naET_SourceMatch</td>
</tr>
<tr>
<td></td>
<td>2 - naET_ReflectionTracking</td>
</tr>
<tr>
<td></td>
<td>3 - naET_TransmissionTracking</td>
</tr>
<tr>
<td></td>
<td>4 - naET_LoadMatch</td>
</tr>
<tr>
<td></td>
<td>5 - naET_Isolation</td>
</tr>
<tr>
<td><code>rcv</code></td>
<td>(long) - Receiver Port</td>
</tr>
<tr>
<td><code>src</code></td>
<td>(long) - Source Port</td>
</tr>
<tr>
<td><code>numPts</code></td>
<td>(long) - number of data points in the real and imaginary arrays.</td>
</tr>
<tr>
<td><code>real()</code></td>
<td>(single) - array containing the <strong>real</strong> part of the calibration data. One-dimensional: the number of data points.</td>
</tr>
<tr>
<td><code>imag()</code></td>
<td>(single) - array containing the <strong>imaginary</strong> part of the calibration data. One-dimensional: the number of data points.</td>
</tr>
</tbody>
</table>

Return Type  Not Applicable

Default  Not Applicable

Examples  

```
Dim eData As ICalData2
Set eData = app.GetCalManager.Cal Sets.Item(1)
eData.putErrorTermComplex naET_LoadMatch, 1, 2, numpts, rel(0), img(0)
```
C++ Syntax
HRESULT putErrorTermComplex(tagNAErrorTerm2 ETerm, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pImag)

Interface
ICalData2
PutErrorTermComplexByString Method

Description
Puts error term data into the Cal Set.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

VB Syntax
ICalData3.PutErrorTermComplexByString errorName, InumPoints, real(0), imag(0)

Variable (Type) - Description

ICalData3 An ICalData3 pointer to a Cal Set object.

errorName (String) The string name used to identify a particular error term in the Cal Set.
An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)". To determine the string names of error terms, see GetErrorTermList2.

InumPoints (Long) The number of data points in the real and imaginary arrays.

real (Single) The real component of the complex data.

imag (Single) The imaginary component of the complex data.

Note: The size of the real and imaginary arrays should be the same.

Return Value Not Applicable

Default Not Applicable

Examples See example

C++ Syntax
HRESULT PutErrorTermComplexByString( BSTR bufferName, long InumPoints, float* real, float* imag);

Interface ICalData3
**PutErrorTermStimulus Method**

**Description**
Adds stimulus data to the specified buffer. The size of vdata must agree with the size of the complex data already attached to the buffer or an error will be generated.

See Also: GetErrorTermStimulus Method

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

**VB Syntax**

```vbnet
calSet.PutErrorTermStimulus(bufferName, vdata)
```

**Variable**

- **calSet** *(Object)* - A CalSet Object
- **bufferName** *(String)* - The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)". To determine the string names of error terms, see GetErrorTermList2.
- **vdata** *(Variant)* - Safearray of variants (doubles).

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

The sequence is:

```vbnet
complexData = calset1.GetErrorTermByString(0, BufferName)
frequencyData = calset1.GetErrorTermStimulus(0,BufferName)
// manipulate complex data here
Calset2.PutErrorTermByString(BufferName, manipulatedComplexData)
Calset2.PutErrorTermStimulus(BufferName, frequencyData);
```

**C++ Syntax**

```cpp
HRESULT PutErrorTermStimulus (BSTR bufferName, VARIANT vardata)
```

**Interface**
ICalSet7
PutScalar Method

Description
Puts Scalar data in the Measurement Result buffer. The putScalar array is not processed by the analyzer; it is just displayed. Any change to the measurement state (changing the format, for example) will cause the putScalar data to be overwritten with the data processed from the raw data buffer.

VB Syntax
measData.putScalar, format, numPts, data

Variable (Type) - Description

measData
An IArrayTransfer interface which supports the Measurement object.

format (enum NADeataFormat) Format of the data. Choose from:

0 - naDataFormat_LinMag
1 - naDataFormat_LogMag
2 - naDataFormat_Phase
3 - naDataFormat_Polar
4 - naDataFormat_Smith
5 - naDataFormat_Delay
6 - naDataFormat_Real
7 - naDataFormat_Imaginary
8 - naDataFormat_SWR
9 - naDataFormat_PhaseUnwrapped
10 - naDataFormat_InverseSmith
11 - naDataFormat_Kelvin
12 - naDataFormat_Fahrenheit
13 - naDataFormat_Centigrade

Learn more about Data Format.
**Note:** Smith, InverseSmith, and Polar formats are not allowed.

- **numPts** *(integer)* - Number of values. Usually the number of points in the trace (chan.NumberOfPoints).

- **data** *(single)* - A one-dimensional array of Scalar data matching the number of points in the current measurement.

**Return Type** Not Applicable

**Default** Not Applicable

**Examples**

```vbnet
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement
measData.putScalar naDataFormat_LogMag, 201, dScalar(0)
```

**C++ Syntax**

```cpp
HRESULT putScalar(tagDataFormat eFormat, long lNumValues, float* pArrayOfScalar)
```

**Interface** IArrayTransfer
PutNAComplex Method

Description
Puts complex data into the specified location. This method forces the channel into Hold mode to prevent the input data from being overwritten. The data is processed and displayed.

Data put in the naRawData store will be **re-processed** whenever a change is made to the measurement attributes such as format or correction.

Data put in the naMeasResult store will be **overwritten** by any measurement attribute changes (such as moving a marker).

**Note:** This method uses NAComplex which is a user-defined data type. If you cannot or prefer not to use this data type, use the **putComplex** method.

### VB Syntax

```vbnet
measData.putNAComplex location, numPts, data, [format]
```

### Variables

- **measData** (An IArrayTransfer interface which supports the Measurement object)
- **location** (enum **NADataStore**) Where the Data will be put. Choose from:
  - 0 - naRawData
  - 1 - naCorrectedData
  - 2 - naMeasResult
  - 3 - naRawMemory
  - 4 - naMemoryResult
  - 5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using **DataToDivisor** Method.
- **numPts** (long integer) - Number of data points in the channel
- **data** (NAComplex) - A one-dimensional array of Complex data matching the number of points in the current measurement.
- **format** (enum **NADataFormat**) - Optional argument. Format of the data. If unspecified, naDataFormat_Polar is assumed. Only used when the destination store is naMeasResult or naMemoryResult.
  - 0 - naDataFormat_LinMag
1 - naDataFormat_LogMag
2 - naDataFormat_Phase
3 - naDataFormat_Polar
4 - naDataFormat_Smith
5 - naDataFormat_Delay
6 - naDataFormat_Real
7 - naDataFormat_Imaginary
8 - naDataFormat_SWR
9 - naDataFormat_PhaseUnwrapped
10 - naDataFormat_InverseSmith
11 - naDataFormat_Kelvin
12 - naDataFormat_Fahrenheit
13 - naDataFormat_Centigrade

Learn more about Data Format.

Return Type  Not Applicable
Default       Not Applicable
Examples
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement
measData.putNAComplex naMemoryResult, 201, dRawComplex(0)

C++ Syntax  HRESULT putNAComplex(tagNADataStore DataStore, long lNumValues, TsComplex* pArrayOfComplex, tagDataFormat displayFormat)

Interface  IArrayTransfer
**put_Output Method**

**Description**
Writes a TTL HI or TTL Low to output pins 3 or 4 of the Material Handler IO connector.

Each pin also has a latched output which is written to with USER. With the latched (USER) outputs, the value is not applied to the associated pin until a positive edge is detected at INPUT1 (pin 2).

**VB Syntax**

```vbnet
handlerIo.put_Output (pin) = value
```

**Variable (Type) - Description**

- **handlerIo** *(object)* - A HandlerIO object
- **pin** *(enum as NAMatHandlerOutput)* - pin to write data to. Choose from:
  - `naOutput1` - (0) - pin3
  - `naOutput1User` - (1) - pin3 latched (applied to pin 3 on positive edge of Input1-pin2)
  - `naOutput2` - (2) - pin4
  - `naOutput2User` - (3) - pin4 latched (applied to pin 4 on positive edge of Input1-pin2)
- **value** *(Variant)* - Value to write to the selected pin. Choose from
  - 0 - TTL LOW
  - 1 - TTL HIGH

**Return Type**
Not Applicable

**Default**
0

**Examples**
```
handlerIo.put_Output(naOutput1)= 1
```

**C++ Syntax**

```
HRESULT put_Output ( tagNAMatHandlerOutput Output, VARIANT Data );
```

**Interface**
IHWMaterialHandlerIO
## put_OutputVoltage Method

**Description**

**E836x and PNA-L:** Sets voltages on the DAC/Analog Output 1|2 of the Auxiliary IO connector.

**PNA-X:** Sets voltage on the Power I/O connector AnalogOut1|2.

Read output voltages using `get_OutputVoltage Method`.

**Note:** The 9-pin PWR I/O (Power I/O) D connector on the rear-panel replaces much of the functionality of the AUX I/O connector on older VNA models. The Power I/O voltages can be set using the following methods:

- **CONTrol:AUXiliary:OUTPut:VOLTage** or `put_OutputVoltage Method` (no GUI equivalent, global scoped, and settings not saved as part of the instrument state)
- **SOURce:DC:START** and **SOURce:DC:STOP** (DC Source dialog is the GUI equivalent, channel scoped, and settings saved as part of the instrument state)
- **Interface Control dialog** (no remote equivalent, channel scoped, and settings saved as part of the instrument state)

To avoid unexpected behavior, choose one method only to set the Power I/O voltages.

### VB Syntax

```vb
AuxIO.put_OutputVoltage output, voltage
```

### Variable (Type) - Description

- **AuxIO** *(object)* - A Hardware Auxiliary Input / Output object
- **output** *(variant)* - Number of the output DAC to write voltage to. Choose from:
  - 1 Output 1 (Aux I/O pin 3) and (Power I/O pin 3)
  - 2 Output 2 (Aux I/O pin 2) and (Power I/O pin 4)
- **voltage** *(double)* - Voltage to write to the output DAC. Choose a voltage from -10 to 10

### Return Type

None

### Default

None

### Examples

```
HWAuxIO.put_OutputVoltage 1, 9 'set Analog Out1 to +9v
```

### C++ Syntax

```cpp
HRESULT put_OutputVoltage (VARIANT Output, double Voltage );
```

### Interface

IHWAuxIO
Write-only

put_OutputVoltageMode Method

<table>
<thead>
<tr>
<th>Description</th>
<th>This command sets the mode of the selected &quot;Analog Out&quot; line on the Power I/O connector. The modes give the user the option to have the requested voltage applied immediately or not until the sweep is done. To read the mode on each output use get_OutputVoltageMode Method.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>auxIo.put_OutputVoltageMode (output, mode)</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>auxIo</td>
<td>(Object) An AuxIO object</td>
</tr>
<tr>
<td>output</td>
<td>Analog Output to receive mode setting. Choose from 1 or 2</td>
</tr>
<tr>
<td>mode</td>
<td>(enum NAOutputVoltageMode )</td>
</tr>
<tr>
<td>naWaitEOS</td>
<td>- While in this mode any voltage changes sent to the selected analog out will only get applied to the output between sweeps.</td>
</tr>
<tr>
<td>naNoWait</td>
<td>- While in this mode any voltage changes sent to the selected analog out will occur right away without waiting until the end of a sweep, the voltage gets applied immediately.</td>
</tr>
<tr>
<td>Return Type</td>
<td>NAOutputVoltageMode</td>
</tr>
<tr>
<td>Default</td>
<td>naWaitEOS</td>
</tr>
<tr>
<td>Examples</td>
<td><code>auxIo.put_OutputVoltageMode 1, naWaitEOS 'Write</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT put_OutputVoltageMode(VARIANT Output, tagNAOutputVoltageMode dNewMode);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IHWAuxIO</td>
</tr>
</tbody>
</table>
**put_Port Method**

**Description**
Writes a value to the specified port. Use the `get_Port` Method to read the settings from the "readable" ports (C, D, E).

**VB Syntax**
```vb
handlerIo.put_Port (port, value)
```

**Variable**

- **handlerIo** *(object)* - A HandlerIO object
- **port** *(enum as NAMatHandlerPort)* - port to put data into. Choose from:
  - `naPortA` - (0)
  - `naPortB` - (1)
  - `naPortC` - (2)
  - `naPortD` - (3)
  - `naPortE` - (4)
  - `naPortF` - (5)
  - `naPortG` - (6)
  - `naPortH` - (7)
- **value** The number of the data bits to set. The following table shows what the `value` represents:

  **Note:** When writing to port G, port C must be set to output mode
  When writing to port H, both port C and port D must be set to output mode. Use Port Mode Property
<table>
<thead>
<tr>
<th>Port</th>
<th>Max allowable &lt;num&gt;</th>
<th>MSB...........................................LSBexplode</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>255</td>
<td>A7...A0</td>
<td>Write-only</td>
</tr>
<tr>
<td>B</td>
<td>255</td>
<td>B7...B0</td>
<td>Write-only</td>
</tr>
<tr>
<td>C</td>
<td>15</td>
<td>C3...C0</td>
<td>Read-Write</td>
</tr>
<tr>
<td>D</td>
<td>15</td>
<td>D3...D0</td>
<td>Read-Write</td>
</tr>
<tr>
<td>E</td>
<td>255</td>
<td>D3...D0 + C3...C0</td>
<td>Read-Write</td>
</tr>
<tr>
<td>F</td>
<td>65535</td>
<td>B7...B0 + A7...A0</td>
<td>Write-only</td>
</tr>
<tr>
<td>G</td>
<td>1048575</td>
<td>C3...C0 + B7...B0 + A7...A0</td>
<td>Write-only</td>
</tr>
<tr>
<td>H</td>
<td>16777215</td>
<td>D3...D0 + C3...C0 + B7...B0 + A7...A0</td>
<td>Write-only</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
handlerIo.put_Port(naPortB, 1)

**C++ Syntax**
HRESULT put_Port ( tagNAMatHandlerPort Port, VARIANT Data );

**Interface**
IHWMaterialHandlerIO
Write-only

**put_PortCData Method**

**Description**
Writes a 4-bit value to Port C on the Aux I/O connector (pins 22-25) and the Material Handler IO (pins 21-24 Anritsu) - (pins 22-25 Avantest).

**Note:** These lines are connected to both the Handler IO and Aux IO in the VNA. Therefore, this command will affect both of these connectors in the same way.

**VB Syntax**

```
AuxIO.put_PortCData num
```

**Variable**

- **AuxIO** *(object)* - A Hardware Auxiliary Input / Output object
- **num** *(variant)* - 4 bit binary value. Choose from 0-15

**Return Type**
None

**Default**
None

**Examples**

```
HWAuxIO.put_PortCData 15  'If Positive Logic, Port C lines C0, C1, C2, C3 go High. If Negative Logic, they go Low.
```

**C++ Syntax**

```
HRESULT put_PortCData( VARIANT Data );
```

**Interface**
IHWAuxIO
PutDataScalar Method

Description

Puts formatted variant scalar data into the measurement result buffer. The data will be immediately processed and displayed. Subsequent changes to the measurement state will be reflected on the display.

Always precede this command by setting the format on the measurement to be consistent with the format of the data being sent to the analyzer. In this way, the display annotation will be correct.

Execution of this command does not change the display format.

VB Syntax

`meas.putDataScalar format, data`

Variable (Type) - Description

meas A measurement (object)

format (enum NADataFormat) Format of the data. This value is presently ignored by the VNA. Data is always presented in the current format.

Choose from:

0 - naDataFormat_LinMag
1 - naDataFormat_LogMag
2 - naDataFormat_Phase
3 - naDataFormat_Polar
4 - naDataFormat_Smith
5 - naDataFormat_Delay
6 - naDataFormat_Real
7 - naDataFormat_Imaginary
8 - naDataFormat_SWR
9 - naDataFormat_PhaseUnwrapped
10 - naDataFormat_InverseSmith
11 - naDataFormat_Kelvin
12 - naDataFormat_Fahrenheit
13 - naDataFormat_Centigrade

**Notes:**

- The `getData` (variant) method includes a "format" argument, which allows scalar (one-dimensional) data. To put data back into the "raw" data buffer using this (putDataComplex) method, specify Polar format when using the `getData` method.

- **Phase** format accepts data in radians (not degrees) and displays in degrees. To convert to degrees: radians * (57.29577951308233) = degrees. The `getData` method returns degrees if the request is for phase data.

`data` (variant) - A 1-dimension array of single precision floating point numbers.

**Return Type** Not Applicable

**Default** Not Applicable

**Examples**

```
' Put 201 points worth of scalar data into the measurement
' 200 is max index, so 0 to 200 is 201 points
Dim data(200) ' array of 201 (scalar) data points
' Fill the array
For i = 0 to 200
data(i) = i/200
Next
app.ActiveMeasurement.putDataScalar 0, data
```

**C++ Syntax**

```cpp
HRESULT putDataScalar(tagNADataStore DataStore, VARIANT scalarArray)
```

**Interface** I Measurement
PutShortcut Method

Description
Defines a Macro (shortcut) file in the analyzer. This command links a file name and path to the Macro file. The file must be put in the VNA at the location indicated by this command.

VB Syntax

```vbnet
app.PutShortcut index,title,path, arguments
```

Variable

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>An Application (object)</td>
<td></td>
</tr>
<tr>
<td>index</td>
<td>(long)</td>
<td>Number of the macro to be stored in the analyzer. Use a number between 1 and 25. If the index number already exists, the existing macro is replaced with the new macro.</td>
</tr>
<tr>
<td>title</td>
<td>(string)</td>
<td>The name to be assigned to the macro</td>
</tr>
<tr>
<td>path</td>
<td>(string)</td>
<td>Full path, file name, and extension of the existing macro &quot;executable&quot; file.</td>
</tr>
<tr>
<td>arguments</td>
<td>(string)</td>
<td>Arguments that may be required for the specified macro to run.</td>
</tr>
</tbody>
</table>

Return Type
Not Applicable

Default
Not Applicable

Examples
```vbnet
app.PutShortcut 1,"Test","C:/Automation/MyTest.vbs",""
```

C++ Syntax

```cpp
HRESULT PutShortcut(long Number, BSTR title, BSTR path, BSTR arguments)
```

Interface
IApplication
putSourcePowerCalDataEx Method

Description

**Note:** This method replaces putSourcePowerCalData Method

Inputs source power calibration data (as variant data type) to this channel for a specific source port.

The effect from this command on the channel is immediate. Do NOT send ApplyPowerCorrectionValuesEX after this command as it may invalidate the uploaded data.

If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the Get X-Axis Values command to return the X-axis values in the displayed order.

The calibration is not valid if the current number of points on the channel is not equal to the number of values that were input.

**Note:** This method sends variant data which is less efficient than methods available on the ISourcePowerCalData interface.

VB Syntax

```vbnet
chan.putSourcePowerCalDataEx buffer, srcPort, data
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chan</td>
<td>(object) – A Channel object</td>
</tr>
<tr>
<td>buffer</td>
<td>(enum NASourcePowerCalBuffer) - The source power cal data buffer to write to.</td>
</tr>
<tr>
<td>srcPort</td>
<td>(long integer) – The source port for which calibration data is being requested.</td>
</tr>
<tr>
<td>data</td>
<td>(variant) – Array of source power cal data being input.</td>
</tr>
</tbody>
</table>

Return Type  None

Default Not Applicable

Examples

```vbnet
chan.putSourcePowerCalDataEx naCorrectionValues, 1, varData
```
C++ Syntax  HRESULT putSourcePowerCalDataEx(tagNASourcePowerCalBuffer bufSelect, long sourcePort, VARIANT varData);

Interface  IChannel4
putSourcePowerCalDataScalarEx Method

Description

Note: This method replaces putSourcePowerCalDataScalar Method

Inputs source power calibration data (as scalar values) to this channel for a specific source port.

The effect from this command on the channel is immediate. Do NOT send ApplyPowerCorrectionValuesEX after this command as it may invalidate the uploaded data.

If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the Get X-Axis Values2 command to return the X-axis values in the displayed order.

VB Syntax

```vbnet
chanData.putSourcePowerCalDataScalarEx buffer, srcPort, numValues, data
```

Variable (Type) - Description

**chanData** (interface) – An ISourcePowerCalData2 interface on the Channel (object)

**buffer** (enum NASourcePowerCalBuffer) - The source power cal data buffer to write to.

- **0 - naCorrectionValues** This is the only buffer currently available.

**srcPort** (long integer) – The source port for which calibration data is being input.

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source VNA-x model, then you must use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port.

**numValues** (long integer) – Number of data values being input.

Note: If this does not equal the current number of points on the channel, the calibration will not be valid.

**data** (single) – Array of source power cal data being input.

Return Type

None

Default

Not Applicable
Dim chanData As ISourcePowerCalData2
Set chanData = app.ActiveChannel
chanData.putSourcePowerCalDataScalarEx naCorrectionValues, 1, 201, scalarCalValues(0)

HRESULT putSourcePowerCalDataScalarEx(tagNASourcePowerCalBuffer bufSelect, long sourcePort, long numValues, float *pData);

Interface ISourcePowerCalData2
PutStandard Method  **Superseded**

**Description**

This command is replaced with PutStandardByString

Puts standard acquisition data into the Cal Set.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data.

**VB Syntax**

`CalSet.putStandard class, rcv, src, data`

**Variable (Type) - Description**

*CalSet* (object) - A Cal Set object

*class* (enum NACalClass) Standard. Choose from:

1 - naClassA
2 - naClassB
3 - naClassC
4 - naClassD
5 - naClassE
6 - naReferenceRatioLine
7 - naReferenceRatioThru

**SOLT Standards**

1 - naSOLT_Open
2 - naSOLT_Short
3 - naSOLT_Load
4 - naSOLT_Thru
5 - naSOLT_Isolation

**TRL Standards**

1 - naTRL_Reflection
2 - naTRL_Line_Reflection
3 - naTRL_Line_Tracking
4 - naTRL_Thru
5 - naTRL_Isolation

*rcv* (long) - Receiver Port
**src** (long) - Source Port

**data** (variant) Error term data in a two-dimensional array (0:1, 0:numpts-1). The data must be complex pairs.

**Note:** See also Put Standard Complex on the ICalData2 interface to avoid using the variant data type.

**Return Type** Not Applicable

**Default** Not Applicable

**Examples** See an Example

**C++ Syntax** HRESULT putStandard(tagNACalClass stdclass, long ReceivePort, long SourcePort, VARIANT varData)

**Interface** ICalSet
PutStandardByString

**Description**  Puts standard acquisition data into the Cal Set.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data.

**VB Syntax**  

```
PutStandardByString(stdName, vdata)
```

**Variable**

**stdName**  (String)  The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be "S11C(3,3)".

**vdata**  (Variant)  The variant containing a safearray of variants. This data is usually two dimensional.

**Note:** The vardata array is a safearray of variants wrapped in a variant. This structure is compatible with scripting clients who can only use variants. For alternative methods that used typed arrays, see ICalData3.

**Return Type**  Not Applicable

**Default**  Not Applicable

**Examples**  See an Example

**C++ Syntax**  

```
HRESULT PutStandardByString(BSTR bufferName, VARIANT vdata);
```

**Interface**  ICalSet2
PutStandardComplex Method  **Superseded**

**Description**
This command is replaced with **PutStandardComplexByString**

Puts standards acquisition data into the Cal Set.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

**VB Syntax**

```
ICalData2.putStandardComplex class, rcv, src, numPts, real(), imag()
```

**Variable**

- **ICalData2** An ICalData2 pointer to the Cal Set object
- **class** (enum NACalClass) Standard. Choose from:
  1 - naClassA
  2 - naClassB
  3 - naClassC
  4 - naClassD
  5 - naClassE
  6 - naReferenceRatioLine
  7 - naReferenceRatioThru

**SOLT Standards**
  1 - naSOLT_Open
  2 - naSOLT_Short
  3 - naSOLT_Load
  4 - naSOLT_Thru
  5 - naSOLT_Isolation

**TRL Standards**
  1 - naTRL_Reflection
  2 - naTRL_Line_Reflection
  3 - naTRL_Line_Tracking
  4 - naTRL_Thru
  5 - naTRL_Isolation

- **rcv** (long) - Receiver Port
**src** (long) - Source Port

**numPts** (long) - The number of data points in the real and imaginary arrays.

**real()** (single) - one-dimensional array containing the real part of the acquisition data. (0:points-1)

**imag()** (single) - one-dimensional array containing the imaginary part of the acquisition data. (0:points-1)

**Return Type** Not Applicable

**Default** Not Applicable

**Examples**

```cpp
Dim sdata As ICalData2
Set sdata = calmanager.CreateCal Set( 1 )
sdata.putStandardComplex naSOLT_Open, 1, 1, numpts, rel(0),
    img(0)
```

**C++ Syntax**

```cpp
HRESULT putStandardComplex(tagNACalClass stdclass, long ReceivePort,
    long SourcePort, long lNumValues, float* pReal, float* pImag)
```

**Interface** ICalData2
PutStandardComplexByString

**Description**

Puts standard acquisition data into the Cal Set.

Learn more about [Reading and Writing Cal Data](#).

See examples of [Reading and Writing Cal Set Data](#).

**VB Syntax**

```vbnet
ICalData3.PutStandardComplexByString(stdName, lnumPoints, real(o), imag(0))
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalData3</td>
<td>An ICalData3 pointer to a Cal Set object.</td>
</tr>
<tr>
<td>stdName</td>
<td>(String) The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be &quot;S11C(3,3)&quot;.</td>
</tr>
<tr>
<td>lnumpoints</td>
<td>(long) - The number of data points in the real and imaginary arrays.</td>
</tr>
<tr>
<td>real</td>
<td>(Single) The real component of the complex data.</td>
</tr>
<tr>
<td>imag</td>
<td>(Single) The imaginary component of the complex data.</td>
</tr>
</tbody>
</table>

**Return Value**

Single

**Default**

Not Applicable

**Examples**

[See an Example](#)

**C++ Syntax**

```c++
HRESULT PutStandardComplexByString(BSTR bufferName, long lnumPoints, float* real, float* imag);
```

**Interface**

ICalData3
## Quit Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Terminates the Network Analyzer application.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.Quit</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>app.Quit</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Quit()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>Remarks</strong></td>
<td>Under the rules of COM, the server should not exit until all references to it have been released. This method is a brute force way of terminating the application. Be sure to release all references (or terminate the client program) before attempting to restart the Network Analyzer application. An alternate approach to terminating the application is to make the application invisible (app.Visible = False) and release all references. The server will shutdown.</td>
</tr>
</tbody>
</table>
## RangeCount Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of frequency ranges in the DIQ channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>count = DIQ.RangeCount</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>count</td>
<td>(Long) Number of frequency ranges in the channel.</td>
</tr>
<tr>
<td>DIQ</td>
<td>A DIQ Object</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>value=Diq.RangeCount</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT RangeCount(long* count);</td>
</tr>
<tr>
<td>Interface</td>
<td>IDIQ</td>
</tr>
</tbody>
</table>
### ReadBinary Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns data from the VISA pass-through device as a Safe Array of variants. Note that binary data transfers over sockets are not supported.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = Vpassthru.ReadBinary(visaID)</code></td>
</tr>
</tbody>
</table>
| **Variable** | *(Type) - Description*  
  *value (variant) - Variable to store the returned data.*  
  *Vpassthru (object) - A VISAPassthrough object*  
  *visaID (long) VISA session number (see Open method).* |
| **Return Type** | Safe Array of variants |
| **Default** | Not Applicable |
| **Examples** | `value=Vpassthru.ReadBinary(1)` |
| **C++ Syntax** | `HRESULT ReadBinary(long session_num)` |
| **Interface** | IVISAPassthrough |
## ReadBinaryCompact Method

**Description**
Reads binary data the same as the `ReadBinary` method, but returns the data in a more compact form of Safe Array. This is significantly faster but is not supported in all client environments. Note that binary data transfers over sockets are not supported.

**VB Syntax**

```vbnet
value = Vpassthru.ReadBinaryCompact(visaID)
```

**Variable**

- `value` *(variant)* - Variable to store the returned data.
- `Vpassthru` *(object)* - A VISAPassthrough object
- `visaID` *(long)* - VISA session number (see Open method).

**Return Type**
Safe Array of variants

**Default**
Not Applicable

**Examples**

```vbnet
value = Vpassthru.ReadBinaryCompact(1)
```

**C++ Syntax**

```cpp
HRESULT ReadBinaryCompact(long session_num)
```

**Interface**
IVISAPassthrough
About the ExtTestSetIO connector

ReadData Method

Description  Reads a 13-bit data word from the specified address. Data is read using the AD0 through AD12 lines of the external test set connector. The instrument generates the appropriate timing signals. It automatically controls timing signals LDS, LAS and RLW to strobe the address, and then read the data, from the external test set. See the timing diagram for Address and Data I/O read.

VB Syntax  

    value = ExtIO.ReadData (address)

Variable  

    value  (variant) - Variable to store the returned data
    ExtIO  (object) - An ExternalTestSetIO object
    address  (variant) - address to read data from.

Return Type  Variant

Default  Not Applicable

Examples  

    value = ExtIO.ReadData (15)

C++ Syntax  

    HRESULT ReadData (VARIANT Address, VARIANT* Data);

Interface  IHWExternaTestSetIO
ReadRaw Method

Description
Reads a 16-bit value from the external test set. The 16-bit value is comprised of lines AD0 - AD12, Sweep Holdoff In and Interrupt In (inverted).

When this command is used the analyzer does NOT generate the appropriate timing signals; it simply reads the lines. The user needs to first use the WriteRaw method to do the initial setup. The RLW line (pin25) must be set to the appropriate level in order to read the test set connected.

Below is the format of data that is read with ReadRaw:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Bit</th>
<th>Signal name</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>0</td>
<td>AD0*</td>
</tr>
<tr>
<td>23</td>
<td>1</td>
<td>AD1*</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>AD2*</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>AD3*</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>AD4*</td>
</tr>
<tr>
<td>21</td>
<td>5</td>
<td>AD5*</td>
</tr>
<tr>
<td>20</td>
<td>6</td>
<td>AD6*</td>
</tr>
<tr>
<td>19</td>
<td>7</td>
<td>AD7*</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>AD8*</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>AD9*</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>AD10*</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>AD11*</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>AD12*</td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>Sweep Holdoff In</td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>Interrupt In (inverted internally)</td>
</tr>
<tr>
<td>na</td>
<td>15</td>
<td>Always Zero, grounded internally</td>
</tr>
</tbody>
</table>
*These lines are dependent on the state of RLW (pin25). Writing a 0(low) to RLW will set lines AD0-AD12 to write mode. Writing a 1(high) to RLW will set lines AD0-AD12 to read mode.

**VB Syntax**

```
value = ExtIO.ReadRaw (address)
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(variant) - Variable to store the returned data</td>
</tr>
<tr>
<td>ExtIO</td>
<td>(object) - An External IO object</td>
</tr>
<tr>
<td>address</td>
<td>(variant) - Address to read data from</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Real</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**Examples**

```
value = ExtIO.ReadRaw (address)
```

**C++ Syntax**

```c++
HRESULT ReadRaw( VARIANT* Input );
```

**Interface**

IHWEternalTestSetIO
### ReadString Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns data from the VISA pass-through device.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = Vpassthru.ReadString(visaID)</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(string) - Variable to store the returned string data.</td>
</tr>
<tr>
<td>Vpassthru</td>
<td>(object) - A VISAPassthrough object</td>
</tr>
<tr>
<td>visaID</td>
<td>(long) VISA session number (see Open method).</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value=Vpassthru.ReadString(1)</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT ReadString(long session_num)</td>
</tr>
<tr>
<td>Interface</td>
<td>IVISAPassthrough</td>
</tr>
</tbody>
</table>
# ReCalculate Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Repeats the last calculation that was performed, including all ON (state) segments in segment table.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>conv.ReCalculate()</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>conv</code></td>
<td>A Converter Object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mxr.ReCalculate()</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT ReCalculate()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IConverter5</td>
</tr>
</tbody>
</table>

See example program
Recall Method

Description: Recalls a measurement state, calibration state, or both, from the hard drive into the analyzer.

Use app.Save to save files.

VB Syntax: app.Recall (filename.ext)

Variable (Type) - Description

- app: An Application (object)
- filename.ext: (string) - Full path, file name, and extension, of the file.

Files are typically stored in "D:" Use one of the following extensions:

- .sta - Instrument State
- .cal - Calibration file
- .cst - Both Instrument State and Calibration reference
- .cti - Citifile (data will always be formatted. See Recalling Citifiles Using the VNA)
- .csa - Instrument state and calibration data (not a reference pointer).

Return Type: Not Applicable

Default: Not Applicable

Examples: app.Recall ("D:\MyState.cst") 'Recalls "mystate.cst" from the specified folder

C++ Syntax: HRESULT Recall(BSTR bstrFile)

Interface: IApplication
## Recall Kits Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Recalls the calibration kits definitions that were stored with the SaveKits command.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.RecallKits</code></td>
</tr>
</tbody>
</table>

### Variable

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
</tbody>
</table>

### Return Type

Not Applicable

### Default

Not Applicable

<table>
<thead>
<tr>
<th>Examples</th>
<th><code>app.RecallKits</code></th>
</tr>
</thead>
</table>

### C++ Syntax

`HRESULT RecallKits()`

### Interface

`IApplication`
Recall Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Loads an uncertainty ‘workspace’ (*.ml4) file into the Uncertainty Manager.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>uncertMan.Recall(fileName)</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>uncertMan</code></td>
<td>An UncertaintyManager Object</td>
</tr>
<tr>
<td><code>fileName</code></td>
<td>(String). Filename and *.ml4 extension of the Uncertainty Manager workspace file to load.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>uncertMan.Recall&quot;MyWorkspace.ml4&quot;</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT Recall([in] BSTR fileName);</td>
</tr>
<tr>
<td>Interface</td>
<td>IUncertaintyManager</td>
</tr>
</tbody>
</table>

See example program
Remove Method

**Description**
Removes an item from a collection of objects.

**VB Syntax**
```
Object.Remove item
```

**Variable**
*(Type) - Description*

**Object**
Any of the following *(objects)*

- CalFactorSegments collection
- CalFactorSegmentsPMAR Collection
- Cal Sets collection
- Channels Collection
- ExternalDevices Collection
- GuidedCalibrationPowerSensors Collection
- Measurements collection
- NAWindows collection
- PowerLossSegments collection
- PowerLossSegmentsPMAR Collection
- Segments collection

**Note:** Segments, CalFactorSegments, and PowerLossSegments have an OPTIONAL argument `[size]` referring to the number of segments to remove, starting with the *item* parameter.

**Note:** Segments - When ALL segments are deleted, SweepType is automatically set to Linear because there are no segments to sweep.

**item** *(variant)* - Collection Item number to be removed.

**Note:** The ExternalDevices Collection requires that you specify *item* as the string name of the device. For example:
```
extDevices.Remove ('mySource')
```

**Return Type**
Not Applicable

**Default**
Not Applicable
Examples

Measurements.Remove 3 'Removes the third measurement in the collection

segments.Remove 2,20 'Removes 20 segments (2 - 21)

C++ Syntax

HRESULT Remove(VARIANT index); //Measurements
HRESULT Remove(VARIANT index); //Cal Sets
HRESULT Remove(long windowNumber); //NAWindows
HRESULT Remove(VARIANT index, long size); //Segments
HRESULT Remove(VARIANT index, long size); //CalFactorSegments(PMAR)
HRESULT Remove(VARIANT index, long size); //PowerLossSegments(PMAR)
HRESULT Remove(BSTR name) //ExternalDevices
HRESULT Remove(VARIANT index) //Channels - specify collections index, not the channel number.
HRESULT Remove(VARIANT index); //GuidedCalibrationPowerSensors

Interface

All listed above
## RemoveAll Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Removes ALL power sensors from the GuidedCalibrationPowerSensors Collection</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>guidedSensors.RemoveAll</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>guidedSensors</code></td>
<td>A GuidedCalibrationPowerSensors Collection</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>sensors.RemoveAll</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT RemoveAll();</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IGuidedCalibrationPowerSensors</td>
</tr>
<tr>
<td>Description</td>
<td>Deletes a channel by specifying the channel number. Use <strong>Remove Method</strong> to delete a channel by specifying the index in the channels collection.</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chans.RemoveChannelNumber(chan)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>chans</code></td>
<td>A <strong>Channels (collection)</strong></td>
</tr>
<tr>
<td><code>chan</code></td>
<td>The channel number to delete.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chan.RemoveChannelNumber(2)</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT RemoveChannelNumber(VARIANT channelNumber)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannels3</td>
</tr>
</tbody>
</table>
RemovItem Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Removes a name-value pair from the Cal Set. Send the Save (CalSet) Method to save the edited CalSet to the VNA.</th>
</tr>
</thead>
</table>

**See Also**

EnumerateItems Method

Item Property (Learn about name-value pairs)

**VB Syntax**

```vbnet
CalSet.RemoveItem(name)
```

**Variable**

- **Type** - Description
  - `CalSet` - A CalSet object
  - `name` - (String) Name of the item.

**Return Type**

Not Applicable

**Default**

Not Applicable

**Examples**

See example

**C++ Syntax**

```c++
HRESULT RemoveItem (VARIANT* itemNames);
```

**Interface**

ICalSet6
### RemoveLibrary Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Removes an imported an Equation Editor DLL from the VNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>equation.RemoveLibrary location</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>equation</code></td>
<td>A <code>MeasurementEquation</code> object</td>
</tr>
<tr>
<td><code>location</code> (string)</td>
<td>Full path and filename of the *.dll to be removed.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>equation.RemoveLibrary &quot;C:\Program Files(x86)\Keysight\Network Analyzer\UserFunctions\Expansion.dll&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT RemoveLibrary(BSTR filename);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IMeasurementEquation</code></td>
</tr>
</tbody>
</table>
### Reset Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Removes all existing windows and measurements from the application. (Unlike <em>Preset</em>, does not create a new measurement.)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.Reset</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>app.Reset</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Reset()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>
## Reset Method (Cal All)

<table>
<thead>
<tr>
<th>Description</th>
<th>Resets all properties associated with the Cal All session to their default values.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>calAll.Reset</code></td>
</tr>
</tbody>
</table>
| Variable             | *calAll* A *CalibrateAllChannels* *
| (Type) - Description | object                                                                         |
| Return Type          | Not Applicable                                                                  |
| Default              | Not Applicable                                                                  |
| Examples             | `calAll.Reset`                                                                  |
| C++ Syntax           | HRESULT Reset()                                                                 |
| Interface            | ICalibrateAllChannels                                                           |

---

2932
### Reset (Independent CalAll) Method

<table>
<thead>
<tr>
<th>Description</th>
<th>This command resets all ranges for the given source port (&lt;n&gt;).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>IndependentPwrrCalPort. Reset</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>IndependentPwrrCalPort</code></td>
<td>A <code>IndependentPower CalibrationPort</code> <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>CalibrateAllChannels.IndependentPowerCalibrationPort(3).Reset</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Reset</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IIndependentPowerCalibrationPort</td>
</tr>
</tbody>
</table>
### Reset Method (PhaseRef)

<table>
<thead>
<tr>
<th>Description</th>
<th>Resets all properties associated with the Phase Reference Cal to their default values. For predictable results, send this command before performing each Phase Reference Cal.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><em>phaseRef</em>.Reset</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><em>phaseRef</em></td>
<td>A PhaseReferenceCalibration <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><em>phaseRef</em>.Reset()</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Reset()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPhaseReferenceCalibration</td>
</tr>
</tbody>
</table>
## Reset (Power Range) Method

**Description**  
Resets all PowerRange properties to default values, as if the instrument had been preset. Power range type is set to "Specified", port number is set to "1" with all path configuration elements in their default states.

**VB Syntax**  
`powerRange.Reset`

**Variable**  
*(Type)* - Description

`powerRange` *(object)* - A PowerRange object

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`powerRange.Reset`

**C++ Syntax**  
`HRESULT Reset()`  
Interface  
IPowleRange
# Reset Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Closes all currently open VISA sessions.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>$Vpassthru.Reset$</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>($Type$) - Description</td>
</tr>
<tr>
<td>$Vpassthru$</td>
<td>(object) - A VISA Passthrough object</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>$Vpassthru.Reset$</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Reset()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IVISAPassthrough</td>
</tr>
</tbody>
</table>
## ResetLOFrequency Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Resets the LO Delta Frequency to 0 (zero) Hz.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>obj.ResetLOFrequency</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>obj</code></td>
<td>An EmbeddedLO (object) or</td>
</tr>
<tr>
<td></td>
<td>A ConverterEmbeddedLO (object)</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>embedLO.ResetLOFrequency</code> <code>write</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT ResetLOFrequency();</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IEmbeddedLO</td>
</tr>
</tbody>
</table>
### ResetNoise Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Resets ( clears) the characterized noise data for the VNA port object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>oPort.ResetNoise()</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>oPort</code></td>
<td>A Port Object</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>oPort.ResetNoise()</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT ResetNoise();</td>
</tr>
<tr>
<td>Interface</td>
<td>IUncertaintyPort</td>
</tr>
</tbody>
</table>

See example program
### ResetNoiseForAllPorts Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Resets (clears) the characterized noise data for ALL VNA port objects in the Ports collection.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>$oPorts.ResetNoiseForAllPorts()$</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>$oPorts$ (Type) - Description</td>
</tr>
<tr>
<td>$oPorts$</td>
<td>A Ports Collection</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>$oPorts.ResetNoiseForAllPorts()$</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT ResetNoiseForAllPorts();</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IUncertaintyPorts</td>
</tr>
</tbody>
</table>
**Write-only**

### ResetPortValues Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Resets the full and response list to their default values. <a href="#">Learn more.</a></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>corrMethods.ResetPortValues()</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>corrMethods</code></td>
<td><code>CorrectionMethods (object)</code></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>corrMethods.ResetPortValues()</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT ResetPortValues();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>ICorrectionMethods2</code></td>
</tr>
</tbody>
</table>
### ResetRepeatability Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Resets (clears) the characterized repeatability data associated with the cable object.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>cable.ResetRepeatability()</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>cable</code></td>
<td>A Cable Object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>cable.ResetRepeatability()</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT ResetRepeatability();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IUncertaintyCable</td>
</tr>
</tbody>
</table>
Write-only

About Display Colors

ResetTheme Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Resets the current theme to the default VNA colors.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>colors.ResetTheme()</td>
</tr>
<tr>
<td>Variable</td>
<td>colors</td>
</tr>
<tr>
<td>(Type) - Description</td>
<td>A ComColors (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>colors.ResetTheme()</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT ResetTheme();</td>
</tr>
<tr>
<td>Interface</td>
<td>IColors</td>
</tr>
</tbody>
</table>
# ResetTuningParameters Method

**Description**  
Resets the tuning parameters to their default values.

**VB Syntax**  
`obj.ResetTuningParameters`

**Variable**  
(Description)

- `obj`  
  An EmbeddedLO (object) or
  A ConverterEmbeddedLO (object)

**Default**  
Not Applicable

**Examples**  
`embedLO.ResetTuningParameters`  

**C++ Syntax**  
`HRESULT ResetTuningParameters();`

**Interface**  
IEmbededLO
## RestoreCalKitDefaults Method

**Description**
Restores the original properties of the specified Cal Kit, overwritting the last definition with the factory defaults.

**NOTE:** ONLY works with VNA releases 1.0 through 1.6.

**VB Syntax**
```vbnet
app.RestoreCalKitDefaults (calKit)
```

**Variable (Type) - Description**

- **app** An Application (object)
- **calKit** (enum NAcalKit) - Calibration Kit to restore. Choose from:

  1 - naCalKit_85032F_N50
  2 - naCalKit_85033E_3_5
  3 - naCalKit_85032B_N50
  4 - naCalKit_85033D_3_5
  5 - naCalKit_85038A_7_16
  6 - naCalKit_85052C_3_5_TRL
  7 - naCalKit_User7
  8 - naCalKit_User8
  9 - naCalKit_User9
  10 - naCalKit_User10

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vbnet
app.RestoreCalKitDefaults naCalKit_MechKit10
```

**C++ Syntax**
```cpp
HRESULT RestoreCalKitDefaults/(tagNACalKit kit)
```

**Interface**
IApplication
### RestoreCalKitDefaultsAll Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Restores the original properties of ALL of the Cal Kits, overwriting the last definitions with the factory defaults.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NOTE:</strong></td>
<td>ONLY works with VNA releases 1.0 through 1.6.</td>
</tr>
</tbody>
</table>

#### VB Syntax
```
app.RestoreCalKitDefaultsAll
```

#### Variable
- **(Type)** - Description
- **app** - An Application (object)

#### Return Type
- Not Applicable

#### Default
- Not Applicable

#### Examples
```
app.RestoreCalKitDefaultsAll
```

#### C++ Syntax
```
HRESULT RestoreCalKitDefaultsAll()
```

#### Interface
- IApplication
### RestoreDefaults Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Resets the VNA preferences to their factory default settings.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pref.RestoreDefaults</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>pref</code></td>
<td>A Preferences (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>pref.RestoreDefaults</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT RestoreDefaults()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IPreferences9</code></td>
</tr>
</tbody>
</table>
### Resume Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Resumes the trigger mode of all channels that was in effect before sending the <code>channels.Hold</code> method. Channels.Hold must be sent before channels.Resume, using the same instance of the Channels object.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chans.Resume</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>chans</code></td>
<td>A Channel collection <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chans.Resume</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT Resume();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannels2</td>
</tr>
</tbody>
</table>
Save Method

Description
Saves the appropriate content to the hard drive depending on the extension that is provided.

Some saved files can be recalled using app.Recall. depending on the content.

VB Syntax
app.Save(filename.ext)

Variable (Type) - Description
app An Application (object)
filename.ext (string) - Full path, file name, and extension of the file.

Files are typically stored in "D:".
Use one of the following extensions:

- .cst - Saves both Instrument State and Cal Set reference - Recalls a calibrated measurement. (Recallable)
- .sta - Saves Instrument State only - recalls the instrument state without calibration. (Recallable)
- .cal - Calibration file – saves the active Cal Sets currently in use by any channel. Use this mode for archival purposes only. All Cal Sets are saved to a Cal Set data file. This mode provides a method of safeguarding calibration data. This data can be restored to the list of Cal Sets available in the instrument. (Recallable)
- .csa - Saves both instrument state AND actual calibration data, not a reference pointer to the Cal Set.
- .prn - Saves active trace in comma-separated format (not recallable)
- .bmp - Saves a Bitmap of the screen (not recallable)
- .s1p - Saves 1-port measurement data
- .s2p - Saves 2-port measurement data
- .s3p - Saves 3-port measurement data
- .s4p - Saves 4-port measurement data

Return Type Not Applicable
Default Not Applicable
<table>
<thead>
<tr>
<th>Examples</th>
<th><code>app.Save(&quot;C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Newfolder\MyState.cst&quot;)</code> 'Saves &quot;mystate.cst&quot; to the specified folder'</th>
</tr>
</thead>
<tbody>
<tr>
<td>C++ Syntax</td>
<td>HRESULT Save(BSTR bstrFile)</td>
</tr>
<tr>
<td>Interface</td>
<td>IApplication</td>
</tr>
</tbody>
</table>
### Save Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves the current Cal Set to disk. This is the recommended method for saving a Cal Set.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn more about <a href="#">reading and writing Cal data using COM</a></td>
<td></td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>CalSet.Save</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>CalSet</code> <em>(object)</em></td>
<td>A <code>CalSet</code> object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>myCalSet.Save</code></td>
</tr>
<tr>
<td>See <a href="#">Copy Method</a> for an example application of this command.</td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT Save();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalSet</td>
</tr>
</tbody>
</table>
This command is replaced by ICalSet::Save which saves the data for only the current Cal Set to the disk.

Writes new or changed Cal Sets to disk. All Cal Sets are saved in a single file. This file is updated at the following times:

- When a Cal Set has been deleted.
- When a calibration has been performed through the front panel interface.
- When this method is called.
- When ICalSet::Save is called.

Learn more about reading and writing Cal data using COM

**VB Syntax**  
`object.SaveCalSets`

**Variable (Type) - Description**

- `object` *(object)* - A CalManager object or a Calibrator object

**Return Type**  
None

**Default**  
Not Applicable

**Example**  
`calMgr.SaveCalSets`

**C++ Syntax**  
`HRESULT SaveCalSets();`

**Interface**  
ICalManager  
ICalibrator
SaveCitiDataMethod - **Superseded**

**Description**
This command is replaced with **SaveData Method**

Saves UNFORMATTED trace data to .cti file. [Learn more about citifiles](#).

**VB Syntax**
```vbnet
app.SaveCitiDataData(filename.cti)
```

**Variable**

- `app` (Type) - Description
  - **app** An Application (object)

- `filename.cti` (Type) - Full path, file name, and .cti extension of the file.
  - Files are typically stored in "D:\".

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vbnet
app.SaveCitiDataData("C:\Program Files(x86)\Keysight\Network Analyzer\Documents\myDDCitifile.cti") 'Saves "myDDCitifile.cti" to the specified folder
```

**C++ Syntax**
```cpp
HRESULT SaveCitiDataData (BSTR bstrFile)
```

**Interface**
IApplication5
**SaveCitiFormattedData Method - Superseded**

<table>
<thead>
<tr>
<th>Description</th>
<th>This command is replaced with SaveData Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Saves FORMATTED trace data to .cti file. Learn more about citifiles.</td>
<td></td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
app.SaveCitiFormattedData(filename.cti)
```

**Variable**

- **app** (Type) - Description
  - An Application (object)

- **filename.cti** (string) - Full path, file name, and .cti extension of the file.

  Files are typically stored in "D:\"

**Return Type**

Not Applicable

**Default**

Not Applicable

**Examples**

```vbnet
app.SaveCitiFormattedData ("C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Newfolder\myFDCitifile.cti") 'Saves "myFDCitifile.cti" to the specified folder
```

**C++ Syntax**

```cpp
HRESULT SaveCitiFormattedData (BSTR bstrFile)
```

**Interface**

IApplication5
**SaveData Method**

**Description**
Stores trace data to the following file types: *.prn, *.cti, *.csv, *.mdf

To save snp files, use `WriteSnpFileWithSpecifiedPorts`

To save instrument state and calibration files, use `Save`.

This command replaces the following:

- `SaveCitiDataData Method`
- `SaveCitiFormattedData Method`
- `CitiContents Property`
- `CitiFormat Property`

Some saved files can be recalled using `app.Recall`, depending on the content.

**VB Syntax**

```
app.SaveData filename, type, scope, format, selector
```

**Variable (Type) - Description**

- `app` An Application (object)
- `filename` (string) - Full path, file name, and extension of the file.

Files are typically stored in "D:\"

Choose from the following valid parameter combinations for ALL measurement classes:

<table>
<thead>
<tr>
<th>Type of file to save</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>*.prn</td>
<td>&lt;type&gt; (String)</td>
</tr>
<tr>
<td>&quot;PRN Trace Data&quot;</td>
<td>&quot;Trace&quot;</td>
</tr>
</tbody>
</table>

Example: `app.SaveData "myData.prn","PRN Trace Data","Trace","Displayed",2`
<table>
<thead>
<tr>
<th>File Extension</th>
<th>Description</th>
<th>Trace or Auto</th>
<th>RI or MA or DB</th>
<th>Displayed</th>
<th>Measurement Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>*.cti (unformatted)</td>
<td>Citifile Data Data</td>
<td>&quot;Trace&quot; or &quot;Auto&quot;</td>
<td>&quot;RI&quot;</td>
<td>&quot;RI&quot;</td>
<td>Number</td>
</tr>
<tr>
<td>Example:</td>
<td>app.SaveData &quot;myData.cti&quot;,&quot;Citifile Data Data&quot;,&quot;AUTO&quot;,&quot;RI&quot;,3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*.cti (unformatted)</td>
<td>Citifile Data Data</td>
<td>&quot;Displayed&quot;</td>
<td>&quot;RI&quot;</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>app.SaveData &quot;myData.cti&quot;,&quot;Citifile Data Data&quot;,&quot;AUTO&quot;,&quot;RI&quot;,3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*.cti (formatted)</td>
<td>Citifile Formatted Data</td>
<td>&quot;Trace&quot; or &quot;Auto&quot;</td>
<td>&quot;RI&quot; or &quot;MA&quot; or &quot;DB&quot;</td>
<td>Measurement</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>app.SaveData &quot;myData.cti&quot;,&quot;Citifile Formatted Data&quot;,&quot;AUTO&quot;,&quot;MA&quot;,3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*.cti (formatted)</td>
<td>Citifile Formatted Data</td>
<td>&quot;Displayed&quot;</td>
<td>&quot;RI&quot; or &quot;MA&quot; or &quot;DB&quot; or &quot;Displayed&quot;</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>app.SaveData &quot;myData.cti&quot;,&quot;Citifile Formatted Data&quot;,&quot;DISPLAYED&quot;,&quot;MA&quot;,-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*.csv</td>
<td>CSV Formatted Data</td>
<td>&quot;Trace&quot; or &quot;Auto&quot;</td>
<td>&quot;RI&quot; or &quot;MA&quot; or &quot;DB&quot; or &quot;Displayed&quot;</td>
<td>Measurement</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>app.SaveData &quot;myData.csv&quot;,&quot;CSV Formatted Data&quot;,&quot;Trace&quot;,&quot;DB&quot;,3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*.csv</td>
<td>CSV Formatted Data</td>
<td>&quot;Displayed&quot;</td>
<td>&quot;RI&quot; or &quot;MA&quot; or &quot;DB&quot;</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>app.SaveData &quot;myData.csv&quot;,&quot;CSV Formatted Data&quot;,&quot;displayed&quot;,&quot;RI&quot;,-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*.mdf</td>
<td>MDIF Data</td>
<td>&quot;Trace&quot; or &quot;Auto&quot;</td>
<td>&quot;RI&quot; or &quot;Displayed&quot;</td>
<td>Measurement</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>app.SaveData &quot;myData.mdf&quot;,&quot;MDIF Data&quot;,&quot;trace&quot;,&quot;displayed&quot;,1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*.mdf</td>
<td>MDIF Data</td>
<td>&quot;Displayed&quot;</td>
<td>&quot;RI&quot; or &quot;Displayed&quot;</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>app.SaveData &quot;myData.mdf&quot;,&quot;MDIF Data&quot;,&quot;displayed&quot;,&quot;displayed&quot;,-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes (for above file types)
Use `meas.Number` to read the measurement number of a trace.

**Scope:**

"Trace" - specified measurement number only.

"Channel" - all measurements that are in the channel in which the selected measurement reside are saved.

"Displayed" - all displayed measurements.

"Auto" - for all Standard Meas Class (S-parameter) channels:

- When correction is OFF, saves the specified trace
- When correction is ON, saves all corrected parameters associated with the calibrated ports in the Cal Set.

"Auto" - for all other channels:

- When correction is OFF or ON, saves the specified trace

The following parameter combinations save *.csv files in specific formats for GCA and Swept IMD classes:

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;type&gt;</td>
<td>(String)</td>
<td>&lt;scope&gt;</td>
<td>(String)</td>
</tr>
<tr>
<td>GCA channels ONLY:</td>
<td>&quot;GCA Sweep Data&quot;</td>
<td>&quot;Auto&quot;</td>
<td>&quot;DB&quot;</td>
</tr>
<tr>
<td>Swept IMD channels ONLY:</td>
<td>&quot;IMD Sweep Data&quot;</td>
<td>&quot;Auto&quot;</td>
<td>&quot;DB&quot;</td>
</tr>
</tbody>
</table>

Example: `app.SaveData "myData.csv","GCA Sweep Data","Auto","db",1`

Example: `app.SaveData "myData.csv","IMD Sweep Data","Auto","db",1`

**Return Type** Not Applicable

**Default** Not Applicable
| **C++ Syntax** | HRESULT SaveData(BSTR File, BSTR Type, BSTR Scope, BSTR Format, Long selector); |
| **Interface**  | IApplication18 |
### SaveENRFFile Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves an ENR table to disk.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>enr.SaveENRFFile(filename)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td></td>
</tr>
<tr>
<td><code>enr</code></td>
<td>An ENRFile (object)</td>
</tr>
<tr>
<td><code>filename</code></td>
<td>(String) - Absolute path and filename of the ENR file.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><a href="#">See example program</a></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SaveENRFFile(BSTR filename);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IENRFile</td>
</tr>
</tbody>
</table>

*Last Modified:* 2958
**SaveFile Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves the mixer/ converter test setup to a mixer attributes (.mxr) file.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>obj.SaveFile (filename)</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>obj</code></td>
<td>A Mixer Interface pointer to the Measurement (object)</td>
</tr>
<tr>
<td>Or</td>
<td>A Converter Object</td>
</tr>
<tr>
<td><code>filename</code></td>
<td>(String) Full path, file name, and .mxr extension of the file. Files are typically stored in &quot;D: &quot;.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mixer.SaveFile (&quot;D:\myMixer.mxr&quot;)</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT SaveFile(BSTR newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMixer</td>
</tr>
<tr>
<td></td>
<td>IConverter</td>
</tr>
</tbody>
</table>
### SaveFile Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves an external device configuration file to the VNA hard drive. Currently, only DC Supply and DC Meter configuration files are supported. See more DC Device commands.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>extDev.SaveFile (filename)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>extDev</code></td>
<td>An <code>ExternalDevice</code> Object</td>
</tr>
<tr>
<td><code>filename</code></td>
<td>(String) File name and .xml extension of the external device configuration file. Currently, only DC Supply and DC Meter configuration files are supported. See more DC Device commands.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>extDev.SaveFile &quot;MyDCSupply&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SaveFile(BSTR newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IExternalDevice</code></td>
</tr>
</tbody>
</table>
## SaveToDiskMemory Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves the User Characterization to VNA disk memory. To save to ECal internal memory, use SaveToECal Method. User Characterization can be saved to both VNA disk memory and ECal module memory.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note:</td>
<td>An ECal confidence check can NOT be performed remotely from User Characterizations that are stored on the VNA disk.</td>
</tr>
<tr>
<td>VB Syntax</td>
<td><code>userChar.SaveToDiskMemory(name)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>userChar</code></td>
<td>An ECalUserCharacterizer (object)</td>
</tr>
<tr>
<td><code>name</code></td>
<td>(String) User characterization name. Although there is no limit to the number of characters, only about 10 characters appear in the Cal Wizard dialog when selecting a user characterization for use.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>userChar.SaveToDiskMemory &quot;DUT1&quot;</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT SaveToDiskMemory(BSTR name);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IECalUserCharacterizer2</td>
</tr>
</tbody>
</table>
### SaveToECal Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves the User Characterization to the ECal module. This can take several minutes to complete. To save to VNA disk memory, use <code>SaveToDiskMemory Method</code>. User Characterization can be saved to both VNA disk memory and ECal module memory.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>userChar.SaveToECal</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>userChar</code></td>
<td>An <code>ECalUserCharacterizer</code> <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>userChar.SaveToECal</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SaveToECal();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IECalUserCharacterizer</code></td>
</tr>
</tbody>
</table>
## SaveKits Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves the cal kits, typically after modifying a calibration kit. To load a cal kit into the analyzer from the hard drive, use <code>app.RecallKits</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>app.SaveKits</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td></td>
<td><code>app</code> An <em>Application</em> <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>app.SaveKits</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT SaveKits()</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IApplication</td>
</tr>
</tbody>
</table>
## Save Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves an uncertainty ‘workspace’ (*.ml4) file from the Uncertainty Manager.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>uncertMan.Save (fileName)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>uncertMan</code></td>
<td>An <code>UncertaintyManager</code> Object</td>
</tr>
<tr>
<td><code>fileName</code></td>
<td>(String). Filename and *.ml4 extension of the Uncertainty Manager workspace file to save.</td>
</tr>
</tbody>
</table>

If `fileName` is omitted or NULL (0) is passed for this argument, then Uncertainty Manager’s current workspace is saved to whichever filename that current workspace had been recalled from, even if changes had been made to the workspace since that last Recall.

| Return Type | Not Applicable                  |
| Default     | Not Applicable                  |
| Examples    | `uncertMan.Save "MyWorkspace.ml4"` 'saves to the specified workspace filename.' |
|             | `uncertMan.Save` 'saves to the current workspace filename.'                     |

See example program

| C++ Syntax  | HRESULT Save([in, defaultvalue(0)] BSTR fileName); |
| Interface   | IUncertaintyManager |
## About Compression Markers

### SearchCompressionPoint Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Searches the markers domain for the specified compression level.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mkr.SearchCompressionPoint</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><em>mkr</em></td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mkr.SearchCompressionPoint 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT SearchCompressionPoint()</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker4</td>
</tr>
</tbody>
</table>
### SearchPowerNormalOperatingPoint Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Initiates a PNOP marker search.</th>
<th>Turns on and sets markers 1, 2, 3, and 4 to calculate various PNOP parameters.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>First set BackOff and PinOffset.</td>
<td>To turn off these markers, either turn them off individually or DeleteAllMarkers.</td>
</tr>
<tr>
<td></td>
<td>To search a UserRange, first activate marker 1 and set the desired UserRange. Then send the SearchPowerNormalOperatingPoint command. The user range applies only to marker 1 searching for the max value. The other markers may fall outside the user range.</td>
<td></td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
pnop.SearchPowerNormalOperatingPoint()
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pnop</code></td>
<td>A PNOP (object)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Not Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**Examples**

```vbnet
pnop.SearchPowerNormalOperatingPoint
```

See example program

**C++ Syntax**

```c++
HRESULT SearchPowerNormalOperatingPoint()
```

<table>
<thead>
<tr>
<th>Interface</th>
<th>IPNOP</th>
</tr>
</thead>
</table>
SearchPowerSaturation Method

Description

Initiates a Power Saturation marker search.

Turns on and sets markers 1, 2, and 3 to calculate various Power Saturation parameters.

First set PMaxBackOff.

To turn off the Power Saturation markers, either turn them off individually or use DeleteAllMarkers Method.

To search a User Range with the PSAT search, first activate marker 1 and set the desired User Range. Then send this command. The user range used with the PSAT search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

VB Syntax  
```
psat. SearchPowerSaturation()
```

Variable  

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>psat</td>
<td>A PSaturation (object)</td>
</tr>
</tbody>
</table>

Return Type  
Not Applicable

Default  
Not Applicable

Examples  
```
psat. SearchPowerSaturation
```

See example program

C++ Syntax  
```
HRESULT SearchPowerSaturation()
```

Interface  
IPSaturation
SearchFilterBandwidth Method

**Description**

Searches the measurement data with the current BandwidthTarget (default is -3). To continually track the filter bandwidth, use BandwidthTracking.

This feature uses bandwidth markers 1-4 to determine the bandwidth, center frequency, loss, and Q factor (Center Freq / Bandwidth). If not already, they are activated. To turn off these markers, either turn them off individually or DeleteAllMarkers.

The bandwidth statistics are displayed on the analyzer screen. To get the bandwidth statistics, use either GetFilterStatistics or FilterBW, FilterCF, FilterLoss, or FilterQ.

The analyzer screen will show either Bandwidth statistics OR Trace statistics; not both.

To search a UserRange with the bandwidth search, first activate marker 1 and set the desired UserRange. Then send the SearchFilterBandwidth command. The user range used with bandwidth search only applies to marker 1 searching for the max value. The other markers may fall outside the user range.

**VB Syntax**

```
meas.SearchFilterBandwidth
```

**Variable (Type) - Description**

- `meas` A Measurement (object)

**Return Type**

Not Applicable

**Default**

Not Applicable

**Examples**

```
meas.SearchFilterBandwidth
```

**C++ Syntax**

```
HRESULT SearchFilterBandwidth()
```

**Interface**

IMeasurement
# SearchMax Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Searches the marker domain for the maximum value.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mark.SearchMax</code></td>
</tr>
<tr>
<td>Variable (Type) - Description</td>
<td>A Marker (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mark.SearchMax</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT SearchMax()</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker</td>
</tr>
</tbody>
</table>
## SearchMin Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Searches the marker domain for the minimum value.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mark.SearchMin</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>mark</code></td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mark.SearchMin</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT SearchMin()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker</td>
</tr>
</tbody>
</table>
### SearchNextPeak Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Searches the marker's domain for the next peak value.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>mark.SearchNextPeak</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>mark</td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>mark.SearchNextPeak</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT SearchNextPeak()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker</td>
</tr>
</tbody>
</table>
### SearchPeakLeft Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Searches the marker's domain for the next <strong>VALID</strong> peak to the left of the marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mark.SearchPeakLeft</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>mark</code></td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mark.SearchPeakLeft</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SearchPeakLeft()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IMarker</code></td>
</tr>
</tbody>
</table>
SearchPeakRight Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Searches the marker's domain for the next <strong>VALID</strong> peak to the right of the marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mark.SearchPeakRight</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td>mark</td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mark.SearchPeakRight</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT SearchPeakRight()</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker</td>
</tr>
</tbody>
</table>

**About Marker Search**
### SearchTarget Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Searches the marker's domain for the target value (specified with <code>mark.TargetValue</code>). Searches to the right; then at the end of the search domain, begins again at the start of the search domain.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mark.SearchTarget</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>mark</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mark.SearchTarget</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT SearchTarget()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker</td>
</tr>
</tbody>
</table>
### SearchTargetLeft Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Moving to the left of the marker position, searches the marker's domain for the target value (specified with <code>mark.TargetValue</code>).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mark.SearchTargetLeft</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>mark</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mark.SearchTargetLeft</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SearchTargetLeft()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker</td>
</tr>
</tbody>
</table>
# SearchTargetRight Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Moving to the right of the marker position, searches the marker's domain for the target value (specified with <code>mark.TargetValue</code>).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mark.SearchTargetRight</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>mark</code></td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mark.SearchTargetRight</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SearchTargetRight()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker</td>
</tr>
</tbody>
</table>
SegmentCalculate Method

**Description**
Calculates the specified parameter for the segment.

**VB Syntax**
```
conv.SegmentCalculate index, param
```

**Variable** *(Type) - Description*
- `conv` A Converter Object
- `index` (Long integer) Segment for which calculation is performed. Choose a segment between 1 and the current segment count. Use `SegmentCount Property` to read the current count in the Applied Mixer.
- `param` (Enum as ConverterCalculation) Mixer port for which to calculate start and stop frequencies. Choose from:

<table>
<thead>
<tr>
<th>enum</th>
<th>1st or only stage requires:</th>
<th>In addition, 2nd stage requires:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>naCalculateINPUT</td>
<td>● Output Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● LO frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● Output sideband (High or Low)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● 2nd LO frequency</td>
</tr>
<tr>
<td>1</td>
<td>naCalculateINPUT AndOUTPUT</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● Both LO frequencies</td>
</tr>
<tr>
<td>(2 stage mixers ONLY)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>naCalculateOUTPUT</td>
<td>● Input Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● LO frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● Output sideband (High or Low)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● 2nd LO frequency</td>
</tr>
<tr>
<td>3</td>
<td>naCalculateLO1</td>
<td>● Input Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● Output frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● Output sideband (High or Low)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● 2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● IF sideband (High or Low)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
<td></td>
</tr>
<tr>
<td>Examples</td>
<td><code>mxr.SegmentCalculate 1,2</code> 'Calculates the output frequencies for segment 1.'</td>
<td></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT SegmentCalculate(long index, ConverterCalculation param);</td>
<td></td>
</tr>
<tr>
<td>Interface</td>
<td>IConverter5</td>
<td></td>
</tr>
</tbody>
</table>

- Input Start and stop frequencies
- 1st LO start and stop frequencies
- Output frequency
- IF sideband (High or Low)
- Output sideband (High or Low)
SelectCalSet Method

Description
Selects and applies a Cal Set to the specified channel.

Note: Error Correction is not automatically applied as a result of this command being issued. If there is more than one Cal Type in the Cal Set, you must explicitly choose the Cal Type you want to apply. (See meas.Caltype)

VB Syntax
channel.SelectCalSet calSet, restore

Variable
(Type) - Description

(channel) - A Channel object
(calSet) - Cal Set to make active. Specify the Cal Set by GUID or Name. Use EnumerateCalSets to list the available Cal Sets.

(restore) -
True (1) - The stimulus stored with the cal set will be applied to the channel.
False (0) - If a conflict is detected between the existing channel settings and the Cal Set stimulus settings, then the following will occur:

If interpolation is ON, then interpolation will be attempted. This may fail if the channel frequency is outside the range of the Cal Set.

If interpolation is OFF, the selection will be abandoned and an error is returned: E_NA_CAL_STIMULUS_VALUES_EXCEEDED

Return Type
Not Applicable

Default
Not Applicable

Example
channel.SelectCalSet GUID, 1
chan.SelectCalSet "MyCalSet", 0

C++ Syntax
HRESULT SelectCalSet (BSTR strCset, bool bRestore);

Interface
IChannel
SetAllSegments Method

Description
Uploads a segment table to the VNA replacing any existing segment table.

Segments must be ascending in frequency and non-overlapping. If they are not, the segments are 'adjusted' as they are from the User Interface control. The total number of points for all segments cannot exceed the VNA maximum number of points for a sweep.

VB Syntax
`Segs.SetAllSegments (segdata)`

Variable (Type) - Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>segs</td>
<td>Segments (Collection)</td>
<td></td>
</tr>
<tr>
<td>segdata</td>
<td>Variant</td>
<td>A 2-dimensional array of Segment data:</td>
</tr>
</tbody>
</table>

- dimension 0 is the number of elements in each segment.
- dimension 1 is the number of segments that will be used.

The following is a list of dimension 0 elements for each segment:

**Note:** All elements must be **dimensioned** as either ALL Double or ALL Variant.

- 0 = Segment state (Boolean True or False)
- 1 = Number of Points in this segment (Integer)
- 2 = Start Freq (Double)
- 3 = Stop Freq (Double)
- 4 = IFBW (Double) optional
- 5 = Dwell Time (Double) optional
- 6 + = Power (Double) optional; see table below.

The first four data elements must always be supplied. After those values, data must be supplied for successive optional elements. For example, to set dwell time values, you must also supply IFBW values, because IFBW (#4) precedes dwell time (#5) in the array order.

The **IFBandwidthOption**, **SweepTimeOption**, and **SourcePowerOption** settings
do NOT affect the order in which elements are interpreted.

The number of elements to supply for Power depends on the following two settings:

1. **SourcePowerOption** = True allows segments to have independent power levels.
2. **CouplePorts** = False allows different power levels for each test port.

<table>
<thead>
<tr>
<th>CouplePorts</th>
<th>SourcePowerOption</th>
<th>Number of Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>Each port has its own channel-wide power setting, which is set using <strong>TestPortPower</strong>. Provide exactly 7 elements per segment. The last element (power) is ignored.</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>Provide 6 elements + total number of ports. The first 7 elements are still interpreted the same. The remaining elements (in-order) are interpreted as the power levels to set on that segment for Ports 2 through N, where N is the total number of ports currently enabled for the VNA or for a VNA with multiport external test set.</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>Provide exactly 7 elements per segment. The last element (power) is ignored.</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>Provide exactly 7 elements per segment. The last element (power) is honored.</td>
</tr>
</tbody>
</table>

**Return Type** Not Applicable

**Default** Not Applicable

**Examples**

See a VB example using this command

See a C++ example using this command

**C++ Syntax**

```
HRESULT SetAllSegments (VARIANT Segments);
```
| Interface | ISegments2 |
### SetBBPorts Method

<table>
<thead>
<tr>
<th>Description</th>
<th>For a Balanced - Balanced device type, maps the VNA ports to the DUT ports. Set the Balanced device type using the DUTTopology Property.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>balTopology.SetBBPorts p1Pos, p1Neg, p2Pos, p2Neg</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>balTopology</strong> - A BalancedTopology (object) <strong>p1Pos, p1Neg, p2Pos, p2Neg</strong> (Long Integer) VNA port number that connects to each of the following DUT ports:</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not applicable - To read port mappings, use the BalancedTopology properties.</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>balTop.SetBBPorts 1,2,3,4</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT SetBBPorts (long p1Pos, long p1Neg, long p2Pos, long p2Neg)</td>
</tr>
<tr>
<td>Interface</td>
<td>IBalancedTopology</td>
</tr>
</tbody>
</table>

![Diagram of port connections](image)
### SetBPort Method

**Description**
For a single port balanced device type, maps the VNA ports to the DUT ports.

Set the Balanced device type using the `DUTTopology Property`

**VB Syntax**
`balTopology.SetBPort balanced_pos, balanced_neg`

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>balTopology</code></td>
<td>A <code>BalancedTopology</code> (object)</td>
</tr>
<tr>
<td><code>balanced_pos</code>, <code>balanced_neg</code></td>
<td>(Long Integer) VNA port number that connects to each of the following DUT ports:</td>
</tr>
</tbody>
</table>

![Bal Port 1](image)

**Return Type**
Not applicable - To read port mappings, use the `BalancedTopology` properties.

**Default**
Not Applicable

**Examples**
`balTopology.SetBPort 1,2`

**C++ Syntax**
`HRESULT SetBPort(long Pos, long Neg)`

**Interface**
`IBalancedTopology3`
SetBSPorts Method

**Description**
For a Balanced-Single-ended device type, maps the VNA ports to the DUT ports.

Set the Balanced-Single-ended device type using the `DUTTopology Property`.

**VB Syntax**
```
balTopology.SetBSPorts bPos, bNeg, se
```

**Variable (Type) - Description**
- `balTopology`: A `BalancedTopology` (object)
- `bPos, bNeg, se`: VNA port number that connects to each of the following DUT ports:

![Diagram of VNA ports connected to DUT ports]

**Return Type**
Not applicable - To read port mappings, use the `BalancedTopology` properties.

**Default**
Not Applicable

**Examples**
```
balTop.SetBSPorts 1,2,3
```

**C++ Syntax**
```
HRESULT SetBSPorts (long bPos, long bNeg, long se)
```

**Interface**
`IBalancedTopology2`
SetBSSPorts Method

**Description**
For a Balanced-Single-ended - Single-ended device type, maps the VNA ports to the DUT ports.

Set the Balanced-Single-ended - Single-ended device type using the DUTTopology Property

**VB Syntax**
```vbnet
balTopology.SetBSSPorts balanced_Pos, balanced_Neg, singleEnded1, singleEnded2
```

**Variable (Type) - Description**
- **balTopology**
  A `BalancedTopology` (object)
- **balanced_Pos, balanced_Neg, singleEnded1, singleEnded2**
  VNA port number that connects to each of the DUT ports:

**Return Type**
Not applicable - To read port mappings, use the `BalancedTopology` properties.

**Default**
Not Applicable

**Examples**
```vbnet
balTop.SetBSSPorts 1,2,3,4
```

**C++ Syntax**
```cpp
HRESULT SetBSPorts (long balanced_Pos, long balanced_Neg, long singleEnded1, long singleEnded2)
```

**Interface**
IBalancedTopology4
SetCalInfo Method

Description
Specifies the type of Unguided calibration. This method should be the first method called on the calibrator object. It prepares the internal state for the rest of the calibration.

Note: You can NOT perform a 3 or 4-port cal using SetCalInfo even though there is enumCalTypes. You must use the GuidedCalibration object.

Learn more about reading and writing Cal data using COM

The analyzer can measure both ports simultaneously, assuming you have two of each standard type. For a 2-port cal, See cal.Simultaneous2PortAcquisition

VB Syntax
```
cal.SetCalInfo (type,rcvPort,srcPort)
```

Variable
(Variable) - Description
```
cal       A Calibrator (object)

type      (enum NACalType) - Calibration type. Choose from:
           0  - naCalType_Response_Open
           1  - naCalType_Response_Short
           2  - naCalType_Response_Thru
           3  - naCalType_Response_Thru_And_Isol
           4  - naCalType_OnePort
           5  - naCalType_TwoPort_SOLT
           6  - naCalType_TwoPort_TRL
           7  - naCalType_None
           8  - naCalType_ThreePort_SOLT
           9  - Custom
           10 - naCalType_FourPort_SOLT
```

Note: For 1-port cals, the source port = receiver port. For 2, 3,4-port SOLT and TRL, it doesn't matter which port is specified as source and receiver

rcvPort    (long integer) - Receiver Port
srcPort    (long integer) - Source Port

Return Type NACalType

Default 7- naCalType_None
Examples

```
cal.setCalInfo(naCalType_Response_Open,1,1)
```

C++ Syntax

```
HRESULT SetCalInfo(tagNACalType calType, long portA, long portB)
```

Interface

ICalibrator
### SetCalInfoEx Method (for source power cals)

**Description**

This command replaces SetCalInfo2 Method.

Specifies the channel and the source port to be used for the source power calibration about to be performed.

**VB Syntax**

```vbnet
powerCalibrator.SetCalInfoEx channel, srcPort, [powerOffset,] [display]
```

**Variable (Type) - Description**

- `powerCalibrator` *(object)* - A `SourcePowerCalibrator` object
- `channel` *(long integer)* - Number of the VNA channel (not power meter channel) on which the source power cal will be performed. If the channel does not already exist, it will be created.
- `srcPort` *(long integer)* - Port number on which the source power cal will be performed.
- `[powerOffset]` *(double)* - Optional argument. Sets or returns a power level offset from the VNA test port power. This can be a gain or loss value (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier at the input of your DUT. Following the calibration, the VNA power readouts are adjusted by this value. This argument performs the same function as `chan.SourcePowerCalPowerOffset Property`
- `[display]` *(boolean)* Optional argument. Enables and disables the display of power readings on the VNA screen. After the source power cal data is acquired, this setting is reset to ON. If unspecified, value is set to ON.
  - **True** - Display of power readings is ON
  - **False** - Display of power readings is OFF

**Return Type**

None

**Default**

Not Applicable

**Examples**

```vbnet
powerCalibrator.SetCalInfoEx 1, 1, -10, True
```

**C++ Syntax**

```cpp
HRESULT SetCalInfoEx( long Channel, long SourcePort, double PowerOffset = 0., VARIANT_BOOL bDisplay = VARIANT_TRUE);
```
<table>
<thead>
<tr>
<th>Interface</th>
<th>ISourcePowerCalibrator4</th>
</tr>
</thead>
</table>

2990
SetCustomDUTTopology Method

**Description**
Maps the physical VNA ports to a device of balanced and single-ended logical ports for multi-port systems with greater than 4 ports.

The device type is set using the following commands:

- **SetBPort** - single balanced port DUT topology.
- **SetBBPorts** - balanced - balanced DUT topology.
- **SetBSPorts** - balanced - single-ended DUT topology.
- **SetSBPorts** - single-ended - balanced DUT topology.
- **SetSSBPorts** - single-ended - single-ended - balanced DUT topology.

The **Parameter** command returns the list of parameters available for the currently selected topology.

**VB Syntax**
```
balTopology.SetCustomDUTTopology (portTypeSequence, physicalPortsSequence)
```

**Variable**

- **balTopology** *(Type)* - Description
  A BalancedTopology *(object)*

- **portTypeSequence** *(String)* - Device type for the balanced measurement. ‘B’ means the Balanced port; ‘S’ means the Single-ended port. Choose from:
  - B – 1 port balanced device (2 ports)
  - BB – Balanced - Balanced device (4 ports)
  - BS – Balanced - Single-ended device (3 ports)
  - SB – Single-ended - Balanced device (3 ports)
  - SSB – Single-ended - Single-ended - Balanced device (4 ports)

- **physicalPortsSequence** *(Variant)* - Physical port numbers mapped to the logical ports, separated by ‘,’.
  - ‘B’ (Balanced) requires 2 physical port numbers: <nPos>, <nNeg>.
  - ‘S’ (Single-ended) requires 1 physical port number.
Return Type: Not applicable - To read port mappings, use the BalancedTopology properties.

Default: Not Applicable

Examples:
The following example sets up 8 physical ports into 6 logical ports:

Logical port 1 is a single ended port mapped to physical port 1
Logical port 2 is a single ended port mapped to physical port 2
Logical port 3 is a balanced port mapped to physical ports 3 and 4
Logical port 4 is a single ended port mapped to physical port 5
Logical port 5 is a single ended port mapped to physical port 6
Logical port 6 is a balanced port mapped to physical ports 7 and 8

Int portlist[] = {1,2,3,4,5,6,7,8}
balTopology.SetCustomDUTTopology ("SSBSSB", portlist)

C++ Syntax: HRESULT SetCustomDUTToplogy(BSTR portTypeSequence, VARIANT physicalPortsSequence)

Interface: IBalancedTopology3
### SetCenter Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Changes the center stimulus to the stimulus value of the marker. The start stimulus stays the same and the stop is adjusted.</th>
</tr>
</thead>
<tbody>
<tr>
<td>This command does not work with channels that are in CW or Segment Sweep mode.</td>
<td></td>
</tr>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mark.SetCenter</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>mark</code></td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mark.SetCenter</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SetCenter()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker</td>
</tr>
</tbody>
</table>
## SetCW Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Changes the analyzer to sweep type CW mode and sets the CW frequency to the marker's frequency. Does not change anything if current sweep type is other than a frequency sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mark.SetCW</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>mark</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mark.SetCW</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT SetCW()</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker</td>
</tr>
</tbody>
</table>
About Marker Functions

SetCWFreq Method

Description
Sets the CW frequency to the frequency of the active marker. Does NOT change sweep type.

Use ONLY when the current sweep type is sweeping frequency - NOT available in CW or Power Sweep.

Use this command to first set the CW Frequency to a value that is known to be within the current calibrated range, THEN set Sweep Type to naPowerSweep or naCWTimesSweep.

VB Syntax
mark.SetCWFreq

Variable (Type) - Description
mark A Marker (object)

Return Type
Not Applicable

Default
Not Applicable

Examples
mark.SetCWFreq

C++ Syntax
HRESULT SetCWFreq()

Interface IMarker3
## SetDutPorts Method

**Description**  
Sets the VNA to DUT port map for FCA measurements. Use DeviceInputPort and DeviceOutputPort to read these values.

Changing the ports may limit your ability to use an internal second source. If a selected port is shared by one of the sources, then that source will not be available as an LO source. Learn more about Internal second sources.

**VB Syntax**  
`mixer.SetDUTPorts (inputPort,outputPort)`

**Variable**  
**Type** - Description

- `mixer`  
  A IMixer Interface pointer to the Meas object

- `inputPort`  
  (Long) VNA port to be connected to the DUT input.
  
  - For SMC, choose any unused VNA port.
  - For VMC, set to 1

- `outputPort`  
  (Long) VNA port to be connected to the DUT output. Choose any unused port for SMC and VMC.

**Return Type**  
Not Applicable

**Default**  
1,2

**Examples**  
`mixer.SetDUTPorts =2,1`

**C++ Syntax**  
`HRESULT SetDutPorts(long inputPort,long OutputPort);`

**Interface**  
IMixer8
### SetElectricalDelay Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Changes the measurement's electrical delay to the marker's delay value.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mark.SetElectricalDelay</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>mark</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mark.SetElectricalDelay</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SetElectricalDelay()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker</td>
</tr>
</tbody>
</table>
SetFailOnOverRange Method

Description
When set TRUE, configures the analyzer to report outOfRange conditions with an error code. Any overrange error will return E_NA_LIMIT_OUTFRANGE_ERROR.

Note: This method is for the benefit of VB clients. The analyzer automatically adjusts overrange conditions to the closest acceptable setting. The VB user will not see that an overrange occurred because the HRESULT is not returned if it has a success code. For more information, see Events/OverRange.

VB Syntax
```
app.SetFailOnOverRange state
```

Variable
(through - Description

```
app An Application (object)
state (boolean) -
True (1) - Overrange conditions report an error code
False (0) - Overrange conditions report a success code
```

Return Type
Not Applicable

Default
False (0)

VB Example
```
app.SetFailOnOverRange TRUE
On Error Goto ERRHANDLER
' the following overrange will cause ERRHANDLER to be invoked
channel.StartFrequency = 9.9 GHZ
exit
```
```
ERRHANDLER:
    print "something failed"
```

C++ Syntax
```
HRESULT put_SetFailOnOverRange(VARIANT_BOOL mode)
```

Interface
IApplication
### SetIsolationPaths Method

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Adjusts the list of paths (port pairings) for which isolation standards will be measured during calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>guidedCal.SetIsolationPaths specifier, pathList</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>obj</code></td>
<td>Any of the following:</td>
</tr>
<tr>
<td></td>
<td>GuidedCalibration (object)</td>
</tr>
<tr>
<td><code>specifier</code></td>
<td>(Enum) - Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naPathsAll - Measure isolation on all pairings of the ports that are to be calibrated.</td>
</tr>
<tr>
<td></td>
<td>1 - naPathsNone - Do not measure isolation on any pairing of the ports to be calibrated.</td>
</tr>
<tr>
<td></td>
<td>2 - naPathsAdd - Add one or more specific pairings of ports to the list of port pairings for which isolation will be measured.</td>
</tr>
<tr>
<td></td>
<td>3 - naPathsRemove - Remove one or more specific pairings of ports from the list of port pairings for which isolation will be measured.</td>
</tr>
<tr>
<td><code>pathlist</code></td>
<td>(Variant) port numbers in pairs. One-dimensional array of Long Integers.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> <code>pathList</code> is evaluated only when <code>specifier</code> is <code>naPathsAdd</code> or <code>naPathsRemove</code>. For <code>naPathsAll</code> and <code>naPathsNone</code>, <code>pathList</code> is ignored.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>```vbnet</td>
</tr>
<tr>
<td></td>
<td>'selecting to measure isolation on all possible paths for the ports about to be calibrated</td>
</tr>
<tr>
<td></td>
<td>guidedCal.SetIsolationPaths naPathsAll, pathList</td>
</tr>
<tr>
<td></td>
<td>'now removing the paths 1-to-2, 2-to-3 and 2-to-4 from the set of all paths</td>
</tr>
<tr>
<td></td>
<td>pathList = Array(1,2,2,3,2,4)</td>
</tr>
<tr>
<td></td>
<td>guidedCal.SetIsolationPaths naPathsRemove, pathList</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SetIsolationPaths(enum NAPortPathSpecifier specifier, VARIANT pathList);</code></td>
</tr>
</tbody>
</table>
Interface IGuidedCalibration3
SetIPConfiguration Method

Description
Modifies settings of the VNA computer networking configuration.

VB Syntax
app.SetIPConfiguration AutoIPAddress, DNSServer1, DNSServer2, HostName, DomainName, IPAddress, SubNet, Gateway, DNSSuffix1, DNSSuffix2

or
retStr = app.SetIPConfiguration (AutoIPAddress, DNSServer1, DNSServer2, HostName, DomainName, IPAddress, SubNet, Gateway, DNSSuffix1, DNSSuffix2)

Variable (Type) - Description

app An Application (object)

AutoIPAddress (boolean) - Choose either:

True - VNA is assigned an IP address by a DHCP server, or will use AutoIP (Dynamic Link-Local Addressing) if DHCP server not found.

False – VNA will use the static IP address value specified by IPAddress.

DNSServer1 (string) IP address of primary DNS server.

When AutoIPAddress = True and an empty string is specified for DNSServer1, the VNA will attempt to obtain the addresses of primary and secondary DNS servers automatically.

When AutoIPAddress = False, an IP address must be specified for DNSServer1 and/or DNSServer2 or else the VNA’s host name will not be resolvable on the computer network.

DNSServer2 (string) IP address of secondary DNS server. When specifying an empty string for DNSServer1, then specify an empty string here also.

HostName (string) DNS host name (computer name) to be assigned to this VNA.

Note: If specifying a name different than the VNA’s current host name, the change will not take effect until after you reboot the VNA.

DomainName (string) DNS domain name associated with this VNA.

IPAddress (string) Static IP address to assign to this VNA when AutoIPAddress = False. When AutoIPAddress = True, the value of IPAddress is ignored.
**SubNet** *(string)* Subnet mask value to assign to the VNA network configuration.

**Gateway** *(string)* Gateway address to assign to the VNA network configuration.

**DNSSuffix1** *(string)* Primary suffix to set in the VNA DNS suffix search order. An empty string is allowed.

**DNSSuffix2** *(string)* Secondary suffix to set in the VNA DNS suffix search order. An empty string is allowed.

**retStr** *(string)* String returned by this method should be ignored. It is intended for Keysight diagnostic use.

**Return Type** String

**Default** Not Applicable

**Examples**

```c
app.SetIPConfiguration True, "", "", "MyHostName", "MyRegion.MyCompany.com", "", "255.255.255.0", "123.45.67.890", "", ""
```

```c
app.SetIPConfiguration False, "123.456.78.90", "234.56.78.901", "MyHostName", "MyRegion.MyCompany.com", "123.456.789.0", "255.255.255.0", "123.45.67.890", "MyCompany.com", ""
```

**C++ Syntax**

```c
HRESULT SetIPConfiguration(VARIANT_BOOL AutoIPAddress, BSTR DNSServer1, BSTR DNSServer2, BSTR HostName, BSTR DomainName, BSTR IPAddress, BSTR SubNet, BSTR Gateway, BSTR DNNSuffix1, BSTR DNNSuffix2, BSTR *pRetStr);
```

**Interface** IApplication14
SetPowerAcquisitionDevice Method

Description
Sets the power sensor channel (A or B) to be used. This performs the same function as the **Use this sensor only** checkbox in the Power Sensor Settings dialog.

**Note:** This method is only necessary when performing an SMC calibration.

VB Syntax

```
pwrCal.SetPowerAcquisitionDevice sensor
```

Variable (Type) - Description

- **pwrCal** (Object) A SourcePowerCalibrator object
- **sensor** (enum NAPowerAcquisitionDevice) The power sensor channel. Choose from:
  
  - 0 – naPowerSensor_A
  - 1 – naPowerSensor_B

Default
Not Applicable

Examples

```
pwrCal.SetPowerAcquisitionDevice naPowerSensor_A
```

C++ Syntax

```
HRESULT SetPowerAcquisitionDevice(
    tagNAPowerAcquisitionDevice enumAcqDevice);
```

Interface

ISourcePowerCalibrator3
### SetFrequencyLowPass Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Set the start frequencies when <code>trans.Mode = LowPass</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>trans.SetFrequencyLowPass</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>trans</code></td>
<td>A Transform (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>trans.SetFrequencyLowPass</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT SetFrequencyLowPass(void)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ITransform</td>
</tr>
</tbody>
</table>
SetPortMap Method

Description
Set the DUT-to-VNA port mapping for the Noise Figure, Gain Compression, IMD, IMDx, IMS, or IMSx measurement. Use the DeviceInputPort and DeviceOutputPort commands to read the DUT input and output ports.

For Noise Figure:
Port mapping is allowed without restriction when the standard PNA receiver is used (NoiseReceiver is set to naStandardReceiver). When the low-noise receiver is selected (NoiseReceiver is set to naNoiseReceiver) the following restrictions apply:

- If the low-noise receiver is selected, the DUT output port must be port 2.

On high-frequency PNAs that have an internal tuner on port 1, the input port must be port 1 if the internal tuner is selected as the noise tuner. Conversely, if the input port is something other than 1, the internal tuner cannot be selected.

For PNAs that have a maximum frequency of 26.5 GHz or less, any port can be selected as the DUT input port.

- If a vector calibration is desired, the tuner must be connected to the selected input port.

When setting IMD and IMS channels:

- When input is 1, output can be 2 or 4.
- When input is 3, output must be 4.
- This setting is necessary only when using the limited port mapping feature.

Learn more.

VB Syntax
obj.SetPortMap in,out

Variable (Type) - Description

obj
A GainCompression (object) or
A SweptIMD (object) or
An IMSpectrum (object)
A NoiseFigure (object) - See example program

in VNA port which is connected to the DUT input.
out  VNA port which is connected to the DUT output.

Return Type  Not Applicable

To read port map, use:

DeviceInputPort Property

DeviceOutputPort Property

Default  1,2

Examples  gca.SetPortMap 2,1

C++ Syntax  HRESULT SetPortMap(long input_port,long output_port);

Interface  IGainCompression

ISweptIMD

IMSpectrum

INoiseFigure6
### SetReferenceLevel Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Changes the measurement's reference level to the marker's Y-axis value.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mark.SetReferenceLevel</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>mark</td>
<td>A Marker (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mark.SetReferenceLevel</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT SetReferenceLevel()</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker</td>
</tr>
</tbody>
</table>
### SetSBPorts Method

**Description**  
For a Single-ended - Balanced device type, maps the VNA ports to the DUT ports.

Set the Single-ended - Balanced device type using the `DUTTopology` Property.

**VB Syntax**  
`balTopology.SetSBPorts se, bPos, bNeg`

**Variable**  
- `balTopology`: A `BalancedTopology` (object)
- `se, bPos, bNeg`: VNA port number that connects to each of the following DUT ports:

```
Single-end Port 1               Bal Port 2
<se>                             <bPos>
<se>                             <bNeg>
```

**Return Type**  
Not applicable - To read port mappings, use the `BalancedTopology` properties.

**Default**  
Not Applicable

**Examples**  
`balTop.SetSBPorts 1,2,3`

**C++ Syntax**  
`HRESULT SetSBPorts (long se, long bPos, long bNeg)`

**Interface**  
`IBalancedTopology`
### SetSSBPorts Method

**Description**
For a Single-ended - Single-ended - Balanced device type, maps the VNA ports to the DUT ports.

Set the Single-ended - Single-ended - Balanced device type using the DUTTopology Property.

**VB Syntax**
```vbnet
dim balTopology as BalancedTopology
balTopology.SetSSBPorts se, se2, bPos, bNeg
```

**Variable**
- **balTopology**: A BalancedTopology (object)
- **se, se2, bPos, bNeg**: VNA port number that connects to each of the following DUT ports:

![Diagram of port connections: Single-end Port 1 (<se1>, <se2>) to DUT to Bal Port 3 (<bPos>, <bNeg>), Single-end Port 2 (se2)]

**Return Type**
Not applicable - To read port mappings, use the BalancedTopology properties.

**Default**
Not Applicable

**Examples**
```vbnet
balTop.SetSSBPorts 1,2,3,4
```

**C++ Syntax**
```cpp
HRESULT SetSSBPorts (long se, long se2, long bPos, long bNeg)
```

**Interface**
IBalancedTopology
About Cal Window

SetupMeasurementsForStep Method

Description
Show the Cal Window, and optionally one or more other specific windows, before acquiring a Cal standard. This command will cause the Cal Window to display the specific measurements that are to be made for that particular Cal standard.

See custom Cal window commands.

VB Syntax
`guidedCal.SetupMeasurementsForStep (n)`

Variable (Type) - Description
`guidedCal` A GuidedCalibration (object)

`n` Step number in the calibration process.

Use `GenerateSteps` to determine the total number of steps.

Use `GetStepDescription` to read the description of each step.

Return Type
Not Applicable

Default
Not Applicable

Examples
`guidedCal.SetupMeasurementsForStep 3`

See example using this command

C++ Syntax
`HRESULT SetupMeasurementsForStep(long step);`

Interface
IGuidedCalibration4
# SetStandardsForClass Method

**Description**
Set the calibration standard numbers for a specified calibration class. To read the cal standard numbers use `GetStandardsForClass` Method.

**VB Syntax**
```vbnet
calKit.SetStandardsForClass (calclassorder, std1, std2, std3, std4, std5, std6, std7)
```

**Variable**
- **calKit** (Type) - A CalKit (object)
- **calclassorder** (enum NACalClassOrder) - Cal. Class. Choose from:
  - 0 - naRefl_1_S11
  - 1 - naRefl_2_S11
  - 2 - naRefl_3_S11
  - 3 - naTran_1_S21
  - 4 - naRefl_1_S22
  - 5 - naRefl_2_S22
  - 6 - naRefl_3_S22
  - 7 - naTran_1_S12
  - 8 - naRefl_1_S33
  - 9 - naRefl_2_S33
  - 10 - naRefl_3_S33
  - 11 - naTran_1_S32
  - 12 - naTran_1_S23
  - 13 - naTran_1_S31
  - 14 - naTran_1_S13
  - 15 - naTRL_T
  - 16 - naTRL_R
  - 17 - naTRL_L

- **std1…std7** (long) - Calibration Standard Number. Choose from 1 through 30. Std2 through Std7 are optional.

**Return Type**
Not applicable

**Default**
Not applicable
Examples

calkit.SetStandardsForClass naRefl_3_S11, 3, 5, 6

calkit.SetStandardsForClass naTran_1_S21, 4

C++ Syntax

HRESULT SetStandardsForClass(NACalClassOrder calclassorder, long std1, long std2, long std3, long std4, long std5, long std6, long std7)

Interface

ICalKit
### SetStart Method

**Description**  Changes the start stimulus to the stimulus value of the marker. The stop stimulus stays the same and the span is adjusted.

This command does not work with channels that are in **CW** or **Segment Sweep** mode.

**VB Syntax**  
```
mark.SetStart
```

**Variable (Type) - Description**

- **mark**  A Marker (object)

**Return Type**  Not Applicable

**Default**  Not Applicable

**Examples**  
```
mark.SetStart
```

**C++ Syntax**  
```
HRESULT SetStart()
```

**Interface**  IMarker
### SetStop Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Changes the stop stimulus to the stimulus value of the marker. The start stimulus stays the same and the span is adjusted.</th>
</tr>
</thead>
<tbody>
<tr>
<td>This command does not work with channels that are in CW or <strong>Segment Sweep</strong> mode.</td>
<td></td>
</tr>
</tbody>
</table>

**VB Syntax**  
`mark.SetStop`

**Variable**  
`mark`  
A Marker *(object)*

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`mark.SetStop`

**C++ Syntax**  
`HRESULT SetStop()`

**Interface**  
IMarker
### SetVISATimeout Method

**Description**
Sets the timeout value (in milliseconds) for subsequent pass-through commands for the specified VISA session.

**VB Syntax**
```vbnet
Vpassthru.SetVISATimeout visaID, timeoutVal
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vpassthru</td>
<td>A VISAPassthrough object</td>
</tr>
<tr>
<td>visaID</td>
<td>(long) VISA session number (see Open method).</td>
</tr>
<tr>
<td>timeoutVal</td>
<td>(long) Timeout value for the specified session identification number that is used when communicating with this device.</td>
</tr>
</tbody>
</table>

**Default**
2000

**Examples**
```
Vpassthru.SetVISATimeout 2,6000
```

**C++ Syntax**
```cpp
HRESULT SetVISATimeout(long session_num, long timeout)
```

**Interface**
IVISAPassthrough
ShowMarkerReadout Method

Description
Shows and Hides the Marker readout for the active marker in the upper-right corner of the window.

VB Syntax
`win.ShowMarkerReadout state`

Variable
(Type) - Description
`win` A NAWindow (object)
`state` (boolean) -
True (1) - Show the Marker readout
False (0) - Hide the Marker readout

Return Type
Not Applicable

Default
Not Applicable

Examples
`win.ShowMarkerReadout True`

C++ Syntax
`HRESULT ShowMarkerReadout(VARIANT_BOOL bState)`

Interface
INAWindow
ShowStatusBar Method

Description
Shows and Hides the Status Bar. The Status Bar is located across the bottom of the display. The following information is shown for the active measurement:

- Channel number
- Parameter
- Correction On or Off
- Remote or Local operation

VB Syntax

```vbnet
app.ShowStatusBar state
```

Variable

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><code>state</code></td>
<td>(boolean) -</td>
</tr>
<tr>
<td>True (1)</td>
<td>Show the Status Bar</td>
</tr>
<tr>
<td>False (0)</td>
<td>Hide the Status Bar</td>
</tr>
</tbody>
</table>

Return Type
Not Applicable

Default
Not Applicable

Examples

```
app.ShowStatusBar True
```

C++ Syntax

```
HRESULT ShowStatusBar (VARIANT_BOOL bState)
```

Interface
IApplication
ShowStimulus Method

**Description**
Shows and Hides the Stimulus (X-axis) information located at the bottom of the display. The start and stop stimulus values are shown for the active measurement.

**VB Syntax**
```
app.ShowStimulus state
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
<tr>
<td>state</td>
<td>(boolean) -</td>
</tr>
<tr>
<td></td>
<td>True (1) - Show the Stimulus information</td>
</tr>
<tr>
<td></td>
<td>False (0) - Hide the Stimulus information</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
app.ShowStimulus True
```

**C++ Syntax**
```
HRESULT ShowStimulus(VARIANT_BOOL bState)
```

**Interface**
IApplication
**ShowTable Method**

**Description**
Shows or Hides the specified table for the window's active measurement in the lower part of the window.

**VB Syntax**
win.ShowTable value

**Variable**
- **(Type) - Description**
  - **win** A NAWindow (object)
  - **value** (enum naTable) - The table to show or hide. Choose from:
    - 0 - naTable_None
    - 1 - naTable_Marker
    - 2 - naTable_Segment
    - 3 - naTable_Limit

**Return Type** Not Applicable

**Default** Not Applicable

**Examples**
win.ShowTable naTable_limit

**C++ Syntax**
HRESULT ShowTable (tagNATableType table)

**Interface** INAWindow
ShowTitleBars Method

Description: Shows and hides the Title Bars. The Title Bars are across the top of the Network Analyzer Window and each of the measurement windows. The Window name is shown in the Title Bar.

VB Syntax: `app.ShowTitleBars state`

Variable: 
- **app** (Type) - Description
  - An Application (object)
- **state** (boolean)
  - **True (1)** - Show the Title Bars
  - **False (0)** - Hide the Title Bars

Return Type: Not Applicable

Default: Not Applicable

Examples: `app.ShowTitleBars True`

C++ Syntax: `HRESULT ShowTitleBars(VARIANT_BOOL bState)`

Interface: IApplication
# ShowToolbar Method

**Description**
Shows and Hides the specified Toolbar.

**VB Syntax**
```
app.ShowToolbar toolbar, state
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
<tr>
<td>toolbar</td>
<td>(enum NAToolbarType) - The toolbar to show or hide. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naToolbar_None</td>
</tr>
<tr>
<td></td>
<td>1 - naToolbar_ActiveEntry</td>
</tr>
<tr>
<td></td>
<td>2 - naToolbar_Markers</td>
</tr>
<tr>
<td></td>
<td>3 - naToolbar_Measurement - OBSOLETE</td>
</tr>
<tr>
<td></td>
<td>4 - naToolbar_Stimulus - OBSOLETE</td>
</tr>
<tr>
<td></td>
<td>5 - naToolbar_SweepControl - OBSOLETE</td>
</tr>
<tr>
<td></td>
<td>6 - naToolbar_Transform</td>
</tr>
<tr>
<td></td>
<td>7 - naToolbar_PortExtensions</td>
</tr>
<tr>
<td></td>
<td>8 - naToolbar_Keys</td>
</tr>
<tr>
<td>state</td>
<td>(boolean) -</td>
</tr>
<tr>
<td></td>
<td>True (1) - Show the specified toolbar</td>
</tr>
<tr>
<td></td>
<td>False (0) - Hide the specified toolbar</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
1 - naToolbar_ActiveEntry showing; all others hiding.

**Examples**
```
app.ShowToolbar 1,1 'shows the active entry toolbar
```

**C++ Syntax**
```
HRESULT ShowToolbar(tagNAToolbarType toolbar, VARIANT_BOOL bState)
```

**Interface**
IAplication
**Single Method**

**Description**
Sets the trigger count to 1 which will cause the channel to respond once to the trigger source.

**How** the channel responds to a single trigger depends on the trigger mode (point, trace, and so forth.)

With the exception of the 'sync' argument, this command behaves like the channel 'single' setting from the user interface.

This setting has implications on Calibration. Learn more.

**VB Syntax**
```vbnet
chan.Single [sync]
```

**Variable**

- **(Type) - Description**
  - `chan` A Channel *(object)*
  - `[sync]` *(boolean)* - Optional argument.

- **True** - The VNA waits (blocks execution) until the entire acquisition process is completed.

- **False** - The VNA returns immediately - does NOT wait for acquisition to complete (non-blocking). Default setting.

When trigger source is set to Manual:

- with `sync = True`, trigger source automatically changes to Internal which sends AND allows one trigger signal, then changes back to Manual.

- with `sync = False`, a trigger signal must also be sent using `app.ManualTrigger Method`.

**Return Type** Not Applicable

**Default** Not Applicable

**Examples**
```vbnet
sync = True
chan.Single sync
```

**C++ Syntax**
```cpp
HRESULT Single(VARIANT_BOOL bWait)
```

**Interface** IChannel
### Store Method

**Description**  
Saves the path configuration currently associated with channel (`ch`) to the specified configuration name.

This command is identical to `PathConfigurationManager.StoreConfiguration` Method

**VB Syntax**  
`pathMgr.StoreConfiguration ch, name`

**Variable**  
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pathMgr</code></td>
<td><code>PathConfigurationManager (object)</code></td>
</tr>
<tr>
<td><code>ch</code></td>
<td><code>Long</code> Channel number of the configuration to be saved.</td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>String</code> Configuration name. Factory configurations can NOT be overwritten. Specifying the name of a pre-defined factory configuration will result in an error.</td>
</tr>
</tbody>
</table>

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`path.StoreConfiguration(2) "myMixer"`

**C++ Syntax**  
`HRESULT StoreConfiguration( long channelNum, BSTR configName );`

**Interface**  
`IPathConfigurationManager`
### StoreConfiguration Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves the path configuration currently associated with channel (ch) to the specified configuration name.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pathMgr.StoreConfiguration ch, name</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td>pathMgr</td>
<td>PathConfigurationManager <em>(object)</em></td>
</tr>
<tr>
<td>ch</td>
<td><em>(Long)</em> Channel number of the configuration to be saved.</td>
</tr>
<tr>
<td>name</td>
<td><em>(String)</em> Configuration name. Factory configurations can NOT be overwritten. Specifying the name of a pre-defined factory configuration will result in an error.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>path.StoreConfiguration(2) &quot;myMixer&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT StoreConfiguration( long channelNum, BSTR configName );</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPathConfigurationManager</td>
</tr>
</tbody>
</table>
### StoreTheme Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves the current color theme to a disc file.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>colors.StoreTheme(filename)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>colors</code></td>
<td>A <code>ComColors</code> (object)</td>
</tr>
<tr>
<td><code>filename</code></td>
<td>(String) - Path and filename of the theme to save.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>colors.StoreTheme = &quot;c:\Program Files(x86)\Keysight\Network Analyzer\Colors\Theme1.colors&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT StoreTheme(BSTR filename);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IColors</code></td>
</tr>
</tbody>
</table>
## StringToNACalClass Method

**Description**
Converts the returned strings from `GetStandardsList` into the enumeration (NACalClass) and the port numbers required for `PutStandard` and `GetStandard` methods that transmit data in and out of the Cal Set.

Learn more about reading and writing Cal data using COM

**VB Syntax**
```
CalSet.StringToNACalClass (list, std, rcv, src)
```

**Variable (Type) - Description**

- **CalSet** *(object)* - A Cal Set object
- **list** *(string)* - a string containing the textual description of the standard.
- **std** *(enum NACalClass)* Choose from:
  1 - naClassA
  2 - naClassB
  3 - naClassC
  4 - naClassD
  5 - naClassE
  6 - naReferenceRatioLine
  7 - naReferenceRatioThru
- **rcv** *(long)* - port number of the receiver

**SOLT Standards**
1 - naSOLT_Open
2 - naSOLT_Short
3 - naSOLT_Load
4 - naSOLT_Thru
5 - naSOLT_Isolation

**TRL Standards**
1 - naTRL_Reflection
2 - naTRL_Line_Reflection
3 - naTRL_Line_Tracking
4 - naTRL_Thru
5 - naTRL_Isolation
$src$ - (long) - port number of the source

Return Type          Not Applicable
Default             Not Applicable
Examples
        guid = CalSet.StringToNACalClass(list, std, rcv, src)

C++ Syntax
        HRESULT StringtoNACalClass (BSTR* str, NACalClass* item, long *rcv, long *src);

Interface            ICalSet
StringtoNAErrorTerm2 Method

Description
Converts the returned strings from GetErrorTermList into the enumeration (NAErrorTerm2) and the port numbers required for PutErrorTerm and GetErrorTerm methods that transmit data in and out of the Cal Set.

Learn more about reading and writing Cal data using COM

VB Syntax
CalSet.StringToNAErrorTerm2 (list, eterm, rcv, src)

Variable (Type) - Description
 Cal Set (object) - A Cal Set object
   list (string) - a string containing the textual description of the error term.
   eterm (enum As NaErrorTerm2). Choose from:
     0 - naET_Directivity          (rcv = src)
     1 - naET_SourceMatch         (rcv = src)
     2 - naET_ReflectionTracking (rcv = src)
     3 - naET_TransmissionTracking (rcv != src)
     4 - naET_LoadMatch           (rcv != src)
     5 - naET_Isolation           (rcv != src)

   rcv (long) - port number of the receiver
   src (long) - port number of the source

Return Type Not Applicable
Default Not Applicable
Examples CalSet.StringToNAErrorTerm2 str, term, rcv, src

C++ Syntax
HRESULT StringToNAErrorTerm2 (BSTR* str, NAErrorTerm2* item, long *rv, long *src);

Interface ICalSet
### SweepOnlyCalChannelDuringCalAcquisition Method

**Description**
Clears ALL flags for channels to sweep during calibration except the Cal channel. To flag a channel, see AllowChannelToSweepDuringCalAcquisition Method

**VB Syntax**
```vb
calMgr.SweepOnlyCalChannelDuringCalAcquisition
```

**Variable**

- **(Type)** - Description
- **calMgr** *(object)* - A CalManager object

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**
```vb
calMgr.SweepOnlyCalChannelDuringCalAcquisition
```

See example using this command

**C++ Syntax**
```cpp
HRESULT SweepOnlyCalChannelDuringCalAcquisition()
```

**Interface**
ICalManager5
## TestsetCatalog Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a list of supported testsets.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( \text{data} = 	ext{Tsets.TestsetCatalog} )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>( \text{data} )</td>
<td>(variant array) - Variable to store the returned data.</td>
</tr>
<tr>
<td>( \text{Tsets} )</td>
<td>(object) - An <code>ExternalTestSets</code> collection</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>( \text{value} = \text{Tsets.TestsetCatalog} )</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT TestsetCatalog(VARIANT* Data);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IExternalTestSets</code></td>
</tr>
</tbody>
</table>
# toSA Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Creates an SA channel with a marker at the same CW frequency. <a href="#">Learn more</a>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mkr.toSA</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>mkr</code></td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>See example program</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT toSA()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker6</td>
</tr>
</tbody>
</table>
### TraceHoldClear Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Resets the currently-stored data points to the live data trace and restarts the currently-selected Trace Hold type.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><em>meas.TraceHoldClear()</em></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><em>meas.TraceHoldClear</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT TraceHoldClear();</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement16</td>
</tr>
</tbody>
</table>
UserPreset Method

**Description**
Performs a User Preset. There must be an active User Preset state file (see UserPresetLoadFile and UserPresetSaveState) or an error will be returned.

Regardless of the state of the User Preset Enable checkbox, the app.Preset command will always preset the VNA to the factory preset settings, and app.UserPreset will always perform a User Preset.

**VB Syntax**
```vbnet
app.UserPreset
```

**Variable**

```
app
```
An Application (object)

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vbnet
app.UserPreset
```

**C++ Syntax**
```cpp
HRESULT UserPreset();
```

**Interface**
IApplication6
### UserPresetLoadFile Method

**Description**

Loads an existing instrument state file (.sta or .cst) to be used for User Preset. Subsequent execution of `app.UserPreset` will cause the VNA to assume this instrument state.

Regardless of the state of the User Preset Enable checkbox, the `app.Preset` command will always preset the VNA to the factory preset settings, and `app.UserPreset` will always perform a User Preset.

**VB Syntax**

```vbnet
app.UserPresetLoadFile (file)
```

**Variable**

- **(Type)** - Description
  - `app` An Application (object)
  - `file` (String) Full path, name, and extension of the file to be loaded.

**Return Type**

Not Applicable

**Default**

Not Applicable

**Examples**

```vbnet
app.UserPresetLoadFile ("C:\Program Files\Keysight\Network Analyzer\Documents\10MHzto20GHz.sta")
```

**C++ Syntax**

```cpp
HRESULT UserPresetLoadFile (BSTR bstrFile)
```

**Interface**

IApplication6
**UserPresetSaveState Method**

**Description**
Saves the current instrument settings as UserPreset.sta. Subsequent execution of `app.UserPreset` will cause the VNA to assume this instrument state.

Regardless of the state of the User Preset Enable checkbox, the `app.Preset` command will always preset the VNA to the factory preset settings, and `app.UserPreset` will always perform a User Preset.

**VB Syntax**

```vbscript
app.UserPresetSaveState
```

**Variable**

- **Type** - Description
- **app** - An Application (object)

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

`app.UserPresetSaveState`

**C++ Syntax**

```csharp
HRESULT UserPresetSaveState()
```

**Interface**
IApplication6
**Write-only**

**About VISAPassthrough**

## WriteBinary Method

**Description**
Sends binary data to the VISA pass-through device. Note that binary data transfers over sockets are not supported.

**VB Syntax**

```vbnet
Vpassthru.WriteBinary visalD, bin_data
```

**Variable**

<table>
<thead>
<tr>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Vpassthru</code> (object) - A VISAPassthrough object</td>
</tr>
<tr>
<td><code>visalD</code> (long) VISA session number (see Open method).</td>
</tr>
<tr>
<td><code>bin_data</code> (variant) Data to be sent to the VISA pass-through device.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```vbnet
Vpassthru.WriteBinary 2,"*IDN?"
```

**C++ Syntax**

```cpp
HRESULT WriteBinary(long session_num, variant bin_data)
```

**Interface**
IVISAPassthrough
**Write-only**

### About the ExtTestSetIO connector

#### WriteData Method

**Description**
Writes a 13-bit value to the specified address using the AD0 through AD12 lines of the external test set connector. The VNA generates the appropriate timing signals. It automatically controls timing signals LDS, LAS and RLW to strobe the address, then the data, to the external test set. See the timing diagram for Address and Data I/O read.

**VB Syntax**

```
ExtIO.WriteData address, value
```

**Variable**

- `ExtIO` *(object)* - An External IO object
- `address` *(variant)* - Address to be written to.
- `value` *(variant)* - 13-bit word to write

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```
ExtIO.WriteData 15,12
```

**C++ Syntax**

```c++
HRESULT WriteData(VARIANT Address, VARIANT Data);
```

**Interface**

IHWExtTestSetIO
WriteRaw Method

Description  Writes a 16-bit value to the external test set connector lines AD0 - AD12, RLW, LAS and LDS. The analyzer does NOT generate the appropriate timing signals. The user has control of all 16 lines using this write method.

Note: When RLW (pin25) is set to 1 (high) it causes lines AD0 - AD12 to float. It disables their output latches and sets the hardware for reading. LDS and LAS are not affected by this behavior.

Below is the format of data that is written with WriteRaw:

* This Output will float if RLW (bit-13) is set high

<table>
<thead>
<tr>
<th>Pin</th>
<th>Bit</th>
<th>Signal name</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>0</td>
<td>AD0*</td>
</tr>
<tr>
<td>23</td>
<td>1</td>
<td>AD1*</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>AD2*</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>AD3*</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>AD4*</td>
</tr>
<tr>
<td>21</td>
<td>5</td>
<td>AD5*</td>
</tr>
<tr>
<td>20</td>
<td>6</td>
<td>AD6*</td>
</tr>
<tr>
<td>19</td>
<td>7</td>
<td>AD7*</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>AD8*</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>AD9*</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>AD10*</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>AD11*</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>AD12*</td>
</tr>
<tr>
<td>25</td>
<td>13</td>
<td>RLW</td>
</tr>
<tr>
<td>24</td>
<td>14</td>
<td>LDS</td>
</tr>
<tr>
<td>8</td>
<td>15</td>
<td>LAS</td>
</tr>
</tbody>
</table>
VB Syntax  

ExtIO.WriteRaw value

Variable  (Type) - Description

ExtIO  (object) - An External IO object

value  (variant) - Data to be written

Return Type  Not Applicable

Default  Not Applicable

Examples  ExtIO.WriteRaw 12

C++ Syntax  

HRESULT WriteRaw(VARIANT Output);

Interface  

IHWExternalTestSetIO
WriteSnPData Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves the S-parameters and vector noise parameters to an S2P file. For NFX channels, mixer setup information is included as comments at the beginning of the file.</th>
</tr>
</thead>
<tbody>
<tr>
<td>The following is sample data for two data points in a Noise Figure channel:</td>
<td></td>
</tr>
<tr>
<td>! Keysight Technologies,N5242A,USxxxxxxx,A.09.85</td>
<td></td>
</tr>
<tr>
<td>! pnan-nn Thu Nov 01 12:26:27 2012</td>
<td></td>
</tr>
<tr>
<td># HZ S MA R 50</td>
<td></td>
</tr>
<tr>
<td>!freq (Hz)  S11M   S11A   S21M   S21A   S12M   S12A   S22M   S22A</td>
<td></td>
</tr>
<tr>
<td>2000000000  9.038147e-001  6.241193e+001  5.855965e+000  -6.116778e+001  2.232653e-002  -1.475392e+002  5.275644e-001  1.750775e+002</td>
<td></td>
</tr>
<tr>
<td>8000000000  6.951366e-001  -1.458202e+002  5.307699e+000  -7.055212e+001  7.838612e-002  -1.460951e+002  3.986142e-001  -2.226317e+001</td>
<td></td>
</tr>
<tr>
<td>! Noise Parameters</td>
<td></td>
</tr>
<tr>
<td>!freq (Hz)  NFMin(dB)  Rho_opt(Mag)  Rho_opt(deg)  Rn/Z0</td>
<td></td>
</tr>
<tr>
<td>2000000000  1.251697e+000  2.172018e-001  -8.765875e+001  1.806663e-001</td>
<td></td>
</tr>
<tr>
<td>8000000000  1.583849e+000  2.015185e-001  -1.029875e+001  1.320403e-001</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VB Syntax</th>
<th><code>nf.WriteSnPData(data, filename)</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>nf</code></td>
<td>A NoiseFigure (object)</td>
</tr>
<tr>
<td><code>data</code></td>
<td>(string) - Choose &quot;NoiseParameter&quot; - Noise parameter data</td>
</tr>
<tr>
<td><code>filename</code></td>
<td>Path, filename and suffix of location to store SNP data.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
Examples  

```plaintext
nf.WriteSnPData "NoiseParameter", "C:\Program Files(x86)\Keysight\Network Analyzer\Documents\MyNoiseParams.s2p"
```

**C++ Syntax**

```plaintext
HRESULT WriteSnPData(BSTR data, BSTR filename);
```

**Interface**

```plaintext
INoiseFigure7
```
WriteSnpFileWithSpecifiedPorts Method

Description  Note: This command replaces app.Save (.snp). This command is more explicit regarding the data to be saved, and works for VNAs with multiport test sets.

Saves SNP data to the specified file. Learn more about SNP data.

VB Syntax  

```vbnet
data = meas.WriteSnpFileWithSpecifiedPorts ports, filename
```

Variable (Type) - Description

- `data` (Variant) array to store the data.
- `meas` A Measurement (object)
- `ports` (Variant Array) One dimensional array containing a list of port numbers for which snp data is requested.
- `filename` (string) - Full path, filename, and suffix to store the data.

The suffix is not checked for accuracy. If saving 2 ports, specify "filename.s2p"; If saving 3 ports, specify "filename.s3p." and so forth.

SNP data can be output using several data formatting options. See SNPFormat Property

Return Type  Variant array - automatically dimensioned to the size of the data.

Default  Not Applicable

Examples

' This VBScript example can be pasted into a notepad file and run on the VNA as a macro. Learn how.

```vbnet
Set pna = CreateObject("AgilentPnA835x.application")
Set meas = pna.ActiveMeasurement

' List the port numbers for required data
ports = Array(1,2,4)

' specify where to save the data
filename="C:\Program Files(x86)\Keysight\Network Analyzer\Documents\MyData.s3p"
meas.WriteSnpFileWithSpecifiedPorts ports, filename
```

C++ Syntax  HRESULT WriteSnpFileWithSpecifiedPorts(VARIANT portsToMeasure,BSTR filename);
# WriteString Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Sends ASCII string data to the VISA pass-through device.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>Vpassthru.WriteString visaID, text</code></td>
</tr>
<tr>
<td><strong>Variable</strong> (Type) - Description</td>
<td></td>
</tr>
<tr>
<td><code>Vpassthru</code> (object) - A VISAPassthrough object</td>
<td></td>
</tr>
<tr>
<td><code>visaID</code> (long) VISA session number (see Open method).</td>
<td></td>
</tr>
<tr>
<td><code>text</code> (string) Text to be sent to the VISA pass-through device (usually a command).</td>
<td></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>Vpassthru.WriteString 2,&quot;*IDN?&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT WriteString(long session_num, string text)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IVISAPassthrough</td>
</tr>
</tbody>
</table>
### WriteUncertaintyFile Method

**Description**
Saves uncertainty data for the specified ports in three different formats.

**VB Syntax**
\[
data = \text{ uncert.} \text{WriteUncertaintyFile} \text{ ports, filename}
\]

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>(Variant) array to store the data.</td>
</tr>
<tr>
<td>uncert</td>
<td>An Uncertainty (object)</td>
</tr>
<tr>
<td>ports</td>
<td>(Variant Array) One dimensional array containing a list of port numbers for which data is requested.</td>
</tr>
<tr>
<td>filename</td>
<td>(string) - Path, filename, and suffix of location to store the uncertainty data, enclosed in quotes. The suffix is not checked for accuracy.</td>
</tr>
</tbody>
</table>

- **(*.u*p) S-parameter Uncertainty File** - If saving 2 ports, specify "filename.u2p"; If saving 4 ports, specify "filename.u4p.", and so forth.

- **(*.dsd) S-parameter Data Standard Definition file** - Data for ONLY one or two ports is allowed.

- **(*.sdatcv) METAS S-parameter Covariance File.**

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
This VBScript example can be pasted into a notepad file and run on the VNA as a macro. Learn how.

```vbnet
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim oUncert as Uncertainty
Set oUncert = app.ActiveMeasurement.Uncertainty

'List the port numbers for required data
ports = Array(1,2)

'specify where to save the data and data suffix
'remove comment for one of the following:
filename="D:\MyData.u2p"
```
```c++
HRESULT WriteUncertaintyFile(VARIANT portsToMeasure, BSTR filename);
```

**Interface**

IUncertainty
### ZeroTermsInS4PFile Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Creates a new S4P file.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>calmgr.ZeroTermsInS4PFile (original, modified, sParams, format)</code></td>
</tr>
</tbody>
</table>

#### Variable

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>calMgr</code></td>
<td>Cal Manager (Object)</td>
</tr>
<tr>
<td><code>original</code></td>
<td>Path and filename of the original S4P file.</td>
</tr>
<tr>
<td><code>modified</code></td>
<td>Path and filename of the new S4P file.</td>
</tr>
<tr>
<td><code>sParams</code></td>
<td>Comma-separated terms to zero-out.</td>
</tr>
<tr>
<td><code>format</code></td>
<td>(Enum as NAPairedDataFormat) Format in which the data is to be saved to the S4P file. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <code>naLogMagPhase</code></td>
</tr>
<tr>
<td></td>
<td>1 - <code>naLinMagPhase</code></td>
</tr>
<tr>
<td></td>
<td>2 - <code>naRealImaginary</code></td>
</tr>
</tbody>
</table>

| **Return Type** | Not Applicable |
| **Default** | Not Applicable |

**Examples**

```vbnet
CalManager.ZeroTermsInS4PFile "D:\testDiffMatch.s4p","D:\testDiffMatch_com.s4p","S11,S21,S33",0
```

**C++ Syntax**

```c++
HRESULT ZeroTermsInS4PFile(BSTR original, BSTR modified, BSTR sParams, tagNAPairedDataFormat format);
```

**Interface**

ICalManager13
OnCalEvent

Description
Triggered by a calibration event. See a list of CAL Events.

**Note:** Some Severe Events are also used as Error Messages

VB Syntax
Sub app_OnCalEvent(ByVal eventID As Variant, ByVal chanNum As Variant, ByVal measNum As Variant)

Variable (Type) - Description
- `app` An Application (object)
- `eventID` Code number of the event which occurred
- `chanNum` Channel Number of the event
- `measNum` Measurement Number of the event

Return Type
Not Applicable

Default
Not Applicable

Examples
Sub pna_OnCalEvent(ByVal eventID As Variant, ByVal channelNumber As Variant, ByVal measurementNumber As Variant)'
    MsgBox ("A Calibration event has occurred")
End Sub

C++ Syntax
HRESULT OnCalEvent(VARIANT eventID, VARIANT channelNumber, VARIANT measurementNumber)

Interface
IApplication

Selected Cal Events

512 naEventID_CAL_QUESTIONABLE
513 naEventID_CAL_STD_NEEDED
514 naEventID_CAL_STATE_NOT_HW_COMPATIBLE
515 naEventID_CAL_REQUIRED
516 naEventID_CAL_CORRECTION_TURNED_OFF
517 naEventID_CAL_CORRECTION_TURNED_OFF_INTERPOLATION_OFF
518 naEventID_CAL_CORRECTION_RESTORED
519 naEventID_CAL_CORRECTION_TURNED_OFF_FREQANGE_EXCEEDED
520 naEventID_CAL_CALTYPE_SET_TO_NONE
521 naEventID_CAL_CORRECTION_TURNED_OFF_NOT_AN_SPARAM
524 naEventID_SOURCE_POWER_CAL_COMPLETED
592 naEventID_SOURCE_POWER_CAL_NOT_PRESENT
593 naEventID_SOURCE_POWER_CAL_NOT_COMPLETE
594 naEventID_SOURCE_POWER_CAL_REMOVE_TRACE
595 naEventID_SOURCE_POWER_CAL_REMOVE_MEAS
596 naEventID_SOURCE_POWER_CAL_POWER_CHANGED
598 naEventID_INSUFFICIENT_SLIDE_MOVEMENT
613 naEventID_CALSET_NOT_FOUND
615 naEventID_CALSET_CREATED
617 naEventID_CALSET_FILE_NOT_VALID
634 naEventID_CALSET_LOAD_FAILED
635 naEventID_CALSET_SAVE_FAILED
636 naEventID_CALSET_DELETED
637 naEventID_CALSET_FILE_NOT_COMPATIBLE
639 naEventID_NEW_CALSET_FILE_CREATED
640 naEventID_CAL_SET_IN_USE
644 naEventID_CAL_COULD_NOT_TURN_ON
693 naEventID_ERROR_FIXTURING_S2FILE_CANNOT_OPEN
696 naEventID_ERROR_FIXTURING_TURNED_OFF
701 naEventID_MORE_THRU_PATHS_NEEDED

See Also
Errors and the SCPIStringParser Object
### OnChannelEvent

<table>
<thead>
<tr>
<th>Description</th>
<th>Triggered by a channel event.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>Sub app_OnChannelEvent(ByVal eventID As Variant, ByVal chanNum As Variant)</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><code>eventID</code></td>
<td>Code number of the event which occurred</td>
</tr>
<tr>
<td><code>chanNum</code></td>
<td>Channel Number of the event</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>Sub pna_OnChannelEvent(ByVal eventID As Variant, ByVal channelNumber As Variant)</code></td>
</tr>
<tr>
<td></td>
<td><code>If eventID=naEventID_CHANNEL_CREATED then</code></td>
</tr>
<tr>
<td></td>
<td><code>MsgBox &quot;Channel&quot; + channelNumber + &quot; was created&quot;</code></td>
</tr>
<tr>
<td></td>
<td><code>End If</code></td>
</tr>
<tr>
<td></td>
<td><code>End Sub</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT OnChannelEvent(VARIANT eventID, VARIANT channelNumber)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>

### Selected Channel Events

- 1792 `naEventID_CHANNEL_SWEEP_COMPLETE`
- 1793 `naEventID_CHANNEL_TRIGGER_COMPLETE`
- 1796 `naEventID_SET_CHANNEL_DIRTY`
- 1797 `naEventID_CLEAR_CHANNEL_DIRTY`
- 1801 `naEventID_ALL_SWEEPS_COMPLETED_AND_PROCESSED`
- 1805 `naEventID_CHANNEL_CREATED`
- 1806 `naEventID_CHANNEL_DELETED`
- 1876 `naEventID_NO_SOURCE_ATTEN`
- 1879 `naEventID_FREQ_OFFSET_OVERRANGE_SO_TURNED_OFF`
1883 naEventID_PORT_NUMBER_OUT_OF_RANGE

See Also

Errors and the SCPIStringParser Object
### About Analyzer Events

#### OnDisplayEvent

<table>
<thead>
<tr>
<th>Description</th>
<th>Triggered by a display event.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>Sub <code>app_OnDisplayEvent(ByVal eventID As Variant, ByVal winNum As Variant, ByVal traceNum As Variant)</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><code>eventID</code></td>
<td>Code number of the event which occurred</td>
</tr>
<tr>
<td><code>winNum</code></td>
<td>Window Number of the event</td>
</tr>
<tr>
<td><code>traceNum</code></td>
<td>Trace Number of the event</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**Examples**

```vbnet
Sub pna_OnDisplayEvent(ByVal eventID As Variant, ByVal windowNumber As Variant, ByVal traceNumber As Variant)
    MsgBox ("A Display event has occurred")
End Sub
```

**C++ Syntax**

```csharp
HRESULT OnDisplayEvent(VARIANT eventID, VARIANT windowNumber, VARIANT traceNumber)
```

**Interface**

IApplication

### Selected Display Events

- **1541** naEventID_PRINT_SETUP_FAILURE
- **1542** naEventID_PRINT_CANCELED

### See Also

Errors and the SCPIStringParser Object
About Analyzer Events

OnHardwareEvent

Description
Triggered by a hardware event. See a list of Hardware Events

Note: Some Severe Events are also used as Error Messages

VB Syntax
Sub app_OnHardwareEvent(ByVal eventID As Variant)

Variable (Type) - Description
app An Application (object)
eventID Code number of the event which occurred

Return Type Not Applicable
Default Not Applicable

Examples
Private Sub pna_OnHardwareEvent(ByVal eventID As Variant)
    MsgBox ("A Hardware event has occured")
End Sub

C++ Syntax
HRESULT OnHardwareEvent(VARIANT eventID)

Interface IApplication

Selected Hardware Events

848 naEventID_PHASELOCK
852 naEventID_RFPOWEROFF
853 naEventID_RFPOWERON
855 naEventID_UNLEVELED
857 naEventID_OVERLOAD
914 naEventID_TRIGGER_REQUIRES_EDGE_LEVEL_TRIGGER
915 naEventID_TRIGGER_REQUIRES_TRIGGER_OUT

See Also
Errors and the SCPIStringParser Object
OnMeasurementEvent

Description
Triggered by a measurement event.

VB Syntax
Sub OnMeasurementEvent(ByVal eventID As Variant, ByVal measNum As Variant)

Variable
(app) An Application (object)
(eventID) Code number of the event which occurred
(measNum) Measurement Number of the event

Return Type
Not Applicable

Default
Not Applicable

Examples
Private Sub pna_OnMeasurementEvent(ByVal eventID As Variant, ByVal measurementNumber As Variant)
    MsgBox("A Measurement event has occurred")
End Sub

C++ Syntax
HRESULT OnMeasurementEvent(VARIANT eventID, VARIANT measurementNumber)

Interface
IApplication

Selected Measurement Events

1024 naEventID_NO_VALID_MEMORY_TRACE
1028 naEventID_LIMIT_FAILED
1029 naEventID_LIMIT_PASSED
1034 naEventID_MEMORY_NOT_SAVED
1035 naEventID_SET_AVERAGE_COMPLETE
1036 naEventID_CLEAR_AVERAGE_COMPLETE
1111 naEventID_MARKER_BANDWIDTH_NOT_FOUND

1112 naEventID_PEAK_NOT_FOUND

1113 naEventID_TARGET_VALUE_NOT_FOUND

See Also

Errors and the SCPIStringParser Object
## OnSCPIEvent

<table>
<thead>
<tr>
<th>Description</th>
<th>Triggered by a SCPI event.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>Some Severe Events are also used as Error Messages</td>
</tr>
</tbody>
</table>

### VB Syntax

```vbnet
Sub app_OnSCPIEvent(ByVal eventID As Variant)
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><code>eventID</code></td>
<td>Code number of the event which occurred</td>
</tr>
</tbody>
</table>

### Return Type

Not Applicable

### Default

Not Applicable

### Examples

```vbnet
Private Sub pna_OnSCPIEvent(ByVal eventID As Variant)
    MsgBox ("A SCPI event has occurred")
End Sub
```

### C++ Syntax

```cpp
HRESULT OnSCPIEvent(VARIANT eventID )
```

### Interface

IApplication

### Selected SCPI Parser Events

- 1281 `naEventID_NOTHING_TO SAY`
- 1284 `naEventID_SCPI_STATUS_BYTE_CHANGE`
- 1360 `naEventID_BAD_SCPI_EXECUTE`
- 1375 `naEventID_CALC_MEASUREMENT_SET_TO_NONE`

### See Also

Errors and the SCPIStringParser Object
**OnSystemEvent**

**Description**
Triggered by a system event. See a list of System Events, also known as general events.

See also EnableSourceUnleveledEvents Property

**Note:** Some Severe Events are also used as Error Messages

**VB Syntax**
Sub *app_OnSystemEvent*(ByVal *eventID* As Variant)

**Variable**

- *app* (Type) - Description
  An Application (object)
- *eventID* Code number of the event which occurred
- *chanNum* Channel Number of the event

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```vbnet
Private Sub pna_OnSystemEvent(ByVal eventID As Variant)
    MsgBox ("A System event has occurred")
End Sub
```

**C++ Syntax**

```
HRESULT OnSystemEvent(VARIANT *eventID)
```

**Interface**
IApplication

**Selected System Events**

- 2048 naEventID_OPTION_NOT_INSTALLED
- 2049 naEventID_FEATURE_NOT_AVAILABLE
- 2050 naEventID_FEATURE_NOT_VALID
- 2051 naEventID_SAVEFILE_OK
- 2063 naEventID_RECALLFILE_SUCCESS
- 2130 naEventID_PRINTER_TROUBLE
2133 naEventID_TRIGGERDENIED
2134 naEventID_MACRO_FAILED
2144 naEventID_NO_LICENSE
2163 naEventID_PRESET
2166 naEventID_TRIGGERFAILED

See Also

Errors and the SCPIStringParser Object
### OnUserEvent

<table>
<thead>
<tr>
<th>Description</th>
<th>Reserved for future use.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>Sub app <strong>OnUserEvent</strong></td>
</tr>
</tbody>
</table>
Active (Hot) Parameters

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as <filename>.vbs. Learn how to setup and run the macro.

See Also

ActiveParametersApp Object

See other COM Examples

'Demonstration of Active Parameters setup.
Dim app
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset
app.CreateCustomMeasurementEx 2, "Active Parameters", "IPwr"
Set ActParam = app.ActiveChannel.CustomChannelConfiguration
ActParam.SweepType = 3 'naHOTS22MultiSweep
ActParam.StartPowerIn3DSweep = -15 'Start power -15dBm
ActParam.StopPowerIn3DSweep = 0 'Stop power 0dBm
ActParam.PowerStepsIn3DSweep = 4 '4 steps with -15,-10,-5,0dBm
ActParam.DisplayInputPower(2) = 0 ' Set the display as the 0dBm input
The following example uses the smart pointer created by Microsoft Visual Studio. The calls to CoInitialize and CoUninitialize open and close the COM libraries.

Also notice that the pointers local to the main routine are explicitly released. When smart pointers go out of scope, they will perform this duty implicitly. However, we are calling CoUninitialize before they have the chance to be destroyed, so we are obliged to release them.

```c++
// An example program to illustrate the use of #import to bind to the
// PNA type library.

#ifdef _UNICODE
# define _UNICODE
#endif

#include "stdafx.h"
#include "stdio.h"
#include "math.h"

/////////////////////////////////////////////
// import the network analyzer type library
/////////////////////////////////////////////
#import "C:\Program Files\Common Files\Keysight\Pna\835x.tlb" no_namespace,
   named_guids
/////////////////////////////////////////////
// include the error definitions for the PNA so we can implement
// error handling.
/////////////////////////////////////////////
#include "C:\Program Files\Common Files\Keysight\Pna\errorsystemmessage.h"

IAplicationPtr pNA; // top level application pointer
float fScalarData [1601]; // global buffer for data retrieval
float fScalarData2[1601];
DWORD dwCookie;

/////////////////////////////////////////////
// SetupChannel:
/////////////////////////////////////////////
void SetupChannel(IChannelPtr pChannel) {
}
pChannel->put_StartFrequency( 1.2E9 );
pChannel->put_StopFrequency ( 4.2E9 );
pChannel->put_NumberOfPoints ( 201);
}

////////////////////////////////////////////////////////////////////
// AcquireData:
//
// input: pointer to the channel
//
// function: single sweeps the channel
////////////////////////////////////////////////////////////////////
void AcquireData( IChannelPtr pChannel )
{
    pChannel->Single( TRUE );
}

////////////////////////////////////////////////////////////////////
// ReadData:
//
// input: pointer to the Measurement object
//
// function: reads data from the measurement's formatted
// result data buffer
////////////////////////////////////////////////////////////////////
void ReadScalarData(IMeasurementPtr pMeas )
{
    IArrayTransferPtr pDataTransfer;
    pDataTransfer = pMeas;
    long numVals = 1601;
    float* pData = fScalarData;

    if(pDataTransfer){
        pDataTransfer->getScalar( naMeasResult, naDataFormat_LogMag, &numVals, pData);

        for ( int i = 0; i < numVals; i++)
            printf("%d/t%f/n",i,pData[i]);
    }
    TCHAR msg[100];
    BSTR param;
    pMeas->get_Parameter(&param);
    swprintf(msg,L"Review %s data",param);
    MessageBox(NULL,msg,L"User Message",0);
    ::SysFreeString(param);
}

void ReadComplexData(IMeasurementPtr pMeas )
{
    IArrayTransferPtr pDataTransfer;
pDataTransfer = pMeas;
long numVals = 1601;
float* pReal = fScalarData;
float* pImag = fScalarData2;

if(pDataTransfer){

pDataTransfer->getPairedData( naRawData, naRealImaginary, &numVals, pReal, pImag);

for (int i = 0; i < numVals; i++)
printf("%d/t%f/t%f/n",i,pReal[i], pImag[i]);
}
TCHAR msg[100];
BSTR param;
pMeas->get_Parameter(&param);
swprintf(msg,L"Review %s data",param);
MessageBox(NULL,msg,L"User Message",0);
::SysFreeString(param);
}

/**************************************************************************/
// PutData:
//
// input: pointer to the Measurement object
//
// function: writes data to the measurement's raw data buffer
/***************************************************************************/
void PutData( IMeasurementPtr pMeas )
{
    IArrayTransferPtr pDataTransfer;
pDataTransfer = pMeas;
    long numVals = 201;

    if(pDataTransfer){
NAComplex* pComplex = new NAComplex[numVals];
pComplex[0].Im = 0;
pComplex[0].Re = 1;
for (int i = 1; i < numVals; i++)
{
pComplex[i].Im = (float)sin(i)/i;
pComplex[i].Re = (float)cos(i)/i;
}
pDataTransfer->putNAComplex( naRawData, numVals, pComplex, naDataFormat_Polar);
delete [] pComplex;
}

/**************************************************************************/
// printError
/**************************************************************************/
void printError( HRESULT hr)
{
    BSTR text;

    hr = pNA->get_MessageText ((NAEventID) hr, &text);
    MessageBox(NULL,text,L"Network Analyzer error",0);
    ::SysFreeString(text);
}

int main(int argc, char* argv[])
{
    HRESULT hr;
    const long channel1 = 1;
    const long window1 = 1;
    const long srcport = 1;
    IMeasurementPtr pMeasurement;
    IChannelPtr pChannel;

    // initialize COM libraries
    CoInitialize(NULL);

    try {
        pNA = IApplicationPtr("AgilentPNA835x.Application.1");

        pNA->put_Visible(TRUE);
        pNA->Reset();

        pNA->CreateMeasurement (channel1, "S21",srcport, 3);
        hr = pNA->get_ActiveChannel( &pChannel);

        if (SUCCEEDED (hr))
        {
            SetupChannel( pChannel);
            AcquireData(pChannel);
        }

        hr= pNA->get_ActiveMeasurement( &pMeasurement);
        if (SUCCEEDED(hr))
        {
            pMeasurement->put_Format( naDataFormat_Polar);
            ReadScalarData( pMeasurement);
            ReadComplexData( pMeasurement);
            PutData(pMeasurement);
        }
        if (FAILED(hr))
        {
            // error handling
        }
    }
    catch(...)
    {
        // exception handling
    }
}
printError(hr);
}

// make sure to release the remaining pointers
// before calling CoUninitialize

pMeasurement.Release();
pChannel.Release();
pNA.Release();
}
catch (_com_error err)
{
printError( err.Error() );
}

CoUninitialize();
return 0;
}
The following program creates an Independent Calibration Channels which is a comma-separated channel list. Any selected channels will perform their own SmartCal instead of importing the calibration from the SmartCal performed on the special Cal All channel (typically Channel 200).

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as BalancedCOM.vbs. Learn how to setup and run the macro.

```vbs
Dim host: host = "localhost"
Dim pna: set pna = createobject("Agilentpna835x.application",host)
WScript.Echo pna.idstring
PNA.Preset
Set scpi = pna.ScpiStringParser
Dim mgr
Set mgr = pna.GetCalManager
Dim CalAll
Set CalAll = mgr.CalibrateAllChannels
CalAll.Reset
'Perform calibration on Channel 2 SMC:
CalAll.Channels = Array(1)
CalAll.PropertyValue("Include Power Calibration") = "true"
CalAllPropertyValue("Independent Calibration Channels") = "1"
CalAll.PowerLevel(1) = -5
Dim guidedcal
Set guidedcal = calAll.GuidedCalibration
' Specify the DUT connectors
Guidedcal.ConnectorType(1) = "APC 3.5 male"
Guidedcal.ConnectorType(2) = "APC 3.5 female"
Guidedcal.ConnectorType(3) = "Not used"
```
guidedCal.ConnectorType(4) = "Not used"
guidedCal.CalKitType(1) = "85052D"
guidedCal.CalKitType(2) = "85052D"
Numsteps = guidedCal.GenerateSteps
wscript.echo "Numsteps: " + cstr(Numsteps)
For i = 1 to Numsteps
    step = "Step " + CStr(i) + " of " + CStr(Numsteps)
    strPrompt = guidedCal.GetStepDescription(i)
    value = MsgBox(strPrompt, vbOKOnly, step)
    guidedCal.AcquireStep i
Next
guidedCal.GenerateErrorTerms
Cal All Multi-Channel Independent Calibration Channels

The following program creates an Independent Calibration Channels calibration on multiple measurement class channels. Any selected channels will perform their own SmartCal instead of importing the calibration from the SmartCal performed on the special Cal All channel (typically Channel 200).

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as BalancedCOM.vbs. Learn how to setup and run the macro.

```vbs
    dim host: host = "localhost"
    dim pna: set pna = createobject("Agilentpna835x.application",host)
    wscript.echo pna.idstring
    pna.preset
    pna.CreateCustomMeasurementEx 2, "Gain Compression Converters", "CompS11", 1
    pna.CreateCustomMeasurementEx 3, "Scalar Mixer/Converter", "SC12", 1
    pna.CreateCustomMeasurementEx 4, "Spectrum Analyzer", "b1", 1
    pna.CreateCustomMeasurementEx 5, "Standard", "S22", 1
    Set scpi = pna.ScpiStringParser
    Dim mgr
    Set mgr = pna.GetCalManager
    Dim CalAll
    Set CalAll = mgr.CalibrateAllChannels
    CalAll.Reset
    'Perform calibration on Channel 2 SMC:
    CalAll.Channels = Array(1,2,3,4,5)
    CalAll.PropertyValue("Include Power Calibration") = "true"
    CalAll.PropertyValue("Independent Calibration Channels") = "1,3"
    CalAll.PowerLevel(1) = -5
    Dim guidedcal
    set guidedcal = calAll.GuidedCalibration
```
' Specify the DUT connectors

guidedcal.ConnectorType(1) = "APC 3.5 male"
guidedcal.ConnectorType(2) = "APC 3.5 female"
guidedcal.ConnectorType(3) = "Not used"
guidedcal.ConnectorType(4) = "Not used"
guidedCal.CalKitType(1) = "85052D"
guidedCal.CalKitType(2) = "85052D"

Numsteps = guidedcal.GenerateSteps

wscript.echo "Numsteps: " + cstr(Numsteps)

' Note that this calibration is performing multiple calibrations

' But you will only be prompted once for each connection type

For i = 1 to Numsteps
    step = "Step " + CStr(i) + " of " + CStr(Numsteps)
    strPrompt = guidedCal.GetStepDescription(i)
    value = MsgBox(strPrompt, vbOKOnly, step)
    guidedCal.AcquireStep i
Next

guidedCal.GenerateErrorTerms
The following program creates a Split Cal which applies to SMC and GCX channels only. If selected, these channels will perform their own calibrations, performing two 1-port calibrations (no thru).

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as BalancedCOM.vbs. Learn how to setup and run the macro.

```vbs
Dim host: host = "localhost"
Dim pna: set pna = createobject("Agilentpna835x.application", host)
wscript.echo pna.idstring
pna.preset
pna.CreateCustomMeasurementEx 2, "Scalar Mixer/Converter", "SC12", 1
Set scpi = pna.ScpiStringParser
Dim mgr
Set mgr = pna.GetCalManager
Dim CalAll
Set CalAll = mgr.CalibrateAllChannels
CalAll.Reset
'Perform calibration on Channel 2 SMC:
CalAll.Channels = Array(2)
CalAll.PropertyValue("Include Power Calibration") = "true"
CalAll.PropertyValue("Split Cal") = "true"
CalAll.PowerLevel(1) = -5
Dim guidedcal
set guidedcal = calAll.GuidedCalibration
' Specify the DUT connectors
guidedcal.ConnectorType(1) = "APC 3.5 male"
guidedcal.ConnectorType(2) = "APC 3.5 female"
guidedcal.ConnectorType(3) = "Not used"
```
guidedcal.ConnectorType(4) = "Not used"
guidedCal.CalKitType(1) = "85052D"
guidedCal.CalKitType(2) = "85052D"
Numsteps = guidedcal.GenerateSteps
wscript.echo "Numsteps: " + cstr(Numsteps)

'Note that this performs two one-port calibrations

'So the power sensor will be prompted to connect on P1 and P2

For i = 1 to Numsteps
    step = "Step " + CStr(i) + " of " + CStr(Numsteps)
    strPrompt = guidedCal.GetStepDescription(i)
    value = MsgBox(strPrompt, vbOKOnly, step)
    guidedCal.AcquireStep i
Next

guidedCal.GenerateErrorTerms
This example iterates over the entire collection of Cal Sets that currently reside in the VNA. It reads the entire list of error term strings from each Cal Set and queries the data for each term. It then does the same for the standards data.

Learn more about Reading and Writing Calibration data using COM.

Learn more about Cal Sets.

See example: Writing Cal Set Data using COM

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as CalSets.vbs. Learn how to setup and run the macro.

```vbs
Dim pna
Dim cset
Dim calsets

' create the pna object
' to run on a remote PC, substitute 'name' for the full computer name of your VNA
' to run as a macro on the VNA, remove "name"
Set pna = CreateObject("AgilentPNA835x.Application", "name")

wscript.echo pna.IDString

' obtain the calset collection
Set calsets = pna.GetCalManager.calsets

' loop thru the calsets
Dim c
For c = 1 To calsets.count
    Set cset = calsets.Item(c)
    wscript.echo "calset = ", cset.GetGUID, cset.Description

    ' iterate through error terms data
    Dim vterms,fdata
    vterms = cset.GetErrorTermList2(0, "")
    if (Not IsEmpty(vterms)) then
        For i = LBound(vterms) To UBound(vterms)
            wscript.echo vterms(i)
            vdata = cset.GetErrorTermByString(0,vterms(i))
            fdata = cset.GetErrorTermStimulus(0,vterms(i))
        Next
    End If
Next
```
wscript.echo vdata(1,0), vdata(1,1)
wscript.echo fdata(1,0), fdata(1,1)
Next
end if

' iterate through standards data
vterms = cset.GetStandardList2("")
if (Not IsEmpty(vterms)) then
For i = LBound(vterms) To UBound(vterms)
wscript.echo vterms(i)
vdata = cset.GetStandardByString( vterms(i) )
wscript.echo vdata(1,0), vdata(1,1)
Next
end if
Next
Getting Trace Data from the Analyzer

This Visual Basic program:

- Retrieves Scalar Data from the Analyzer and plots it.
- Retrieves Paired Data from the Analyzer and plots it.
- Retrieves Complex Data from the Analyzer and plots it.

To use this code, prepare a form with the following:

- Two MSCharts named **MSChart1** and **MSChart2**
- Three buttons named **GetScalar**, **GetPaired**, **GetComplex**

**Note:** You can get MSChart in Visual Basic by clicking Project / Components / Microsoft Chart Control

```
'Put this in a module
Public dlocation As NADataStore
Public numpts As Long
Public fmt As NADataFormat
Public app As Application
Public measData As IArrayTransfer
Public chan As Channel

Sub Form_Load()
    'Change analyzerName to your analyzer's full computer name
    Set app = CreateObject("AgilentPNA835x.Application", "analyzerName")

    Set measData = app.ActiveMeasurement
    Set chan = app.ActiveChannel

    'To pick a location to get the data from remove the comment from one of these
dlocation = naRawData
    'dlocation = naCorrectedData
    'dlocation = naMeasResult
    'dlocation = naRawMemory
    'dlocation = naMemoryResult

    'setup MSChart1 and MSChart2
    'right click on the chart and select:
    '- line chart
    '- series in rows
```
Sub GetComplex_Click()
ReDim Data(numpts) As NAComplex
Dim Real(201) As Single
Dim Imag(201) As Single
numpts = chan.NumberOfPoints

' You cannot change the format of Complex Data
Call trigger
' get data
measData.GetNAComplex dlocation, numpts, Data(0)
' plot data
Dim i As Integer
For i = 0 To numpts - 1
    Real(i) = Data(i).Re
    Imag(i) = Data(i).Im
Next i
MSChart1 = Real()
MSChart2.Visible = True
MSChart2 = Imag()
Call Sweep
End Sub

Sub GetPaired_Click()
ReDim Real(numpts) As Single
ReDim Imag(numpts) As Single
numpts = chan.NumberOfPoints

' To pick a format, remove the comment from one of these
fmt = naLogMagPhase
'fmt = naLinMagPhase
Call trigger
' Get data
measData.getPairedData dlocation, fmt, numpts, Real(0), Imag(0)
' Plot Scalar
MSChart1 = Real()
MSChart2.Visible = True
MSChart2 = Imag()
Call Sweep
End Sub

Sub GetScalar_Click()
ReDim Data(numpts) As Single
numpts = chan.NumberOfPoints
'To pick a format remove the comment from one of these
fmt = naDataFormat_LogMag
'fmt = naDataFormat_LinMag
'fmt = naDataFormat_Phase
'fmt = naDataFormat_Delay
'fmt = naDataFormat_Real
'fmt = naDataFormat_Imaginary
Call trigger
'Get data
measData.GetScalar dlocation, fmt, numpts, Data(0)
'Plot Data
MSChart1 = Data()
MSChart2.Visible = False
Call Sweep
End Sub

Sub trigger()
'The analyzer sends continuous trigger signals
app.TriggerSignal = naTriggerInternal
'The channel will only accept one, then go into hold
'Sync true will wait for the sweep to complete
sync=True
chan.Single sync
End Sub

Sub Sweep()
'The channel goes back to accepting all triggers
chan.Continuous
End Sub
Perform a Guided Calibration using COM

This example uses the GuidedCalibration interface to perform either a 2-port or 4-port calibration.

Learn more about Reading and Writing Calibration data using COM.

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as Calibrate.vbs. Learn how to setup and run the macro.

```vbnet
Set pna = CreateObject("AgilentPNA835x.Application")
Set calMgr = pna.GetCalManager
Set guidedCal = calMgr.GuidedCalibration
Set chan = pna.ActiveChannel
chanNum = chan.ChannelNumber
' Initialize guided cal to be performed on the active channel.
' The boolean argument of True indicates to store the cal only
' in the channel's calibration register. If instead you wish
' to create a new calset that the new cal will get stored to,
' comment out this next line and uncomment the three lines below it.
guidedCal.Initialize chanNum, True
'Set calset = calMgr.CreateCalSet(chanNum)
'chan.SelectCalSet calset.GetGUID, True
'guidedCal.Initialize chanNum, False

' To perform 2-port cal, Uncomment the following
' Then comment the 4-port cal

' Do 2-port cal
'TwoPortGuidedCal

' Do 4-port cal
FourPortGuidedCal

Sub TwoPortGuidedCal()
' Select the connectors
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 male"
For i = 3 To pna.NumberOfPorts
  guidedCal.ConnectorType(i) = "Not used"
Next
value = MsgBox("Connectors defined for Ports 1 and 2")
' Select the Cal Kit for each port being calibrated.
guidedCal.CalKitType(1) = "85052D"
```
`guidedCal.CalKitType(2) = "85052D"

'To use an ECal module instead, comment out the above two lines
' and uncomment the appropriate lines below:
' Your ECal module must already be connected
' via USB to the VNA.

'guidedCal.CalKitType(1) = "N4691-60004 ECal"
'guidedCal.CalKitType(2) = "N4691-60004 ECal"
'Non-factory characterizations are specified as follows:
'guidedCal.CalKitType(1) = "N4691-60004 User 1 ECal"
'When two or more ECal modules with the same model number are connected
'also specify the serial number as follows:
'guidedCal.CalKitType(1) = "N4691-60004 ECal 01234"
'When Disk Memory ECal user characterizations are used,
'specify both the User char and the serial number as follows:
'guidedCal.CalKitType(1) = "N4691-60004 MyDskChar ECal 01234"

MsgBox("Cal kits defined for Ports 1 and 2")

'Initiate the calibration and query the number of steps
numSteps = guidedCal.GenerateSteps
'Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

Sub FourPortGuidedCal()
'Select the connectors
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 female"
guidedCal.ConnectorType(3) = "APC 3.5 female"
guidedCal.ConnectorType(4) = "APC 3.5 female"
'If a VNA which has more than 4 ports
For i = 5 To pna.NumberOfPorts
  guidedCal.ConnectorType(i) = "Not used"
Next
value = MsgBox("Connectors defined for Ports 1 to 4")
'Select the Cal Kit for each port being calibrated.
' To use an ECal module instead, comment out the above four lines
' and uncomment the following four lines.
' Replace N4691-60003 with your own ECAL model followed by 'ECal'.
' Your ECal module must already be connected to a VNA USB port.
' See above for ECal options

'guidedCal.CalKitType(1) = "N4431-60003 ECal"
guidedCal.CalKitType(2) = "N4431-60003 ECal"
guidedCal.CalKitType(3) = "N4431-60003 ECal"
guidedCal.CalKitType(4) = "N4431-60003 ECal"

value = MsgBox("Cal kits defined for Ports 1 to 4")

' Initiate the calibration

guidedCal.GenerateSteps

' If your selected cal kit is not a 4-port ECal module which can
' mate to all 4 ports at once, then you may want to choose which
' thru connections to measure for the cal. You must measure at
' least 3 different thru paths for a 4-port cal (for greatest
' accuracy you can choose to measure a thru connection for all 6
' pairings of the 4 ports). If you omit this command, the default
' is to measure from port 1 to port 2, port 1 to port 3, and
' port 1 to port 4. For this example we select to measure
' from port 1 to port 2, port 2 to port 3, and port 2 to port 4.
portList = Array(1,2,2,3,2,4)
guidedCal.ThruPortList = portList

' Re-generate the connection steps to account for the thru changes
numSteps = guidedCal.GenerateSteps

' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)

End Sub

Sub MeasureAndComplete(ByVal numSteps)
value = MsgBox("Number of steps is " + CStr(numSteps))

' Measure the standards
'The following series of commands shows that standards
'can be measured in any order. These steps acquire
'measurement of standards in reverse order.
'It is easiest to iterate through standards using
'a For-Next Loop.
For i = NumSteps To 1
step = "Step " + CStr(i) + " of " + CStr(numSteps)
strPrompt = guidedCal.GetStepDescription(i)
value = MsgBox(strPrompt, vbOKOnly, step)
guidedCal.AcquireStep i
Next

' Conclude the calibration
.guidedCal.GenerateErrorTerms
MsgBox ("Cal is done!"
End Sub
Perform a Source Power Cal using COM

This program can be run in either Visual Basic 6 or as a VBScript program. The VNA can run *.vbs programs as macros.

This program demonstrates:

- Performing a source power calibration of Port 2 for Channel 1.
- Reading the calibration data.

Learn more about Power Calibrations

See an example that Uploads a Source Power Cal

See Other COM Example Programs

To run this program, you need:

- One of the following power meters connected to the VNA through GPIB: E4416A, E4417A, E4418A/B, E4419A/B, 437B, 438A, EPM-441A, EPM-442A

**Note:** If your power meter is other than these, you can create your own Power Meter Driver using our template.

- Your PC and VNA both connected to a LAN (for communicating with each other).

To make this program work in VBS, save the following code in a text editor file such as Notepad and save as *.vbs.

To make this program work in Visual Basic 6:

1. Create a new project
2. Click **Project, Add New Module**, click **Open**.
3. Paste the following code into the code window.
4. Delete the first two lines (comment and Main)
5. Click **Project**, **Properties**. Under **Startup Object**, select **Sub Main**

6. Click **Project**, **References**, and select the Keysight PNA Series Type Library.

```vba
' Run the Main subroutine
Main
Public Sub Main()
    Dim PNA, chan, pwrcal ' VNA COM objects
    Const naPowerMeter = 0, naPowerMeterAndReceiver = 1 ' enum NASourcePowerCalMethod
    Const naPowerSensor_A = 0 ' enum NAPowerAcquisitionDevice
    Const naCorrectionValues = 0 ' enum NASourcePowerCalBuffer
    Const port = 2 ' VNA port #2 as source port
    Const offset = 0 ' cal power offset value
    Const bDisplay = True ' whether to display data during acquire
    Dim stimulus, calvalues, strResult

    ' Instantiate our VNA COM objects
    Set PNA = CreateObject("AgilentPNA835x.Application")
    Set chan = PNA.Channels(1)
    Set pwrcal = PNA.SourcePowerCalibrator

    ' Set the number of sweep points to 21 on Channel 1.
    chan.NumberOfPoints = 21

    ' Specify the GPIB address of the power meter
    ' that will be used in performing the calibration.
    pwrcal.PowerMeterGPIBAddress = 13

    ' Turn use of the loss table OFF (this assumes there is
    ' virtually no loss in the RF path to the power sensor
    ' due to a splitter, coupler or adapter).
    pwrcal.UsePowerLossSegments = False

    ' Turn frequency checking OFF (so one power sensor is used for the entire cal
    ' acquisition sweep regardless of frequency span).
    pwrcal.UsePowerSensorFrequencyLimits = False

    ' Specify a nominal power accuracy tolerance (IterationsTolerance) in dB for the
    ' calibration, and the maximum number of iterations to adjust power at each point,
    ' attempting to achieve within tolerance of the desired power. If at any stimulus
    ' point the power fails to reach within the set tolerance of the desired power
    ' after the maximum number of iterations, the power at that point will be set to the
    ' value determined by the last iteration (the Source Power Cal dialog box will
    ' indicate the FAIL, but we can still apply the cal if desired when it's complete).
    ' Each iteration is based upon a SETTLED power reading (see comments preceding the
    ' next two properties below).
    pwrcal.IterationsTolerance = 0.1
    pwrcal.MaximumIterationsPerPoint = 3

    ' The worst-case window of power uncertainty (for a calibration which meets
    ' tolerance) is the sum of the iteration tolerance and the power meter settling
```
At each stimulus point, the VNA takes power meter readings and determines when they have settled by comparing the magnitude difference between consecutive readings versus a nominal dB tolerance limit (ReadingsTolerance) on that magnitude difference. When consecutive readings are within tolerance of each other, or if they are not within tolerance but we’ve taken a maximum number of readings (ReadingsPerPoint), the VNA does a weighted average of the readings taken at that stimulus point and that is considered our settled power reading.

```
pwrcal.ReadingsTolerance = 0.1
pwrcal.ReadingsPerPoint = 5
```

Setup of information pertaining to this specific cal acquisition. Includes the method (type of devices) that will be used to perform the cal -- choose either naPowerMeter or naPowerMeterAndReceiver. naPowerMeterAndReceiver uses the power meter for the first iteration of each point and the VNA’s reference receiver for subsequent iterations, so is much faster than using power meter only naPowerMeter).

But the power meter accounts for compression when calibrating at the output of an active device, whereas the reference receiver cannot unless it is coupled to the cal reference plane (on a VNA which allows direct access to the receivers).

'offset' specifies if the cal power level is offset (positive value for a gain, negative value for a loss) from the VNA port power setting on the channel when no source power cal is active. This is to account for components between the VNA test port and cal reference plane. In this example, we will calibrate at the VNA test port, so there is no offset (it is zero).

'bDisplay' indicates whether to display the source power cal dialog during the source power cal acquisition (the dialog will chart the corrected power readings).

```
pwrcal.SetCalInfo2 naPowerMeter, chan.channelNumber, port, offset, bDisplay
```

Perform synchronous source power cal acquisition sweep using the sensor attached to Channel A of the power meter. This assumes that the power sensor is already connected to Port 2 of the VNA.

```
pwrcal.AcquirePowerReadings naPowerSensor_A, True
```

Conclude the calibration. This applies the cal data to VNA channel memory, and turns the correction ON for Port 2 on Channel 1, but does NOT save the calibration.

```
pwrcal.ApplyPowerCorrectionValues
```

At this point, if you choose to save the instrument state as a ".CST" file, the calibration will be saved with the instrument state in that file.

Read the stimulus values from Channel 1.

```
stimulus = chan.GetXAxisValues
```

Read the source power correction data.

```
calvalues = chan.getSourcePowerCalDataEx(naCorrectionValues, port)
```

Print the data using a message box (here, Chr returns the ASCII characters for Tab (9) and Linefeed (10)).

```
strResult = "Stimulus" & Chr(9) & Chr(9) & "Cal Value" & Chr(10)
For i = 0 To UBound(stimulus)
strResult = strResult & stimulus(i) & Chr(9) & calvalues(i) & Chr(10)
```

3087
Next
MsgBox strResult
End Sub
Independent Power Calibration

The following program creates an independent power calibration over a specified frequency span when performing a Cal All.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as BalancedCOM.vbs. Learn how to setup and run the macro.

```vbs
Dim host: host = "K-N5222B-10034"
dim pna: set pna = createobject("Agilentpna835x.application", host)
wscript.echo pna.idstring
pna.preset
pna.CreateCustomMeasurementEx 2, "Gain Compression", "S21", 1

Dim mgr
set mgr = pna.GetCalManager
Dim CalAll

Set CalAll = mgr.CalibrateAllChannels
CalAll.Reset
CalAll.Channels = Array(1, 2)
CalAll.PropertyValue("Include Power Calibration") = "true"
CalAll.PropertyValue("Enable Extra Power Cals") = "Port 2,Port 3"

'Add power calibrations on port 3:
CalAll.IndependentPowerCalibration(3).AddPowerCalRange
CalAll.IndependentPowerCalibration(3).PowerCalRange(1).StartFrequency = 3e9
CalAll.IndependentPowerCalibration(3).PowerCalRange(1).StopFrequency = 4e9
CalAll.IndependentPowerCalibration(3).PowerCalRange(1).NumberOfPoints = 21
CalAll.IndependentPowerCalibration(3).AddPowerCalRange
CalAll.IndependentPowerCalibration(3).PowerCalRange(2).StartFrequency = 20e9
CalAll.IndependentPowerCalibration(3).PowerCalRange(2).StopFrequency = 21e9
CalAll.IndependentPowerCalibration(3).PowerCalRange(2).NumberOfPoints = 7
CalAll.IndependentPowerCalibration(3).AddPowerCalRange
CalAll.IndependentPowerCalibration(2).PowerCalRange(1).StartFrequency = 3e9
CalAll.IndependentPowerCalibration(2).PowerCalRange(1).StopFrequency = 4e9
CalAll.IndependentPowerCalibration(2).PowerCalRange(1).NumberOfPoints = 21

Dim calPorts
calPorts = CalAll.IndependentPowerCalibration(3).ValidPorts
PrintVector calPorts, "Cal Ports"

src3rangecount = CalAll.IndependentPowerCalibration(3).RangeCount
wscript.echo "src3rangecount: " + cstr(src3rangecount)
src2rangecount = CalAll.IndependentPowerCalibration(2).RangeCount
```

3089
wscript.echo "src2rangecount: " + cstr(src2rangecount)

CalAll.IndependentPowerCalibration(2).Reset
src2rangecount = CalAll.IndependentPowerCalibration(2).RangeCount
wscript.echo "src2rangecount: " + cstr(src2rangecount)

start = CalAll.IndependentPowerCalibration(3).PowerCalRange(1).StartFrequency
stopf = CalAll.IndependentPowerCalibration(3).PowerCalRange(1).StopFrequency
points = CalAll.IndependentPowerCalibration(3).PowerCalRange(1).NumberOfPoints
wscript.echo "start,stop,points: " + cstr(start) +"," + cstr(stopf) + "," + cstr(points)

CalAll.PowerLevel(1) = -5

Dim guidedcal
set guidedcal = calAll.GuidedCalibration
' Specify the DUT connectors
guidedcal.ConnectorType(1) = "APC 3.5 male"
guidedcal.ConnectorType(2) = "APC 3.5 female"
guidedcal.ConnectorType(3) = "Not used"
guidedcal.ConnectorType(4) = "Not used"
guidedCal.CalKitType(1) = "N4691-61004 ECal 13442"
guidedCal.CalKitType(2) = "N4691-61004 ECal 13442"

Numsteps = guidedcal.GenerateSteps
wscript.echo "Numsteps: " + cstr(Numsteps)

For i = 1 to Numsteps
  step = "Step " + CStr(i) + " of " + CStr(Numsteps)
  strPrompt = guidedCal.GetStepDescription(i)
  value = MsgBox(strPrompt, vbOKOnly, step)
  guidedCal.AcquireStep i
Next

guidedCal.GenerateErrorTerms

sub PrintVector (vec, msg)
dim i, str
if (IsArray(vec)) then
  for i = lbound(vec) to ubound(vec)
    str = str + cstr(vec(i)) + " 
  next
else
  str = "vector is EMPTY"
end if
wscript.echo msg +": " + str
end sub
Perform a Guided Calibration using C++

This example uses the GuidedCalibration interface to perform a 4 port Guided Calibration using 4 Thru paths and the Unknown Thru Algorithm.

This example was tested using a N5242A with A.09.42.10 firmware.

Learn more about Reading and Writing Calibration data using COM.

See Other COM Example Programs

```c
#include <afxdisp.h>
#include <stdio.h>
#include "atlbase.h"
#include "objbase.h"

// import the VNA type library

#include "C:\Program Files(x86)\Keysight\Network Analyzer\Automation\835x.tlb"
no_namespace, named_guids

int _tmain(int argc, _TCHAR* argv[]) {

    // interface pointers to retrieve COM interfaces
    IUnknown* pUnk = 0;
    IApplication* pNA = 0;
    IApplication9* pNA9 = 0;
    IChannel* pChan = 0;
    IMeasurement* pMeas = 0;
    IArrayTransfer* pTrans = 0;
    ITriggerSetup* pTrig = 0;
    ICalManager* pCalMgr = 0;

```
ICalManager3* pCalMgr3 = 0;
IGuidedCalibration* pGuidedCal = 0;
IGuidedCalibration6* pGuidedCal6 = 0;
VARIANT thruList;
COleSafeArray sa;

long thurs[8] = {1,2,1,4,2,3,3,4};
long i, num_points = 0, numSteps = 0;
float* pScalarData;

HRESULT hr;

// Initialize the COM subsystem
CoInitialize(NULL);
CoInitializeSecurity(NULL, //security descriptor
-1, // authn svc entries
NULL, // authn svcs
NULL, // reserved
RPC_C_AUTHN_LEVEL_NONE,
RPC_C_IMP_LEVEL_IMPERSONATE,
NULL, // authn info
0, // capabilities
NULL); // reserved

// Create an instance of the network analyzer
// Request the NA's IUnknown interface
hr = CoCreateInstance(CLSID_Application, 0, CLSCTX_ALL, IID_IUnknown, (void**) &pUnk);

if (!FAILED(hr))
{

// QueryInterface for the INetworkAnalyzer interface of the NetworkAnalyzer object
hr = pUnk->QueryInterface(IID_IApplication,(void**)&pNA);
if (!FAILED(hr))
{
    // Reset the analyzer to instrument preset
    pNA->Reset();

    // Create S11 measurement
    pNA->CreateSParameter(1,1,1,1);
    // Set pChan variable to point to the active channel
    pNA->get_ActiveChannel(&pChan);
    if (pChan)
    {
        // Set pMeas variable to point to the active measurement
        pNA->get_ActiveMeasurement(&pMeas);

        // Make the VNA application visible
        pNA->put_Visible(true);

        // Set channel parameters
        pChan->NumberOfPoints = 11;
        pChan->put_StartFrequency(3.0e9);
        pChan->put_FrequencySpan(4.0e9);
        pChan->put_IFBandwidth(1.0e3);
        pChan->put_SweepGenerationMode(naSteppedSweep);

        // Get CalManager3 Access and GuidedCalibration access
        hr=pNA->raw_GetCalManager(&pCalMgr);
        hr=pCalMgr->QueryInterface(IID_ICalManager3, (void**)&pCalMgr3);
pCalMgr3->getGuidedCalibration((IDispatch**)&pGuidedCal);
hr=pGuidedCal->QueryInterface(IID_IGuidedCalibration6, (void**)&pGuidedCal6);

// Initialize the Guided Calibration Process
pGuidedCal6->Initialize(1, VARIANT_TRUE);

// Set Port Connectors
pGuidedCal6->put_ConnectorType(1, (_bstr_t) "APC 3.5 female");
pGuidedCal6->put_ConnectorType(2, (_bstr_t) "APC 3.5 female");
pGuidedCal6->put_ConnectorType(3, (_bstr_t) "APC 3.5 female");
pGuidedCal6->put_ConnectorType(4, (_bstr_t) "APC 3.5 female");

// Set Cal Kit
pGuidedCal6->put_CalKitType(1, (_bstr_t) "85052D");
pGuidedCal6->put_CalKitType(2, (_bstr_t) "85052D");
pGuidedCal6->put_CalKitType(3, (_bstr_t) "85052D");
pGuidedCal6->put_CalKitType(4, (_bstr_t) "85052D");

// Convert thrus to VARIANT
sa.CreateOneDim(VT_I4, sizeof(thrus)/sizeof(long), thruList, 0);

// Set Thru Port List
pGuidedCal6->put_ThruPortList(thruList);

// Set Thru Path Calibration Types
pGuidedCal6->put_PathCalMethod(1, 2, (_bstr_t) "SOLT");
pGuidedCal6->put_PathCalMethod(1, 4, (_bstr_t) "SOLT");
pGuidedCal6->put_PathCalMethod(2, 3, (_bstr_t) "SOLT");
pGuidedCal6->put_PathCalMethod(3, 4, (_bstr_t) "SOLT");

// Set Thru Cal Method
pGuidedCal6->put_PathThruMethod(1, 2, (_bstr_t) "Undefined Thru");
pGuidedCal6->put_PathThruMethod(1, 4, (_bstr_t) "Undefined Thru");
pGuidedCal6->put_PathThruMethod(2, 3, (_bstr_t) "Undefined Thru");
pGuidedCal6->put_PathThruMethod(3, 4, (_bstr_t) "Undefined Thru");
// Initialize after modifying the SmartCal logic
pGuidedCal6->Initialize(1, VARIANT_TRUE);

// Generate Calibration Steps
numSteps=pGuidedCal6->GenerateSteps();
printf("Number of Calibration Steps = %ld/n/n", numSteps);

// Collect calibration steps
for(i=0; i<numSteps; i++)
{
    printf("Step %d >>> %s /n ",i+1, (char *)pGuidedCal6->GetStepDescription(i+1));
    system("PAUSE");
    pGuidedCal6->AcquireStep(i+1);
}

// Complete Calibration by Generating Error Terms
pGuidedCal6->GenerateErrorTerms();

if(pMeas)
{
    // Get handle to IApplication9
    hr = pNA->QueryInterface(IID_IApplication9, (void**)&pNA9);
    // Get handle to ITriggerSetup
    pNA9->get_TriggerSetup(&pTrig);

    // Setup the channel for a single trigger
    pChan->Hold(true);
    pTrig->Source = naTriggerSourceInternal;
    pNA->TriggerSignal = naTriggerManual;
    pChan->TriggerMode = naTriggerModeMeasurement;
// Send a manual trigger to initiate a single sweep

pChan->Single(true);

// QueryInterface for the IArrayTransfer interface of the NetworkAnalyzer object
hr = pMeas->QueryInterface(IID_IArrayTransfer,(void**)&pTrans);

if (!FAILED(hr))
{
    // Store the data in the "result" variable
    num_points = pChan->NumberOfPoints;
    pScalarData = new float[num_points];

    //Get Measurement Values
    pTrans->getScalar(naCorrectedData, naDataFormat_LogMag, &num_points, pScalarData);
        
        // Display the result
        printf("/nSingle Sweep S11 Data:/n");
        for (i = 0; i < num_points; i++)
        {
            //Write value...
            printf("%f/n", pScalarData[i]);
        }
    
}

sa.Destroy();
delete [] pScalarData;
pUnk->Release();
pMeas->Release();
pChan->Release();
    pTrans->Release();
pCalMgr->Release();
pCalMgr3->Release();
pGuidedCal->Release();
pGuidedCal6->Release();
pTrig->Release();
pNA9->Release();
    pNA->Release();
}
else
{
    printf("Programmed failed to connect to the PNA.");
}
}
CoUninitialize();

system("PAUSE");
return 0;
Perform an Unguided Cal using COM

This example uses the ICalibrator interface to do the following:

- perform a two port calibration
- retrieve the error term data
- retrieve the standard data (cal acquisition data)

Learn more about Reading and Writing Calibration data using COM.

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as Calibrate.vbs. Learn how to setup and run the macro.

dim pna
' To run from an external PC, substitute your VNA Name and use the following command.
' set pna = CreateObject("AgilentPNA835x.Application", "VNA Name")
set pna = CreateObject("AgilentPNA835x.Application")
dim calibrator
set calibrator=pna.activechannel.calibrator

wscript.echo "setcalinfo for two port cal"
calibrator.setcalinfo 5,1, 2

' only have one set of standards
calibrator.Simultaneous2PortAcquisition = false

'first acquire forward reflection standards, then reverse
dim p
for p = 1 to 2

if (p = 1) then
  calibrator.AcquisitionDirection = 0
else
  calibrator.AcquisitionDirection = 1
end if

wscript.echo "connect open to port ", p
calibrator.acquirecalstandard 1

calibrator.acquirecalstandard 2

calibrator.acquirecalstandard 3

next

calibrator.acquirecalstandard 4

' Optional - perform isolation

calibrator.acquirecalstandard 5

wscript.echo "calculating"
calibrator.CalculateErrorCoefficients

' Calibration complete

' Now read error terms and standard data

dim termName

termName= Array("Directivity","SourceMatch","ReflectionTracking")
dim vardata

' iterate over error terms

dim t
for t = 0 to 2 ' per error term
for p = 1 to 2 ' per port
wscript.echo "Requesting ",termName(t),p,p
vardata = calibrator.GetErrorTerm( t, p, p)
next
next

' now get the path terms: iterate each one request

termName = Array("Isolation", "LoadMatch", "TransmissionTracking")
for t = 0 to 2

wscript.echo "Requesting Forward term",termName(t),1,2
vardata = calibrator.GetErrorTerm( t, 1,2)
wscript.echo "Requesting Reverse Term",termName(t),2,1
vardata = calibrator.GetErrorTerm( t, 2,1)
next

dim stdname

stdname= Array("","Open","Short","Load","Thru","Isolation")

' iterate over the port standards

for t = 1 to 3
for p = 1 to 2
' request the standard term for each port of interest
wscript.echo "Requesting", stdname(t), p, p
vardata = calibrator.GetStandard( t, p, p)
next
next

' now get the path standards: iterator each one request
for t = 4 to 5
wscript.echo "Requesting Forward", stdname(t), 1, 2
vardata = calibrator.GetStandard( t, 1, 2)
wscript.echo "Requesting Reverse", stdname(t), 2, 1
vardata = calibrator.GetStandard( t, 2, 1)
next
Perform an Unknown Thru or TRL Cal

The following program performs either a 2-port SOLT Unknown Thru Cal or a 2-port TRL Cal. The 85052C Cal Kit used in this program contains both types of standards. This program can be run on 2-port or 4-port VNAs. When run on select PNA-L models, a Delta Match Cal is required. See Delta Match Cal example program.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Unknown.vbs. Learn how to setup and run the macro.

```
Sub PerformUnknownThruOrTRLCal()
    ' Create / Get the VNA application.
    Set app = CreateObject("AgilentPNA835x.Application")
    ' Get the cal manager object
    Set calMgr = app.GetCalManager
    ' Get the guided cal object
    Set guidedCal = calMgr.GuidedCalibration
    Set chan = app.ActiveChannel
    chanNum = chan.ChannelNumber

    ' Initialize guided cal to be performed on the active channel.
    ' The boolean argument of True specifies the creation of a new calset
    ' for storing the new calibration.
    guidedCal.Initialize chanNum, True

    ' Specify connectors for Ports 1 and 2
    guidedCal.ConnectorType(1) = "APC 3.5 female"
    guidedCal.ConnectorType(2) = "APC 3.5 male"

    ' If your VNA has more than 2 ports, uncomment one or both of
    ' these next two lines, to explicitly specify this is
    ' just a 2-port cal.
    'guidedCal.ConnectorType(3) = "Not used"
    'guidedCal.ConnectorType(4) = "Not used"

    ' Specify cal kit for Ports 1 and 2
    guidedCal.CalKitType(1) = "85052C"
    guidedCal.CalKitType(2) = "85052C"

    ' Since the 85052C cal kit contains SOLT standards and also TRL
    ' standards, these next lines determine whether the cal becomes
    ' unknown thru (SOLT), or TRL.
    ' Specify cal and Thru method
    guidedCal.PathThruMethod (1,2) = "Undefined Thru"

End Sub
```
' To set up the cal as TRL, comment the previous line and uncomment this next line.
The Thru method is set by default.
guidedCal.PathCalMethod (1,2) = "TRL"

' Always send Initialize after modifying the SmartCal logic
guidedCal.Initialize chanNum, True
numSteps = guidedCal.GenerateSteps
MsgBox "Number of steps is " + CStr(numSteps)

' Query the list of ports that need delta match
portList = guidedCal.PortsNeedingDeltaMatch
' If portList contains just one element and it's value is 0, then that indicates
' none of the ports being calibrated require delta match data.
' If each testport on the VNA has it's own reference receiver (R channel),
' then delta match is never needed, so portList will always be just 0.
lowerBound = LBound(portList)
If (UBound(portList) <> lowerBound) Or (portList(lowerBound) <> 0) Then
' Delta match data is required for at least one port.
' For this example, we assume a Global Delta Match Cal has previously been
' performed so the Global Delta Match CalSet exists.
' Supplying an empty string to ApplyDeltaMatchFromCalSet indicates to use
' the Global Delta Match CalSet.
guidedCal.ApplyDeltaMatchFromCalSet ""
End If

' Measure the standards
For i = 1 To numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
strPrompt = guidedCal.GetStepDescription(i)
retVal = MsgBox(strPrompt, vbOKCancel, step)
If retVal = vbCancel Then Exit Sub
guidedCal.AcquireStep i
Next

' Conclude the calibration
guidedCal.GenerateErrorTerms
MsgBox "Cal is done!"

End Sub
Perform Global Delta Match Cal

The following program performs a Global Delta Match Calibration. This is required when performing an Unknown Thru cal or TRL cal on VNAs without a reference receiver for each test port. See example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Delta.vbs. Learn how to setup and run the macro.

```vbs
Sub PerformGlobalDeltaMatchCal()
    ' Create / Get the VNA application.
    Set app = CreateObject("AgilentPNA835x.Application")
    ' Get cal manager object
    Set calMgr = app.GetCalManager
    ' Get guided cal object
    Set guidedCal = calMgr.GuidedCalibration

    ' Initiate a Global Delta Match calibration, choosing connector and cal kit
    numSteps = guidedCal.GenerateGlobalDeltaMatchSequence("APC 3.5 female", "85033D/E")
    MsgBox "Number of steps is " + CStr(numSteps)

    ' Measure the standards
    For i = 1 To numSteps
        step = "Step " + CStr(i) + " of " + CStr(numSteps)
        strPrompt = guidedCal.GetStepDescription(i)
        retVal = MsgBox(strPrompt, vbOKCancel, step)
        If retVal = vbCancel Then Exit Sub
        guidedCal.AcquireStep i
    Next

    ' Conclude the calibration
    guidedCal.GenerateErrorTerms
    MsgBox "Cal is done!"
End Sub
```
Perform a Guided Cal with C#

The following example performs a 2-port or 4-port Guided Cal using C#.

**Note:** Replace `<remote host name>` with the full computer name of your VNA.

**Learn more about using .NET with the VNA**

```csharp
AgilentPNA835x.IApplication pna;
AgilentPNA835x.ICalManager3 calMgr;
AgilentPNA835x.IGuidedCalibration guidedCal;
AgilentPNA835x.IChannel chan;
int chanNum;
AgilentPNA835x.ICalSet calset;

void PerformGuidedCal()
{
    Type pnaType = Type.GetTypeFromProgID("AgilentPNA835x.Application", "<remote host name>"));
    pna = (AgilentPNA835x.IApplication)
          Activator.CreateInstance(pnaType);
    calMgr = (AgilentPNA835x.ICalManager3)pna.GetCalManager();
    guidedCal = (AgilentPNA835x.IGuidedCalibration)calMgr.GuidedCalibration;
    chan = pna.ActiveChannel;
    chanNum = chan.channelNumber;

    // Initialize guided cal to be performed on the active channel. 
    // The boolean argument of True indicates to store the cal only 
    // in the channel's calibration register. If instead you wish 
    // to create a new calset that the new cal will get stored to, 
    // comment out this next line and uncomment the three lines below it.
```
guidedCal.Initialize(chanNum, true);
calset = calMgr.CreateCalSet(chanNum);
chan.SelectCalSet(calset.GetGUID(), true);
guidedCal.Initialize(chanNum, false);

    // To perform 2-port cal, Uncomment the following
    // Then comment the 4-port cal

    // Do 2-port cal
    // TwoPortGuidedCal();

    // Do 4-port cal
    FourPortGuidedCal();
    
}

void TwoPortGuidedCal()
{
    // Select the connectors
    guidedCal.set_ConnectorType(1, "APC 3.5 female");
guidedCal.set_ConnectorType(2,"APC 3.5 male");
for (int i = 3; i <= pna.NumberOfPorts; i++)
    guidedCal.set_ConnectorType(i, "Not used");
MessageBox.Show("Connectors defined for Ports 1 and 2");
    // Select the Cal Kit for each port being calibrated.
guidedCal.set_CalKitType(1,"85052D");
guidedCal.set_CalKitType(2,"85052D");
    // To use an ECal module instead, comment out the above two lines
// and uncomment the following two lines.
// Replace N4691-60004 with your own ECAL model followed by 'ECal'.

// Your ECal module must already be connected to a PNA USB port.
// guidedCal.CalKitType(1) = "N4691-60004 ECal"
// guidedCal.CalKitType(2) = "N4691-60004 ECal"
MessageBox.Show("Cal kits defined for Ports 1 and 2");

// Initiate the calibration and query the number of steps
int numSteps = guidedCal.GenerateSteps();

// Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps);

void FourPortGuidedCal()
{

    //Select the connectors
    guidedCal.set_ConnectorType(1,"APC 3.5 female");
    guidedCal.set_ConnectorType(2,"APC 3.5 female");
    guidedCal.set_ConnectorType(3,"APC 3.5 female");
    guidedCal.set_ConnectorType(4,"APC 3.5 female");
    // If a PNA which has more than 4 ports
    for (int i = 5;i<=pna.NumberOfPorts; ++i)
    {
        guidedCal.set_ConnectorType(i,"Not used");
    }
    MessageBox.Show("Connectors defined for Ports 1 to 4");

    // Select the Cal Kit for each port being calibrated.
    guidedCal.set_CalKitType(1,"85052D");
    guidedCal.set_CalKitType(2,"85052D");
    guidedCal.set_CalKitType(3,"85052D");
}
guidedCal.set_CalKitType(4, "85052D");

// To use an ECal module instead, comment out the above four lines
// and uncomment the following four lines.
// Replace N4691-60003 with your own ECAL model followed by 'ECal'.
// Your ECal module must already be connected to a PNA USB port.
//guidedCal.CalKitType(1) = "N4431-60003 ECal";
//guidedCal.CalKitType(2) = "N4431-60003 ECal";
//guidedCal.CalKitType(3) = "N4431-60003 ECal";
//guidedCal.CalKitType(4) = "N4431-60003 ECal";
MessageBox.Show("Cal kits defined for Ports 1 to 4");

// Initiate the calibration
guidedCal.GenerateSteps();

// If your selected cal kit is not a 4-port ECal module which can
// mate to all 4 ports at once, then you may want to choose which
// thru connections to measure for the cal. You must measure at
// least 3 different thru paths for a 4-port cal (for greatest
// accuracy you can choose to measure a thru connection for all 6
// pairings of the 4 ports). If you omit this command, the default
// is to measure from port 1 to port 2, port 1 to port 3, and
// port 1 to port 4. For this example we select to measure
// from port 1 to port 2, port 2 to port 3, and port 2 to port 4.
long[] portList = new long[6]{1,2,2,3,2,4};
guidedCal.ThruPortList = portList;

// Re-generate the connection steps to account for the thru changes
int numSteps = guidedCal.GenerateSteps();

// Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps);
```csharp
void MeasureAndComplete(int numSteps)
{
    MessageBox.Show("Number of steps is " + numSteps.ToString());

    // Measure the standards
    // The following series of commands shows that standards
    // can be measured in any order. These steps acquire
    // measurement of standards in reverse order.
    // It is easiest to iterate through standards using
    // a For-Next Loop.
    for (int i = numSteps; i >= 1; --i)
    {
        string strPrompt = guidedCal.GetStepDescription(i);
        MessageBox.Show(strPrompt);
        guidedCal.AcquireStep(i);
    }

    // Conclude the calibration
    guidedCal.GenerateErrorTerms();
    MessageBox.Show("Cal is done!");
}
```
Perform an ECal using COM

This example uses the GuidedCalibration interface to perform a 2-port ECal calibration.

Learn more about Reading and Writing Calibration data using COM.

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as ECal.vbs. Learn how to setup and run the macro.

```vbs
Set pna = CreateObject("AgilentPNA835x.Application")
Set calMgr = pna.GetCalManager
Set guidedCal = calMgr.GuidedCalibration
Set chan = pna.ActiveChannel
chanNum = chan.ChannelNumber
' Initialize guided cal to be performed on the active channel.
' The boolean argument of True indicates to create a new calset
guidedCal.Initialize chanNum, True
' To perform 3-port cal, Uncomment the following
' Then comment the 2-port cal

' Do 2-port cal
TwoPortGuidedCal

' Do 3-port cal
' ThreePortGuidedCal

Sub TwoPortGuidedCal()
'Change the following to match the connectors on your ECal module
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 female"
For i = 3 To pna.NumberOfPorts
    guidedCal.ConnectorType(i) = "Not used"
Next
value = MsgBox("Connectors defined for Ports 1 and 2")

' Select the ECal module for each port being calibrated.

' Replace N4691-60004 with your own ECAL model followed by 'ECal'.

' Your ECal module must already be connected
```
' via USB to the VNA.

guidedCal.CalKitType(1) = "N4691-60004 ECal"
guidedCal.CalKitType(2) = "N4691-60004 ECal"

' Non-factory characterizations are specified as follows:

'guidedCal.CalKitType(1) = "N4691-60004 User 1 ECal"

' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:

'guidedCal.CalKitType(1) = "N4691-60004 ECal 01234"

' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:

'guidedCal.CalKitType(1) = "N4691-60004 MyDskChar ECal 01234"

' Turn on auto orientation for the ECal (default behavior).

'guidedCal.AutoOrient = 1'

MsgBox("Cal kits defined for Ports 1 and 2")

' Initiate the calibration and query the number of steps
numSteps = guidedCal.GenerateSteps
'Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

Sub ThreePortGuidedCal()
' Change the following to match the connectors on your ECal module
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 female"
guidedCal.ConnectorType(3) = "APC 3.5 female"

' Select the ECal module for each port being calibrated.
' Replace N4691-60003 with your own ECAL model followed by 'ECal'.
' Your ECal module must already be connected to a VNA USB port.
guidedCal.CalKitType(1) = "N4431-60003 ECal"
guidedCal.CalKitType(2) = "N4431-60003 ECal"
guidedCal.CalKitType(3) = "N4431-60003 ECal"

value = MsgBox("Cal kits defined for Ports 1 to 3")
' Initiate the calibration
numSteps = guidedCal.GenerateSteps
'Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub
Sub MeasureAndComplete(ByVal numSteps)
    value = MsgBox("Number of steps is " + CStr(numSteps))
    ' Measure the standards
    For i = 1 To numSteps
        step = "Step " + CStr(i) + " of " + CStr(numSteps)
        strPrompt = guidedCal.GetStepDescription(i)
        value = MsgBox(strPrompt, vbOKOnly, step)
        guidedCal.AcquireStep i
    Next

    ' Conclude the calibration
    guidedCal.GenerateErrorTerms
    MsgBox("Cal is done!")
End Sub
Perform an ECal User Characterization

This example performs a user-characterization and stores it to both the ECal module memory and VNA disk memory.

It then performs two 2-port cals: the first using the characterization from module memory, then using the characterization from disk memory.

**Note:** This example requires that channel 1 be already calibrated.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as ECal.vbs.

### See Also
- How to setup and run the macro.
- ECalUserCharacterizer Object
- About User Characterization

```vbnet
Option Explicit

Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")

Dim calMgr
Set calMgr = pna.GetCalManager

' Get ECal User Characterizer COM object
Dim ecalCharacterizer
Set ecalCharacterizer = calMgr.GetECalUserCharacterizer

' Substitute here the model number and serial number of your own ECal.
' Note that this example corresponds to a 4-port ECal module with
' serial number 00001. If instead you have a 2-port ECal module,
' their model numbers are '5x5' numbers -- for example, 'N4691-60001'.
```
Dim ecalModelNum
ecalModelNum = "N4433A"
Dim ecalSerialNum
ecalSerialNum = "00001"
ecalCharacterizer.ECalID = ecalModelNum & "," & ecalSerialNum
MsgBox "ECal module to be characterized is: " & ecalCharacterizer.ECalID

' Set which user characterization number (1-12) the new characterization
' will be stored to in the ECal module when it is done.  If you intend to
' store your user characterization just to VNA Disk Memory and NOT the
' ECal module's memory, then omit the setting of this property.
ecalCharacterizer.CharacterizationNumber = 1

' The following commented-out lines of code show how you can access
' the list of connector type names you can set for the ports of an
' ECal when you user-characterize it.  However, please note that if
' you are writing the user characterization to the ECal module's memory,
' as of yet only the Factory Defined set of connector choices will work
' properly (see the ValidConnectorType property).
'
' If you will be saving your characterization to just
' VNA Disk Memory only, then all connector names returned by this call
' will work, user-defined connector names as well as factory-defined.
'
'Dim connTypeArray
'connTypeArray = ecalCharacterizer.ValidConnectorTypes
'MsgBox connTypeArray(1)

' Access element 1 in the string array
' For each port of the ECal module, specify which connector type
' is at the end of the adapter (or cable or fixture) that is
' connected to that port of the ECal for the characterization
' (must be one of the connector types that is included in the
list that the ValidConnectorTypes method returns). The default
is "No adapter", which assumes you are characterizing that port
of the ECal "as is" (nothing attached to it). So in this example,
Ports C and D of the ECal are being characterized to just the
ECal's connectors.

ecalCharacterizer.ConnectorType(1) = "APC 3.5 male" ' ECal Port A
ecalCharacterizer.ConnectorType(2) = "APC 3.5 male" ' ECal Port B

As with the connector types, the information set in these next
few properties also gets stored within the characterization.
Set the name of the person and/or company that is producing
this characterization.

ecalCharacterizer.UserName = "John Doe, Acme Inc."

Set user-specified description of the VNA being used.

ecalCharacterizer.UserDescriptionOfPNA = "SN US12345678"

Set descriptions of what you have connected to the ECal module's
ports for the characterization.

ecalCharacterizer.PortDescription(1) = "3.5 mm adapter, SN 00001" ' Port A of the
ECal
ecalCharacterizer.PortDescription(2) = "3.5 mm adapter, SN 00002" ' Port B of the
ECal

Begin a user characterization on Channel 1.

If you will be storing this characterization to the ECal module's memory, then
the boolean argument to this command should be set to True. If you will be storing
this characterization to VNA disk memory ONLY, then you should specify False for
that argument. In this example we will be storing the characterization to both
module memory and VNA disk memory, so we use True.

ecalCharacterizer.InitializeEx 1, True

Generate the measurement steps for the user characterization.
Dim numSteps

numSteps = ecalCharacterizer.GenerateSteps

' Measure the steps.
' You must ensure you have already applied the appropriate calibration to the channel
' already, or else an error will be thrown indicating that.
Dim i

For i = 1 To numSteps
    MsgBox ecalCharacterizer.GetStepDescription(i)
    ecalCharacterizer.AcquireStep(i)
    MsgBox "Acquire is complete"
Next

MsgBox "Now the user characterization will be saved to the ECal module and to PNA disk memory"

' Save the user characterization to the ECal module's memory.
' Note that this can take multiple minutes, depending on how many sweep points the channel has.
ecalCharacterizer.SaveToECal

' Save the user characterization to VNA Disk Memory.
Dim characterizationName

characterizationName = "test"
ecalCharacterizer.SaveToDiskMemory(characterizationName)

MsgBox "User characterization is complete. Now we will calibrate using it. First we will use it from ECal module memory."

Dim moduleMemCalKitName

moduleMemCalKitName = GetCalKitName("User " & CStr(ecalCharacterizer.CharacterizationNumber))

DoTwoPortCal moduleMemCalKitName

MsgBox "Now we will calibrate using the characterization from PNA Disk Memory."
Dim pnaDiskMemCalKitName

pnaDiskMemCalKitName = GetCalKitName(characterizationName)

DoTwoPortCal pnaDiskMemCalKitName

MsgBox "Example has completed"

Function GetCalKitName(characterizationName)

Dim calKitName

calKitName = ecalModelNum

If Len(characterizationName) > 0 Then calKitName = calKitName & " " & characterizationName

calKitName = calKitName & " ECal " & ecalSerialNum

GetCalKitName = calKitName

End Function

Sub DoTwoPortCal(calKitName)

' Initialize guided cal to be performed on Channel 1.

Dim guidedCal

Set guidedCal = calMgr.GuidedCalibration

guidedCal.Initialize 1, True

' Specify the DUT connector for each VNA port to be calibrated (DUT connector = ECal characterization's connector)

guidedCal.ConnectorType(1) = "APC 3.5 male"

guidedCal.ConnectorType(2) = "APC 3.5 male"

' Specify the "cal kit" for each of those ports

guidedCal.CalKitType(1) = calKitName

guidedCal.CalKitType(2) = calKitName

' We know this example will result in a calibration sequence of a single "connection step"

Dim numSteps

' 3116
numSteps = guidedCal.GenerateSteps

' Acquire the cal connection step

guidedCal.AcquireStep 1

' Conclude the cal and turn it on

guidedCal.GenerateErrorTerms

End Sub
Perform a Comprehensive 2-Port Guided Cal

This example program performs a Guided Calibration on the active channel between ports 1 and 2. The following calibration features are demonstrated:

- Guided Power Cal
- Optional functions when using ECal
- Select the Thru method
- Save to a new CalSet

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as CompGuided.vbs. Learn How to setup and run the macro.

See Also

VNA Object Model
CalManager Object
GuidedCalibration Object
Calibrator Object

See Other COM Example Programs

' Performing a Guided 2-port cal (Ports 1 and 2)
TwoPortGuidedCal
Sub TwoPortGuidedCal
' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set calMgr = app.GetCalManager
Set guidedCal = calMgr.GuidedCalibration
Set chan = app.ActiveChannel
chanNum = chan.ChannelNumber
Initialize guided cal to be performed on the active channel.
The boolean argument of True indicates to create a new calset for storing the new calibration to.
guidedCal.Initialize chanNum, True

Query the connectors that the VNA system recognizes
conns = guidedCal.ValidConnectorTypes

Format the list string with linefeed characters between each substring
connList = FormatList(conns)

Select the connector for Port 1
selectedConn1 = InputBox("Enter your DUT connector for Port 1. Choose from this list:" & 
                       Chr(10) & Chr(10) & connList)

If selectedConn1 = "" Then Exit Sub

guidedCal.ConnectorType(1) = selectedConn1

Select the connector for Port 2

selectedConn2 = InputBox("Enter your DUT connector for Port 2. Again, choose from this list:" & 
                         Chr(10) & Chr(10) & connList)

If selectedConn2 = "" Then Exit Sub

guidedCal.ConnectorType(2) = selectedConn2

Note: If your VNA has more than 2 ports, you would need to uncomment one or both of these next two lines, to explicitly specify this is just a 2-port cal.

'guidedCal.ConnectorType(3) = "Not used"
'guidedCal.ConnectorType(4) = "Not used"

Query the list of acceptable cal kits and ECal module characterizations for Port 1.
kits = guidedCal.GetCompatibleCalKits(selectedConn1)

Format the list string with linefeed characters between each substring
kitList = FormatList(kits)

' Select the Cal Kit or ECal module characterization to use for Port 1.
selectedKit = InputBox("Enter your cal kit or ECal module characterization for Port 1. " & _
                       "Choose from this list:" & Chr(10) & Chr(10) & kitList)
If selectedKit = "" Then Exit Sub
guidedCal.CalKitType(1) = selectedKit

' Query the list of acceptable cal kits and ECal module characterizations for Port 2.
kits = guidedCal.GetCompatibleCalKits(selectedConn2)

' Format the list string with linefeed characters between each substring
kitList = FormatList(kits)

' Select the Cal Kit or ECal module characterization to use for Port 2.
selectedKit = InputBox("Enter your cal kit or ECal module characterization for Port 2. " & _
                       "Choose from this list:" & Chr(10) & Chr(10) & kitList)
If selectedKit = "" Then Exit Sub
guidedCal.CalKitType(2) = selectedKit

' This determines whether the cal will be a "Guided Power Cal"
' or just a traditional S-parameter cal.
message = "On which port number shall power be measured? "
message = message & "For a traditional guided cal without power cal, enter 0"
Dim powerPort
powerPort = CInt( InputBox(message) )
If powerPort > 0 Then
    guidedCal.PerformPowerCalibration(powerPort) = True
    Dim retVal
    retVal = MsgBox("Is the power sensor's connector type or gender different from the DUT connector for that port?", vbYesNo)
    If retVal = vbYes Then
message = "Enter your power sensor's connector. Choose from this list:"
message = message & Chr(10) & Chr(10) & connList

' Select the sensor's connector.
selectedConn1 = InputBox(message)
If selectedConn1 = "" Then Exit Sub
guidedCal.PowerSensorConnectorType(powerPort) = selectedConn1

' Query the list of acceptable cal kits and ECal module characterizations
' that are applicable for the sensor's connector.
kits = guidedCal.GetCompatibleCalKits(selectedConn1)

' Format the list string with linefeed characters between each substring
kitList = FormatList(kits)

message = "Enter your cal kit or ECal module characterization to use for
de-embed of the sensor's connector."
message = message & "Choose from this list:"
message = message & Chr(10) & Chr(10) & kitList

' Select the Cal Kit or ECal module characterization to use for de-embed of
the sensor's connector.
selectedKit = InputBox(message)
If selectedKit = "" Then Exit Sub
guidedCal.PowerSensorCalkitType(powerPort) = selectedKit Else

guidedCal.PowerSensorConnectorType(powerPort) = "Ignored"
End If ' End of block that considers the sensor's connector

' Ask for the power level to perform the power cal at
' (if this command is omitted, the default is 0 dBm).
Dim powerLevel

powerLevel = InputBox("Enter the power level for the power cal to be performed
at")
If powerLevel = "" Then Exit Sub
guidedCal.PowerCalibrationPowerLevel(powerPort) = CDbl(powerLevel)

Else

guidedCal.PerformPowerCalibration(1) = False

End If ' End of block that considers if the cal will include power calibration

'----------------------------------------------------------

' This next block of commented-out code shows optional functions when using ECal.
' These OrientECALModule and ECALPortMapEx properties would need to be set prior to
' calling GenerateSteps on the guidedCal object.
' Read the information about the Keysight factory characterization data
' of ECal module #1 on the USB bus
'Set calibrator = chan.Calibrator
'Const ECalModule1 = 1
'module1Info = calibrator.GetECALModuleInfoEx(ECalModule1)
'MsgBox "Description of ECal module #1:" & Chr(10) & Chr(10) & module1Info
' By default, during calibration the VNA automatically determines the orientation of
' the ECal module (senses which port of the module is connected to which port of
' the VNA).
' However, since this setting could have recently been overridden by another user of
' the instrument, use this next line to ensure the auto orientation setting is
' enabled.
'calibrator.OrientECALModule = True
' Alternatively, if you are measuring at very low power levels where
' the VNA fails to sense the module's orientation, you may need to turn off the auto
' orientation and specify how the module is connected (as in these next two lines of code,
' "A1,B2" would indicate Port A of the module is connected to Port 1 and
' Port B is connected to Port 2).
'calibrator.OrientECALModule = False
'calibrator.ECALPortMapEx( ECalModule1 ) = "A1,B2"
' End of optional ECal setup

'--------------------------------------------------------------------------------
---

' Select the thru method of Default. This instructs the VNA to determine which
' standard measurement technique to use, based upon the selected connectors and
' calibration kit(s) and what model of VNA this is.
guidedCal.ThruCalMethod = 0 ' 0 = naDefaultCalMethod

' Initiate the calibration and query the number of steps
numSteps = guidedCal.GenerateSteps

MsgBox "Number of steps is " + CStr(numSteps)

' Measure the standards
For i = 1 To numSteps
    step = "Step " + CStr(i) + " of " + CStr(numSteps) + ""
    strPrompt = guidedCal.GetStepDescription(i)
    MsgBox strPrompt, vbOKOnly, step
    guidedCal.AcquireStep i
Next

' Conclude the calibration
guidedCal.GenerateErrorTerms

MsgBox "Cal is done!"

End Sub

Function FormatList(tokens)
    For i = 0 To UBound(tokens)
        list = list & tokens(i) & Chr(10)
    Next
Next

FormatList = list

End Function
ECAL Confidence Check

This Visual Basic program:

- Initializes the VNA objects.
- Performs a complete ECAL confidence check

Before using this code:

- The active channel must contain an S11 measurement with a 1-port or N-port calibration
- Prepare a form with two buttons named cmdRun and cmdQuit

Note: A confidence check can NOT be performed remotely from User Characterizations that are stored on the VNA disk.

```vba
Private oPNA As AgilentPNA835x.Application
Private oChan As Channel
Private oCal As Calibrator
Private oMeas As Measurement

Private Sub cmdRun_Click()
Dim iMeasIndex As Integer

Set oPNA = CreateObject("AgilentPNA835x.Application", "MachineName")
Set oChan = oPNA.ActiveChannel
Set oCal = oChan.Calibrator

iMeasIndex = 1

' Loop through measurements until an S11 on the active channel
' is found, or the end of the measurement collection is reached.
```
Do
  Set oMeas = oPNA.Measurements(iMeasIndex)
  If oMeas.Parameter = "S11" And _
    oMeas.channelNumber = oChan.channelNumber Then Exit Do
  iMeasIndex = iMeasIndex + 1
  If iMeasIndex > oPNA.Measurements.Count Then
    MsgBox "No S11 measurement found on the active channel." _
    " Create an S11 measurement, then try again."
    Exit Sub
  End If
Loop

' Set up trace view so we are viewing only the data trace.
oMeas.View = naData

' Acquire the S11 confidence check data from ECal Module A
' into the memory buffer.
oCal.AcquireCalConfidenceCheckECALEx "S11", 1

' Turn on trace math so the trace shows data divided by memory.
' You can be confident the S11 calibration is reasonably good if
' the displayed trace varies no more than a few tenths of a dB
' from 0 dB across the entire span.
oMeas.TraceMath = naDataDivMemory
End Sub

Sub cmdQuit_Click()
' Turn off trace math
' in case someone clicks Quit without having clicked Run
If oMeas <> Nothing Then oMeas.TraceMath = naDataNormal
' Conclude the confidence check to set the ECal module
' back to it's idle state.
If oCal <> Nothing Then oCal.DoneCalConfidenceCheckECAL
' End the program
    End
End Sub
The following text is an example of the content of a user defined power meter uncertainty file.

This simple ascii file contains the power meter uncertainty contributions summarized in the following table. Any user defined file with this format can be loaded by setting the model as “Custom File”.

<table>
<thead>
<tr>
<th>Standard Uncertainty</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>u_Pm</td>
<td>Power meter instrumentation uncertainty.</td>
</tr>
<tr>
<td>u_Pmc</td>
<td>Power meter instrumentation uncertainty (during calibration).</td>
</tr>
<tr>
<td>u_D</td>
<td>Power meter drift uncertainty.</td>
</tr>
<tr>
<td>u_Pcal</td>
<td>Calibrator output power level uncertainty.</td>
</tr>
<tr>
<td>u_Zs</td>
<td>Power meter zero set uncertainty.</td>
</tr>
<tr>
<td>u_N</td>
<td>Power meter and sensor noise uncertainty.</td>
</tr>
<tr>
<td>u_Zc</td>
<td>Power meter zero carryover uncertainty.</td>
</tr>
<tr>
<td>u_Mu</td>
<td>Mismatch gain uncertainty between the sensor and the generator.</td>
</tr>
<tr>
<td></td>
<td>The standard uncertainty is dependent upon the reflection coefficients of the sensor and the generator.</td>
</tr>
<tr>
<td>u_Muc</td>
<td>Mismatch gain uncertainty between the sensor and the calibrator output of the power meter. The standard uncertainty is dependent upon the reflection coefficients of the sensor and the calibrator output.</td>
</tr>
<tr>
<td>u_Kc</td>
<td>Sensor calibration factor uncertainty at the frequency of the power meter calibrator output.</td>
</tr>
</tbody>
</table>

See Also

ExternalDevices Collection

ExternalDevice Object

PowerSensor Object

PowerSensorAsReceiver Object

PowerSensorCalFactorSegmentPMAR Object

PowerLossSegmentsPMAR_Collection
PowerLossSegmentPMAR Object

Uncertainty on Power Meter

Power Meter Uncertainty dialog description

Application Note (5988-9215EN)

//
// This file is an example
// of pwrmtr-sensor uncertainty data
//
[PwrMtr]
SerialNumber=NOTSET

// Power Meter Measurement
// Uncertainty (%)
u_Pm=0.21

// Power Meter Measurement
// Uncertainty during calibration (%)
// If not specified we use the same as u_Pm
u_Pmc=0.21

// Zero drift Uncertainty (W)
u_D=5.5E-9

// Internal output power level
// uncertainty, ie.
// Internal calibration accuracy (%)
// ±0.59% (0 to 55 °C)
u_Pcal=0.59

//
// Pcal
// Power used for the
data mtr calibration (dBm)

Pcal=0.0

// Zero set Uncertainty (W)

u_Zs=25E-9

// Measurement Noise (W)

u_N=45E-9

// Zero Carryover

u_Zc=0.0

[SensGen]

SerialNumber=NOTSET

// Mismatch Gain

// NOT TO BE SET

u_Mu=0.000000

// Mismatch gain from

// the sensor and the ref source

// NOT TO BE SET FOR USB

u_Muc=0.0

// Sensor CalFactor uncertainty

// @ the cal output frequency

// For usb 50MHz (%)

u_Kc=0.0198

//

// Sensor CalFactor
// @ the cal output frequency
// For usb 50MHz
Kc=100
m_TargetPwr=0.000000

[CalFact]

//
// CalFactor vs Frequency
// IN this case is directly
// read from the sensor
// This is left for example
// as comment
// Nele=2 Number of Freqs
// 0=1.000000,1.000000 freq,value
// 1=2.000000,2.100000

[CalFactUnc]

//
// Cal Factor UNcertainty (%) vs Freq

Nele=18
0=0.000000,2.88
1=0.010000,2.88
2=0.010000,2.04
3=0.030000,2.04
4=0.030000,1.98
5=0.50000,1.98
6=0.50000,2.07
7=1.20000,2.07
8=1.20000,2.40
9=6.00000,2.40
10=6.0000,2.99
11=14.0000,2.99
12=14.0000,3.35
13=18.0000,3.35
14=18.0000,4.70
15=26.5000,4.70
16=26.5000,6.41
17=33.0000,6.41

[LinFactUnc]

//

// Power Sensor Linearity

// vs power

// Array of Nele as:

// pwr (dBm), unc (%)

// Full temperature range

//(0 to 55 C)

Nele=4

0=-1.000000,0.55
1=15.000000,0.55
2=15.000000,0.60
3=20.000000,0.60
This example creates a Cal Set and then writes data to the Cal Set.

Learn more about Reading and Writing Calibration data using COM.

Learn more about Cal Sets.

See example: Reading Calset Data

---

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as CalSetsWrite.vbs. Learn how to setup and run the macro.

```vbs
Dim pna
Dim v
Set pna = CreateObject("AgilentPNA835x.Application")
InitPhonyData
PutPhonyData
' This sub creates phony data
Sub InitPhonyData()
Dim i
Dim numpts
wscript.echo "init phony"
umpts = pna.ActiveChannel.NumberOfPoints
ReDim v(numpts - 1, 1)
For i = 0 To numpts - 1
v(i, 0) = i
v(i, 1) = 0
Next
End Sub
' This sub creates a Cal Set, then writes the phony data to it
Sub PutPhonyData()
Dim cmgr
Dim cset
wscript.echo "putphony"
Set cmgr = pna.GetCalManager
Set cset = cmgr.CreateCalSet(1)
cset.OpenCalSet naCalType_OnePort, 1
const directivity = 0
const sourcematch = 1
```

---
const reflectiontracking =2

cset.putErrorTerm directivity, 1, 1, v

cset.putErrorTerm sourcematch, 1, 1, v

cset.putErrorTerm reflectiontracking, 1, 1, v


cset.CloseCalSet

cset.Description = "Phony One Port"

cset.save

End Sub
Upload a Source Power Cal using COM

This program can be run in either Visual Basic 6 or as a VBScript program. The VNA can run *.vbs programs as macros.

This program demonstrates:

- Uploading a source power calibration of Port 2 for Channel 1.
- Reading the calibration data.

Learn more about Power Calibrations

See Other COM Example Programs

To run this program you need:

- Your PC and VNA both connected to a LAN (for communicating with each other).

To make this program work in VBS, save the following code in a text editor file such as Notepad and save as *.vbs.

To make this program work in Visual Basic 6:

1. Create a new project
2. Click Project, Add New Module, click Open.
3. Paste the following code into the code window.
4. Delete the first two lines (comment and Main)
5. Click Project, Properties. Under Startup Object, select Sub Main
6. Click Project, References, and select the Keysight PNA Series Type Library.

' Run the Main subroutine
Main
Public Sub Main()
Dim PNA, chan ' VNA COM objects
Const naCorrectionValues = 0 ' enum NASourcePowerCalBuffer
Const port = 2 ' VNA port #2 as source port
Dim stimulus, calvalues
Dim power, calpower, strResult

' Instantiate our VNA COM objects
Set PNA = CreateObject("AgilentPNA835x.Application")
Set chan = PNA.Channels(1)

' Set the number of sweep points to 2 on Channel 1.
chan.NumberOfPoints = 2

' Ensure there's currently no source power cal on for this channel and port.
chan.SourcePowerCorrection(port) = False

' Specify if the cal power level is offset (positive value for a gain, negative
' value for a loss) from the VNA port power setting on the channel when
' no source power cal is active. This is to account for components
' between the VNA test port and cal reference plane.
' In this example, let's set up our calibration
' at the output of an amplifier with 15 dB gain.
chan.SourcePowerCalPowerOffset(port) = 15

' Send our source power correction data to the VNA. For purpose of simplicity
' in this example, we'll set up for no correction (0) at our start stimulus and
' 0.5 dB at our stop stimulus (recall that our sweep currently has just 2 points).
calvalues = Array(0, 0.5)
chan.putSourcePowerCalDataEx naCorrectionValues, port, calvalues

' Set the number of sweep points to 21 on Channel 1.
chan.NumberOfPoints = 21

' Read the fixed power level for this port on Channel 1.
power = chan.TestPortPower(port)

' Turn the source power cal on.
chan.SourcePowerCorrection(port) = True

' Again read the fixed power level for this port on Channel 1
' (with our calibration turned on, this should now include the 15 dB offset
' we indicated our power amplifier provides).
calpower = chan.TestPortPower(port)

' Read the stimulus values from Channel 1.
stimulus = chan.GetXAxisValues

' Read back the source power correction data, now interpolated for 21 points
calvalues = chan.getSourcePowerCalDataEx(naCorrectionValues, port)

' Print the data using a message box (here, Chr returns the ASCII characters
' for Tab (9) and Linefeed (10)).
strResult = "PNA port power = " & power & Chr(10)
strResult = strResult & "Power at reference plane = " & calpower & Chr(10) & hr(10)
strResult = strResult & "Stimulus" & Chr(9) & Chr(9) & "Cal Value" & Chr(10)
For i = 0 To UBound(stimulus)
    strResult = strResult & stimulus(i) & Chr(9) & calvalues(i) & Chr(10)
Next
MsgBox strResult
End Sub
Upload and Download Segment Table

These example programs use the SetAllSegments Method and GetAllSegments Method to do the following:

- Creates a 2-dimensional array (7 x 10) 7 data elements that define each segment x 10 segments
- Uploads the data to the VNA
- Downloads a segment table from the VNA

This program does not make sweep type = segment or show the segment table.

The comments indicate the order in which the segment elements are specified: Index 0 - segment state, Index 4 is IFBW, and so forth.

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as *.vbs. Learn how to setup and run the macro.

```vbs
' Create the application instance, and preset the application
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset

Dim chan
Set chan = app.ActiveChannel
chan.sweeptype = 4

Dim segs
Set segs = chan.Segments

Dim win
Set win = app.NAWindows(1)
win.ShowTable 2

' Multipliers
kHz = 1000
MHz = kHz*1000
GHz = MHz*1000
' Create segments from 10MHz to 3GHz
StartFreq = 10 * MHz
```
StopFreq = 3 * GHz

' Create 10 segments between StartFreq and StopFreq

' Create a 2-D array of segments.
' 1st dimension is size 7 (6 is max index)
' to hold all the data per segment.
' 2nd dimension is size 10 (9 is max index)
' to hold 10 total segments.
Dim segdata(6, 9)
' Width of frequency segment, used below
SegmentWidth = (StopFreq - StartFreq) / 10
' Fill up all 10 segments (indices 0 to 9) with data
For i = 0 To 9
' element 0=segment state (on or off)
segdata(0, i) = True

' element 1=Num Points in this segment
segdata(1, i) = 500

' element 2=Start Freq
segdata(2, i) = StartFreq + i * SegmentWidth

' element 3=Stop Freq
segdata(3, i) = segdata(2, i) + SegmentWidth

' element 4=IFBW
segdata(4, i) = 35000

' element 5=Dwell Time
segdata(5, i) = 0

' element 6=Power
segdata(6, i) = 0
Next

' Configure Independent segment settings
segs.IFBandwidthOption = 1
segs.SourcePowerOption = 1

' Push the segment data into the VNA's Active Channel
segs.SetAllSegments segdata

Download Segment Table

Option Explicit

Dim app
Set app = CreateObject("AgilentPNA835x.Application")
Dim chan
Set chan = app.ActiveChannel
chan.sweeptype = 4
Dim segs
Set segs = chan.Segments
Dim win
Set win = app.NAWindows(1)
win.ShowTable 2
Dim segData
segData = segs.GetAllSegments
' Get lower bound and upper bound on the data values per each segment
Dim segDataLB, segDataUB
segDataLB = LBound(segData,1)
segDataUB = UBound(segData,1)
' Get lower bound and upper bound corresponding to how many segments
Dim segArrayLB, segArrayUB
segArrayLB = LBound(segData,2)
segArrayUB = UBound(segData,2)
' If the VB LBound and UBound functions didn't generate an error
' before reaching this point, that implies a valid two-dimensional
' array was returned into 'segData'.
WScript.Echo "Number of segments = " & segArrayUB - segArrayLB + 1
WScript.Echo "Number of data values per segment = " & segDataUB - segDataLB + 1
Dim index
index = segDataLB
Dim segInfStr
segInfStr = "Segment 1: state = " & segData(index, segArrayLB)
index = index + 1
segInfStr = segInfStr & ", num points = " & segData(index, segArrayLB)
index = index + 1
segInfStr = segInfStr & ", start freq = " & segData(index, segArrayLB)
index = index + 1
segInfStr = segInfStr & ", stop freq = " & segData(index, segArrayLB)
index = index + 1
segInfStr = segInfStr & ", IFBW = " & segData(index, segArrayLB)
index = index + 1
segInfStr = segInfStr & ", dwell time = " & segData(index, segArrayLB)

' In case of a measurement receiver VNA like N5264B
' which has no source ports, chan.SourcePortNames will
' return an empty variant (no array)

Dim srcPortNames

srcPortNames = chan.SourcePortNames

Dim srcPortNamesLB, srcPortNamesUB

srcPortNamesUB = -1

On Error Resume Next

srcPortNamesLB = LBound(srcPortNames)

srcPortNamesUB = UBound(srcPortNames)

On Error GoTo 0
If (srcPortNamesUB >= 0) And ((srcPortNamesUB - srcPortNamesLB + 1) <> (segDataUB - index)) Then

    WScript.Echo "Mismatch in number of source port names!"

End If

Dim j

For j = index + 1 To segDataUB
    segInfStr = segInfStr & ", " & srcPortNames(j - (index + 1) + srcPortNamesLB) & " power = " & segData(j, segArrayLB)
Next
WScript.Echo segInfStr
Create Multiple Instances of Calibrate All Channels

This example sets up multiple instances of CalibrateAllChannels.

**Note:** You can assign a single client channel to multiple cal all instances. However, care should be taken here. If you create user cal sets, all cal sets will be created but only the last one will be applied. If you are only using a cal register, only the last cal all will be written to the cal register (only supports one calibration). The order that the cal alls are created should be the order in which they are saved (GenerateErrorTerms). This ensures that the client channel imports the proper ETerms.

See Also

CalibrateAllChannels Object

Learn about Cal All

See other COM Examples

```vbs
'Access the CalAllChannels object
Dim app
Set app = CreateObject("Agilentpna835x.application","strange")
Dim mgr
Set mgr = app.GetCalManager
Dim scpi' As ScpiStringParser
Set scpi = app.ScpiStringParser
'Preset PNA
app.Preset
'Add extra channels
app.CreateCustomMeasurementEx 2, "Standard","S22",2
app.CreateCustomMeasurementEx 3, "Standard","S33",3
app.CreateCustomMeasurementEx 4, "Standard","S44",4
'********** Setup CalAll Channel
Dim CalAll
```
Set CalAll = mgr.CalibrateAllChannelsEx(1)
CalAll.Reset

' do it again
Dim CalAll2

Set CalAll2 = mgr.CalibrateAllChannelsEx(2) ' creates second CalAll2.Channels
CalAll.Channels = Array(1,2)
CalAll2.Channels = Array(3,4)
CalAll.UserCalsetPrefix = "MyCalAll"
CalAll2.UserCalsetPrefix = "MyCalAll2"

' CalAll.Reset

' Dim guidedcal

' set guidedcal = CalAll.GuidedCalibration

Dim guidedcal
set guidedcal = CalAll.GuidedCalibration
MsgBox "Doing First Guidedcal"

' Specify the DUT connectors

guidedcal.ConnectorType(1) = "APC 3.5 male"
guidedcal.ConnectorType(2) = "APC 3.5 male"
guidedcal.ConnectorType(3) = "Not used"
guidedcal.ConnectorType(4) = "Not used"
guidedCal.CalKitType(1) = "85052D"
guidedCal.CalKitType(2) = "85052D"
Numsteps = guidedcal.GenerateSteps

For i = 1 to Numsteps
    step = "Step " + CStr(i) + " of " + CStr(Numsteps)
    strPrompt = guidedCal.GetStepDescription(i)
value = MsgBox(strPrompt, vbOKOnly, step)
guidedCal.AcquireStep i
Next
guidedCal.GenerateErrorTerms
set guidedcal = CalAll2.GuidedCalibration
MsgBox "Doing second Guidedcal"

' Specify the DUT connectors
guidedcal.ConnectorType(1) = "APC 3.5 male"
guidedcal.ConnectorType(2) = "Not used"
guidedcal.ConnectorType(3) = "APC 3.5 male"
guidedcal.ConnectorType(4) = "APC 3.5 male"
guidedCal.CalKitType(1) = "85052D"
guidedCal.CalKitType(3) = "85052D"
guidedCal.CalKitType(4) = "85052D"
Numsteps = guidedcal.GenerateSteps
For i = 1 to Numsteps
    step = "Step " + CStr(i) + " of " + CStr(Numsteps)
    strPrompt = guidedCal.GetStepDescription(i)
    value = MsgBox(strPrompt, vbOKOnly, step)
    guidedCal.AcquireStep i
Next
guidedCal.GenerateErrorTerms
Create and Cal an SMC Measurement

This example creates and calibrates an SMC measurement. A power sensor must first be connected to the VNA.

By removing the comments (' ') at the start of the **BLUE code**, it can also do the following:

- Load a Mixer setup file from the VNA at: C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Mixer\MyMixer.mxr.
- Cal using an ECal module.
- Perform manual ECAL orientation
- Load a Phase Reference calibration.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as SMC.vbs. Learn how to setup and run the macro.

### See Also

- Create an SMC Fixed Output Meas
- Use Existing Power Cal for SMC

```vbs
Dim App
Set App = CreateObject("AgilentPNA835x.Application")
App.Preset
Dim Meas
Set Meas = App.ActiveMeasurement
Meas.Delete
App.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter", "SC21"
'Other valid strings that can be specified to create a measurement with a parameter
'other than"SC21" are: "S11", "S22", "IPwr", and "OPwr"
Dim chan
set chan = app.activechannel
```
'Attenuator setting must match optional Phase Ref Cal setting

chan.attenuator(1) = 10
chan.NumberOfPoints = 11
chan.IFBandwidth = 1000

'You can perform mixer setup here or
'recall a previous mixer setup from the VNA Hard drive.
'This is how to perform mixer setup using IConverter.

' Setup Stimulus
dim cv
set cv = chan.Converter
cv.InputStartFrequency = 3.6e9
cv.InputStopFrequency = 4.9e9
cv.LOFixedFrequency(1) = 1e9
cv.LOPower(1) = 10
cv.OutputSideband = 0 'Lowside
cv.Calculate 2 'Calc output
'cv.EnablePhase = True
cv.LOBName(1)="Port 3"
cv.Apply()

' Alternatively, recall a mixer setup from the VNA Hard drive
'Meas.LoadFile "C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Mixer\MyMixer.mxr"

' Begin Calibration
Dim CalMgr
Set CalMgr = App.GetCalManager
Dim SMC
Set SMC = CalMgr.CreateCustomCal("SMC")
SMC.Initialize 1, 1
SMC.ConnectorType(1) = "APC 3.5 male"
SMC.ConnectorType(2) = "APC 3.5 female"
' Use Mechanical cal kits
SMC.CalKitType(1) = "85033D/E"
SMC.CalKitType(2) = "85033D/E"
' To use an ECal module instead, comment out the above two lines
' and uncomment the appropriate lines below:
' Your ECal module must already be connected
' via USB to the VNA.
' SMC.CalKitType(1) = "N4691-60004 ECal"
' SMC.CalKitType(2) = "N4691-60004 ECal"
' Non-factory characterizations are specified as follows:
' SMC.CalKitType(1) = "N4691-60004 User 1 ECal"
' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
' SMC.CalKitType(1) = "N4691-60004 ECal 01234"
' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
' SMC.CalKitType(1) = "N4691-60004 MyDskChar ECal 01234"
' Turn on auto orientation for the ECal (default behavior).
' SMC.AutoOrient = 1'
MsgBox("Cal kits defined for Ports 1 and 2")
' Import power cal data from an existing SMC calset.
' This calset MUST exist on the VNA.
SMC.ImportDataSet "Phase Reference-full-span","POWER_AND_PHASE"
'Omit the isolation part of the 2-port cal (default behavior).
SMC.OmitIsolation = 1

Dim steps
steps = SMC.GenerateSteps
For i = 1 To steps
    MsgBox SMC.GetStepDescription(i)
    SMC.AcquireStep i
Next

Dim calset
calset = SMC.GenerateErrorTerms
Msgbox("SMC Cal Complete!")
Create and Cal a VMC Measurement

The following example program sets up a 1-stage mixer, then performs a VMC calibration using an N4691-60004 ECal module.

By removing the comments (' ') at the start of the **BLUE code**, it can also do the following:

- Use a mechanical cal kit
- Perform manual ECAL orientation
- Load a Mixer Characterization file

**See Also**

Converter Object

VMCType Object

**Example - Perform a VMC Mixer Characterization ONLY**

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as VMC.vbs. **Learn how to setup and run the macro.**

```
dim NASWEPT: NASWEPT = 0
dim NAFIXED: NAFIXED = 1
dim LOWSIDE: LOWSIDE = 0
dim HIGHSIDE: HIGHSIDE = 1
dim MIXEROUT: MIXEROUT = 2

dim pna: set pna = CreateObject("AgilentPNA835x.application")
pna.reset

' Create a VMC channel

' Other valid measurement strings are: "S11", and "S22"
pna.CreateCustomMeasurementEx 1, "Vector Mixer/Converter","VC21",1

' Setup Stimulus
```
dim chan: set chan = pna.activechannel

dim cv: set cv = chan.Converter
chan.NumberOfPoints = 11
chan.IFBandwidth = 1000
cv.InputStartFrequency = 3.6e9
cv.InputStopFrequency = 3.9e9
cv.LOFixedFrequency(1) = 1e9
cv.LOPower(1) = 10
cv.OutputSideband = LOWSIDE
cv.Calculate MIXEROUT
cv.LOName(1) = "Port 3"
cv.Apply()

DoBasicVMCCal (chan.channelNumber)

sub DoBasicVMCCal( channel )

dim myMixerCharFile: myMixerCharFile = "C:\Program Files(x86)\Keysight\Network Analyzer\Documents\MyMixerS2P.s2p"

' construct a VMC calibration object

dim calmanager: set calmanager = pna.GetCalManager
dim guidedCal: set guidedCal = calmanager.CreateCustomCalEx( channel )
dim vmc: set vmc = guidedCal.CustomCalConfiguration

' Initialize the cal object.

' Choose to respect or ignore the Preference: Cal: Auto Save to User Calset
' if you set this true, the behavior will be dependent on the setting
' of the preference.
dim useCalSetPreference: useCalSetPreference = false
vmc.Initialize channel, useCalSetPreference

' Define the DUT connectors and kits at ports 1 and 2 of the VNA
vmc.ConnectorType (1) = "APC 3.5 female"
vmc.ConnectorType (2) = "APC 3.5 male"

' Use Mechanical cal kits
vmc.CalKitType(1) = "85033D/E"
vmc.CalKitType(2) = "85033D/E"

' To use an ECal module instead, comment out the above two lines
' and uncomment the appropriate lines below:
' Your ECal module must already be connected
' via USB to the VNA.
' vmc.CalKitType(1) = "N4691-60004 ECal"
' vmc.CalKitType(2) = "N4691-60004 ECal"

' Non-factory characterizations are specified as follows:
' vmc.CalKitType(1) = "N4691-60004 User 1 ECal"
' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
' vmc.CalKitType(1) = "N4691-60004 ECal 01234"

' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
' vmc.CalKitType(1) = "N4691-60004 MyDskChar ECal 01234"

MsgBox("Cal kits defined for Ports 1 and 2")

' By default, VMC requires the measurement of a Calibration Mixer.
' To determine the conversion loss of the calmixer, the cal wizard
' will add a step to perform a 1 port cal at the output of the mixer.
' The following commands opt to perform the mixer
' characterization using a cal kit.
' Do both characterization and full 2-port cal

vmc.CharacterizeMixerOnly = False

' Define the DUT connectors for the output of the characterization mixer

' Use (logical) Port 3. If it is already used by the DUT,
' then specify port 4.
vmc.ConnectorType(3) = "APC 3.5 male"

' Specify the mechanical cal kit for port 3
vmc.CalKitType(3) = "85033D/E"

' To avoid performing the 1-port cal steps, provide the wizard with a
' mixer characterization file. Uncomment the following two lines to
' specify the characterization file. This is an .S2P file.
' vmc.CharFileName = myMixerCharFile   ' this file will be read
' vmc.LoadCharFromFile = true

' By default, auto orientation of the ecal module is performed

' Uncomment the following lines to manually orient the ecal
' vmc.autoorient = false

' for 2-port portion, ecal port A connected to VNA port 1
' vmc.EcalOrientation2Port(1) ="A1,B2"

' for mixer char, ecal port A connected to cal mixer output
' vmc.EcalOrientation1Port(1) = "A1"

' the main calibration loop

' a description for the connection instructions is read
' and then the standard is acquired
dim steps, connectionPrompt

steps = vmc.GenerateSteps

wscript.echo "Number of Steps = " + cstr(steps)

if (steps > 0) then  ' otherwise an error condition occurred
    for i = 1 to steps
        connectionPrompt = vmc.GetStepDescription( i )
        wscript.echo connectionPrompt
        vmc.AcquireStep( i )
    next

vmc.GenerateErrorTerms

end if

end sub
Create an SMC Fixed Output Measurement with COM

This VBScript example creates a calibrated SMC fixed output measurement using a controlled LO. Then a single sweep is taken and data is retrieved.

Requirements:

- If an external LO is used, it should be configured to match the LOName property of the mixer object.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as SMC.vbs. Learn how to setup and run the macro.

```vbs
option explicit

' Utility function
function ToString(complexDataArray)
    dim dataAsString
    dim point
    for point = 0 to UBound(data)
        dataAsString = dataAsString & "(" & data(point,0) & "," & data(point,1) & ")"
    next
    ToString = dataAsString
end function

dim app
set app = createobject("agilentpna835x.application")
app.preset

' Put the channel in hold (highly recommended)
app.ActiveChannel.Hold 1

' Delete the standard measurement
app.ActiveMeasurement.Delete
```
' Create an SC21 measurement
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","SC21"

' Set the number of points to 11
app.ActiveChannel.NumberOfPoints = 11

' Setup the mixer parameters for a swept LO, fixed output measurement

dim mixer

set mixer = app.ActiveMeasurement

mixer.InputStartFrequency = 200e6
mixer.InputStopFrequency = 700e6
mixer.LORangeMode(1) = 0  ' 0 = Swept mode
mixer.OutputFixedFrequency = 3.4e9
mixer.InputPower = -17
mixer.LOPower(1) = 10

'mixer.LOName(1) = "8360"

' The CALCULATE method calculates the LO frequency from the other parameters,
' It also applies ALL mixer parameters to the channel.
mixer.Calculate 3  ' Calculate the LO range

' Create an S11 in the same channel
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","S11"

dim S11Meas

set S11Meas = app.ActiveMeasurement

' Create an IPwr in the same channel
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","IPwr"

' Create an OPwr in the same channel
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","OPwr"

' Perform a single sweep synchronously.
app.ActiveChannel.Single 1

' Retrieve the SC21 data
dim data

' Get the calibrated values in polar format
data = mixer.GetData(1,3) ' 1 = naCorrectedData, 3 = naDataFormat_Polar
wscript.echo "SC21=" & ToString(data)

' Retrieve the S11 data

' Get the calibrated values in polar format
data = S11Meas.GetData(1,3) ' 3 = naDataFormat_Polar
wscript.echo "S11=" & ToString(data)
Create a Segmented Sweep for Mixers

This example program shows how to setup a segment sweep in FCA.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as Seg.vbs.

Learn how to setup and run the macro.

See Also

Converter Object

See Other COM Example Programs

```vbs
option explicit
Dim app, chan, conv
Set app = CreateObject("AgilentPNA835x.Application")
Set chan = app.ActiveChannel
Set conv = chan.GetConverter
app.Reset

' Create FCA Scalar Mixer/Converter channel with an SC21 measurement:
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter", "SC21", 1

' Delete all existing segments, and create three new ones
conv.DeleteAllSegments()
conv.AddSegment 1, 3

' Turn on segment 1
conv.SegmentState(1) = True

' Set segment sweep
' The sweeptype command discards the changes made to the scratch mixer
' Therefore, precede with Apply
' Also, always do this before setting the LO port
```
conv.Apply  

chan.SweepType = 4  'segment sweep  
' Setup segment #1  
' Input is swept from 1.1GHz to 1.39GHz  
conv.SegmentStartFrequency(1,0)=1.1e9  
conv.SegmentStopFrequency(1,0)=1.39e9  
'Swept input  
conv.SegmentRangeMode(1,0)=0  
' Input power is -10 dBm  
conv.SegmentFixedPower(1,0)=-10.0  
' LO1 is fixed: 2.2 GHz  
conv.SegmentFixedFrequency(1,2)=2.2e9  
' LO1 power is 10.0 dBm  
conv.SegmentFixedPower(1,2)=10.0  
' Number of points is 21  
conv.SegmentPoints(1)=21  
' Output is swept  
conv.SegmentRangeMode(1,1)=0  
' Output is low-side  
conv.SegmentMixingMode(1,1)=0  
' Output is calculated from input and lo1  
conv.SegmentCalculate 1,2  
' Turn on segment 1  
conv.SegmentState(1)=True  
' Setup segment #2 from 1.40 to 1.49 GHz  
' All else the same  
conv.SegmentStartFrequency(2,0)=1.4e9  
conv.SegmentStopFrequency(2,0)=1.49e9
conv.SegmentRangeMode(2,0)=0
conv.SegmentFixedPower(2,0)=-10.0
conv.SegmentFixedFrequency(2,2)=2.2e9
conv.SegmentFixedPower(2,2)=10.0
conv.SegmentPoints(2)=21
conv.SegmentRangeMode(2,1)=0
conv.SegmentMixingMode(2,1)=0
conv.SegmentCalculate 2,2
conv.SegmentState(2)=True

' Setup segment #3 from 1.50 to 1.59 GHz
' All else the same
conv.SegmentStartFrequency(3,0)=1.5e9
conv.SegmentStopFrequency(3,0)=1.59e9
conv.SegmentRangeMode(3,0)=0
conv.SegmentFixedPower(3,0)=-10.0
conv.SegmentFixedFrequency(3,2)=2.2e9
conv.SegmentFixedPower(3,2)=10.0
conv.SegmentPoints(3)=21
conv.SegmentRangeMode(3,1)=0
conv.SegmentMixingMode(3,1)=0
conv.SegmentCalculate 3,2
conv.SegmentState(3)=True

' Mixer Input to be port 1
' Mixer output to Port 2
' Mixer LO to Port 3
conv.LOName(1)="Port 3"

' Apply the scratch mixer
conv.Apply
Use an Existing Power Cal During an SMC Cal

This example shows how to use an existing Source Power Cal instead of the power cal that is performed during an SMC calibration. To run this program without modification, you need the following:

- A Mixer setup file saved on the VNA: C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Mixer\MyMixer.mxr.
- If the mixer file uses an external LO source, it must be connected and configured.
- An ECal module that covers the frequency range of the measurement.
- An SMC cal set named "SMC_CAL". This is the cal set that source power correction data will be imported from. The input and output frequency ranges of the cal set must cover the corresponding ranges used during calibration, or guided cal initialization will fail.

Error Messages

- If you attempt to import power cal data from an SMC calset that uses different ports than the ones currently in use, the message “The necessary calibration standards were not found." will appear.
- If the imported Cal Set does not cover the frequency range of the current cal, the message “Interpolation target is out of range. Cannot interpolate." will appear.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as SMC.vbs. Learn how to setup and run the macro.

See Also

SMC Type Object
ImportDataSet Method

See Other COM Example Programs

Dim App
Set App = CreateObject("AgilentPNA835x.Application")
App.Preset
Dim Meas
Set Meas = App.ActiveMeasurement
Meas.Delete
App.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","SC21"

' Other valid strings that can be specified to create a measurement with a parameter
' other than 'SC21' are: "S11", "S22", "IPwr", and "OPwr"
Set Meas = App.ActiveMeasurement

' You can perform mixer setup here or
' recall a previous mixer setup from the VNA Hard drive.
' This is how the mixer could be configured through the IMixer interface
Dim mix
Set mix = Meas ' reference to IMixer object
mix.ActiveXAxisRange = 0 ' 0 = mixINPUT (Input frequency range)

' Alternatively, recall a previous mixer setup from the VNA Hard drive
Meas.LoadFile "c:\users\public\network analyzer\documents/Mixer/MyMixer.mxr"
app.activechannel.numberofpoints = 21
Dim CalMgr
Set CalMgr = App.GetCalManager
Dim SMC
Set SMC = CalMgr.CreateCustomCal("SMC")
SMC.Initialize 1, 1
SMC.ConnectorType(1) = "APC 3.5 male"
SMC.ConnectorType(2) = "APC 3.5 female"
SMC.CalKitType(1) = "N4691-60004 ECal"
SMC.CalKitType(2) = "N4691-60004 ECal"

' Import power cal data from an existing SMC calset.
SMC.ImportDataSet "SMC_CAL","POWER_STEP"

' Omit the isolation part of the 2-port cal (default behavior).
SMC.OmitIsolation = 1

'Turn on auto orientation for the ECal (default behavior).
SMC.AutoOrient = 1
Dim steps
steps = SMC.GenerateSteps
For i = 1 To steps
    MsgBox SMC.GetStepDescription(i)
    SMC.AcquireStep i
Next
Dim calset
calset = SMC.GenerateErrorTerms
Msgbox("SMC Cal Complete!")
Create a Balanced Measurement using COM

The following program creates several Balanced measurements in separate windows, generates markers, calculates statistics, and sets limit lines and queries results.

Note: By their nature, balanced measurements are extremely sensitive to phase differences between the two RF paths that make up the balanced port, especially at higher frequencies. A good calibration (not performed in this example) is critical to achieving good balanced measurement results.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as BalancedCOM.vbs. Learn how to setup and run the macro.

' PNA application object
Dim app

' Channel 1 object
Dim chan1

' start of marker/limit testing range
Dim minTestStimulus

' end of marker/limit testing range
Dim maxTestStimulus

' Set to true if you want additional balanced measurements.
Dim AdditionalMeasurements
AdditionalMeasurements = 1

' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
' Preset the instrument
app.Preset
' Get the Channel 1 object
Set chan1 = app.Channels(1)
' Stop data taking for now.
chan1.Hold true
' Set up the start / stop frequency for Channel 1 sweep.
MHZ = 1000000
GHZ = 1000*MHZ
chan1.StartFrequency = 10  *MHZ
chan1.StopFrequency = 1  *GHZ
chan1.NumberOfPoints = 801
' Define our test frequency range
minTestStimulus = 100*MHZ
maxTestStimulus = 900*MHZ
This example uses DUT topology Bal-Bal - a DUT with a balanced input and balanced output.

Port mapping for our DUT:
- logical port 1 = physical ports 1 and 4
- logical port 2 = physical ports 2 and 3

The default is:
- logical port 1 = physical ports 1 and 2
- logical port 2 = physical ports 3 and 4

logical 1 logical 2

1 ------| DUT |------ 2 +
|        |
4 ------|_____  |------ 3 -

chan1.BalancedTopology.SetBBPorts 1, 4, 2, 3

Now we create some Bal-Bal measurements.
By creating Bal-Bal measurements ("BBAL:"..."), the channel is set to Bal-Bal topology, so it is not necessary to do this explicitly with the BalancedTopology.DUTTopology command.
We do it here just for clarity:

chan1.BalancedTopology.DUTTopology = 2
0 == SE-Bal, 1 == SE-SE-Bal, 2 == Bal-Bal

Create four windows, each showing one category of balanced measurement:
Create Forward Transmission Measurements in Bal-Bal topology on Channel 1, window 1

differential mode transmission
app.CreateMeasurement 1, "BBAL:SDD21",1,1
Set sdd21_1 = app.ActiveMeasurement

differential to common mode conversion
app.CreateMeasurement 1, "BBAL:SCD21",1,1
Set scd21_1 = app.ActiveMeasurement

common to differential mode conversion
app.CreateMeasurement 1, "BBAL:SDC21",1,1
Set sdc21_1 = app.ActiveMeasurement

common mode transmission
app.CreateMeasurement 1, "BBAL:SCC21",1,1
Set scc21_1 = app.ActiveMeasurement

Optionally create some additional measurements
If AdditionalMeasurements Then

Create (logical) Port 1 reflection measurements, channel 1, window 2
app.CreateMeasurement 1, "BBAL:SDD11",1,2 ' differential mode reflection
app.CreateMeasurement 1, "BBAL:SDC11",1,2 ' C to D mode conversion reflection
app.CreateMeasurement 1, "BBAL:SCD11",1,2 ' D to C mode conversion reflection
app.CreateMeasurement 1, "BBAL:SCC11",1,2 ' common mode reflection

' Create Reverse Transmission Measurements, channel 1, window 3
app.CreateMeasurement 1, "BBAL:SDD12",1,3 ' differential mode transmission
app.CreateMeasurement 1, "BBAL:SCD12",1,3 ' differential to common mode conversion
app.CreateMeasurement 1, "BBAL:SDC12",1,3 ' common to differential mode conversion
app.CreateMeasurement 1, "BBAL:SCC12",1,3 ' common mode transmission

' Create (logical) Port 2 reflection measurements in window 4
app.CreateMeasurement 1, "BBAL:SDD22",1,4 ' differential mode reflection
app.CreateMeasurement 1, "BBAL:SCD22",1,4 ' C to D mode conversion reflection
app.CreateMeasurement 1, "BBAL:SDC22",1,4 ' D to C mode conversion reflection
app.CreateMeasurement 1, "BBAL:SCC22",1,4 ' common mode reflection
End If

' Set up some limit lines to verify a minimum differential insertion loss
sdd21_1.LimitTest(1).BeginStimulus = minTestStimulus
sdd21_1.LimitTest(1).EndStimulus = maxTestStimulus
sdd21_1.LimitTest(1).BeginResponse = -2
sdd21_1.LimitTest(1).EndResponse = -2
sdd21_1.LimitTest(1).Type = 2 ' minimum limit
sdd21_1.LimitTest.State = 1

' Limit lines for maximum common mode to differential conversion
sdc21_1.LimitTest(1).BeginStimulus = minTestStimulus
sdc21_1.LimitTest(1).EndStimulus = maxTestStimulus
sdc21_1.LimitTest(1).BeginResponse = -20
sdc21_1.LimitTest(1).EndResponse = -20
sdc21_1.LimitTest(1).Type = 1 ' maximum limit
sdc21_1.LimitTest.State = 1

' Take a (synchronous) single sweep on channel 1
chan1.Single true

' Show differential forward transmission statistics.
sdd21_1.ShowStatistics = true

' Set up user range 1 to limit marker's search range.
chan1.UserRangeMin(0,1) = minTestStimulus
chan1.UserRangeMax(0,1) = maxTestStimulus

' Find/Show max common mode to differential conversion, and read back the frequency.
sdc21_1.MarkerState(1) = true
' Set marker 1 to use user range 1
sdc21_1.Marker(1).UserRange = 1
sdc21_1.Marker(1).SearchMax

' Find/Show max differential mode insertion loss, and read back the frequency.
sdd21_1.MarkerState(1) = true
' Set marker 1 to use user range 1
sdd21_1.Marker(1).UserRange = 1

3167
If sdd21_1.LimitTestFailed Then
End If

If sdc21_1.LimitTestFailed Then
End If
This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as <filename>.vbs. Learn how to setup and run the macro.

See Also

MultiDimensionalSweep Object

See other COM Examples

'Demonstration of multi-dimensional sweep setup.
set pna = CreateObject("AgilentPNA835x.Application","A-N5242A-10096")
CreateSAMeasurement
SetupMultiSweep

Sub CreateSAMeasurement
' Create a B measurement on channel 1
pna.Channels.RemoveChannelNumber(1)
call pna.CreateCustomMeasurementEx(1,"Spectrum Analyzer","B")
set chan = pna.ActiveChannel
chan.Hold
End Sub

Sub SetupMultiSweep
set chan = pna.ActiveChannel
set mdsweep = chan.MultiDimensionalSweep
set pc = chan.PhaseControl
set dc = chan.DCStimulus

'Turn Port 1 ON
chan.SourcePortMode(1) = naSourcePortOn
'Configure Port 1 frequency start/stop range
chan.SourcePortStartFrequency(1) = 2e9
chan.SourcePortStopFrequency(1) = 4e9
'Set Port 1 frequency domain's order to 3
mdsweep.SourcePortFrequencyOrder(1) = 3
'Enable Port 1 frequency domain in multi-dimensional sweep
mdsweep.SourcePortFrequencyState(1) = 3
'Configure Port 1 power start/stop range
chan.StartPowerEx(1) = -5
chan.StopPowerEx(1) = 5
'Set Port 1 power domain's order to 5
mdsweep.SourcePortPowerOrder(1) = 5
'Enable Port 1 power domain in multi-dimensional sweep
mdsweep.SourcePortPowerState(1) = 1
'Configure Port 1 phase start/stop range
pc.StartPhase(1) = 0
pc.StopPhase(1) = 270
'Set Port 1 phase domain's order to 4
mdsweep.SourcePortPhaseOrder(1) = 4
'Enable Port 1 phase domain in multi-dimensional sweep
mdsweep.SourcePortPhaseState(1) = 1
'Turn Port 3 ON
chan.SourcePortMode(3) = naSourcePortOn
'Configure Port 3 frequency start/stop range
chan.SourcePortStartFrequency(3) = 2e9
chan.SourcePortStopFrequency(3) = 4e9
'Set Port 3 frequency domain's order to 3
mdsweep.SourcePortFrequencyOrder(3) = 3
'Enable Port 3 frequency domain in multi-dimensional sweep
mdsweep.SourcePortFrequencyState(3) = 3
'Configure Port 3 power start/stop range
chan.StartPowerEx(3) = -5
chan.StopPowerEx(3) = 5
'Set Port 3 power domain's order to 5
mdsweep.SourcePortPowerOrder(3) = 5
'Enable Port 3 power domain in multi-dimensional sweep
mdsweep.SourcePortPowerState(3) = 1
'Configure Port 3 phase start/stop range
pc.StartPhase(3) = 0
pc.StopPhase(3) = 270
'Set Port 3 phase domain's order to 4
mdsweep.SourcePortPhaseOrder(3) = 4
'Enable Port 3 phase domain in multi-dimensional sweep
mdsweep.SourcePortPhaseState(3) = 1
'Turn AO1 ON
dc.State("AO1") = 1
'Configure AO1 start/stop range
dc.Start("AO1") = 0
dc.Stop("AO1") = 2
'Set AO1's order to 2
mdsweep.DCOrder("AO1") = 2
'Enable AO1 in multi-dimensional sweep
mdsweep.DCState("AO1") = 1

'Set dimension order 3's point count to 3
mdsweep.DimensionPointCount(3) = 3
'Set dimension order 3's repeat count to 1
mdsweep.DimensionRepeatCount(3) = 1
'Set dimension order 4's point count to 4
mdsweep.DimensionPointCount(4) = 4
'Set dimension order 4's repeat count to 1
mdsweep.DimensionRepeatCount(4) = 1
'Set dimension order 5's point count to 5
mdsweep.DimensionPointCount(5) = 5
'set dimension order 5's repeat count to 1
mdsweep.DimensionRepeatCount(5) = 1

End Sub
Create a PMAR Device and Measurement

The following program creates a new External Device: a Power Meter as Receiver, makes several power meter settings, and then creates a PMAR measurement.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as PMAR.vbs. Learn how to setup and run the macro.

See Also

ExternalDevices Collection

ExternalDevice Object

PowerSensorAsReceiver Object

PowerSensorCalFactorSegmentPMAR Object

PowerLossSegmentsPMAR_Collection

PowerLossSegmentPMAR Object

dim app
Set app = CreateObject("AgilentPNA835x.Application")
dim externalDevices
Set externalDevices = app.ExternalDevices
dim devicecount
devicecount = externalDevices.count
externalDevices.Add "NewPMAR"
dim newExternalDevice
Set newExternalDevice = externalDevices.Item("NewPMAR")
newExternalDevice.DeviceType = "Power Meter"
newExternalDevice.IOConfiguration= "GPIB0::14::INSTR"
newExternalDevice.IOEnable = true
Dim PMAR
Set PMAR = newExternalDevice.ExtendedProperties
PMAR.SensorIndex = 1
PMAR.ReadingsPerPoint = 10
Dim avr
avr = PMAR.ReadingsPerPoint
PMAR.ReadingsTolerance = 0.1
Dim tole
tole = PMAR.ReadingsTolerance
PMAR.MinimumFrequency = 100000000
PMAR.MaximumFrequency = 10000000000
PMAR.LimitFrequency = False
PMAR.referenceCalFactor = 99
Set powerCalFactorSegments = PMAR.CalFactorSegments
powerCalFactorSegments.Add 1,10
Set calpair = powerCalFactorSegments(1)
calpair.Frequency = 1e9
calpair.CalFactor = 99
Set calpair = powerCalFactorSegments(2)
calpair.Frequency = 2e9
calpair.CalFactor = 98
powerCalFactorSegments.Remove 3,8
PMAR.UsePowerLossSegments = True
Set pls = PMAR.PowerLossSegments
pls.Add 1,5
Set pl = pls(1)
pl.Loss = -1
pl.Frequency = 1e9
Set pl = pls(2)
pl.Loss = -2
pl.Frequency = 2e9
pls.Remove 3,3
newExternalDevice.active = true
'Create a PMAR trace with power meter connected to port 3
app.CreateMeasurement 1,"NewPMAR",3,1
Create a Wideband Pulsed Measurement using the PNA-X

This Visual Basic COM example shows you how to configure the PNA-X internal pulse generators and modulators to make wideband pulsed measurements in **pulse profile** mode using the PNA-X.

[Visit the VNA website](#) where you can download a free Wideband Pulsed Application that performs this measurement on the PNA-X.

[See all COM Pulsed examples](#)

```
'Create an VNA Application instance
Dim pnaApp As New AgilentPNA835x.Application

'Create a PathConfiguration instance
Dim pathConf As AgilentPNA835x.PathConfiguration

'Create an PulseGenerator instance
Dim pulseGen As AgilentPNA835x.PulseGenerator

'Create a Channel instance
Dim myChan As AgilentPNA835x.Channel

'Preset PNA-X
pnaApp.Preset

'Assign current active channel to myChan object
Set myChan = pnaApp.ActiveChannel

'Let PNA-X work in CW mode because of doing pulse profile measurement
myChan.SweepType = naCWTimeSweep

'Set CW Freq to 4 GHz
myChan.CWFrequency = 4000000000#

'Set IF Bandwidth to 5 MHz to get the best time resolution
myChan.IFBandwidth = 5000000#

'Assign current active channel path configuration to pathConf object
Set pathConf = myChan.PathConfiguration
```
'Let PNA-X source work in ALC Open Loop mode
myChan.ALCLevelingMode(1) = naALCOpenLoop

'Make the Pulse1 as modulation pulse to generate Pulsed-RF signal
pathConf.Element("PulseModDrive").Value = "Pulse1"

'Enable pulse modulation at Source1Out1 path
pathConf.Element("Src1Out1PulseModEnable").Value = "Enable"

'Assign current active channel pulse generator to pulseGen object
Set pulseGen = myChan.PulseGenerator

'Internal pulse generator has five channels,
'default the channel 0 use as internal ADC trigger signal
'Enable channel 0 of internal pulse generator as trigger signal
pulseGen.State(0) = True

'Enable channel 1 of internal pulse generator as modulation signal
pulseGen.State(1) = True

'Set pulse period to 10 us
pulseGen.Period = 0.00001 '10 us

'Set pulse width of channel 0 to 1 us
pulseGen.Width(0) = 0.000001 '1 us

'Set pulse width of channel 1 to 5 us
pulseGen.Width(1) = 0.000005 '5 us

End Sub
Create an IM Spectrum Measurement

This VBScript example creates IM Spectrum measurement based on passed parameters.

This subprogram is extracted from the macro on the VNA that produces an IM spectrum channel from the Marker function. You can see the entire program at C:\Program Files\Keysight\Network Analyzer\Applications\IMD\IMD.VBS".

This VBScript (*.vbs) program must be used as part of a program that supplies the required parameters. When complete, it can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as IMD.vbs.

Learn how to setup and run the macro.

See SweptIMD Object.

```vbs
'' SetupIMSpectrum
'' Setup an IM Spectrum (non-converter) channel based upon supplied parameters
sub SetupIMSpectrum(app, MkrPos, ToneSpacing, TonePower)
    dim objIMXChan, objIMDChan
    dim Fstart, Fstop, NumPoints, ToneFc

    set objIMXChan = objIMSChan.CustomChannelConfiguration
    set objIMDChan = objSIMDChan.CustomChannelConfiguration

    NumPoints = objSIMDChan.NumberOfPoints

    select case objIMDChan.SweepType

    case naIMDToneCWSweep
        ToneFc = objIMDChan.FrequencyCenter

    case naIMDTonePowerSweep
        Fstart = objIMDChan.TonePowerStart(0)
```

Fstop = objIMDChan.TonePowerStop(0)
TonePower = CalcMkrValue(Fstart, Fstop, MkrPos, NumPoints)
ToneFc = objIMDChan.FrequencyCenter

case naIMDToneCenterFreqSweep
  Fstart = objIMDChan.FrequencyCenterStart
  Fstop = objIMDChan.FrequencyCenterStop
  ToneFc = CalcMkrValue(Fstart, Fstop, MkrPos, NumPoints)
end Select

ToneFc = objIMDChan.FrequencyCenter
Fstart = objIMDChan.DeltaFrequencyStart
Fstop = objIMDChan.DeltaFrequencyStop
ToneSpacing = CalcMkrValue(Fstart, Fstop, MkrPos, NumPoints)

case naIMDDeltaFrequencySweep
  ToneFc = MarkerXValue
end Select

objIMXChan.FrequencyCenter = ToneFc
objIMXChan.DeltaFrequency = ToneSpacing
objIMXChan.TonePower(0) = TonePower
objIMXChan.TonePower(1) = TonePower
app.ActiveMeasurement.Trace.ReferenceValue = TonePower + 10
end Sub
Create an iTMSA Measurement

The following VB Script example shows how to create an iTMSA measurement with Power Sweep. Click each link to see a detailed description of each command.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as Testset.vbs. Learn how to setup and run the macro.

```vbs
'VNA App
dim app

'Measurement
dim meas

dim balancemeas

dim balstimulus

dim chan

set app = createobject("agilentpna835x.application")
set chan = app.activechannel

chan.SweepType = 2 ' Set the sweep type to power sweep

set meas = app.ActiveMeasurement
set balancemeas = meas.BalancedMeasurement

balancemeas.BalancedTopology.DUTTopology = 2 ' Bal-Bal topology

balancemeas.BalancedStimulus.Mode = 1 ' Turn on true mode

'The PNA-X balanced port numbers are always (0)=Bal 1; (-1)=Bal2

chan.StartPowerEx(0) = -5 ' Set the balanced port 1 start power to -5 dbm
chan.StopPowerEx(0) = 5 ' Set the balanced port 1 stop power to 5 dbm

chan.StartPowerEx(-1) = -10 ' Set the balanced port 2 start power to -5 dbm
chan.StopPowerEx(-1) = 0 ' Set the balanced port 2 stop power to 5 dbm
```
Create and Cal a Gain Compression Measurement

This VBScript example creates and calibrates a Gain Compression measurement and performs Compression analysis.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as GCA.vbs. Learn how to setup and run the macro.

See Gain Compression Object.

```vbnet
option explicit

dim CompLevel, Tolerance, StartFreq, StopFreq, NumFreqs, Scale, LinearPower

dim AcqMode, BackOff, StartPower, StopPower, NumPowers, EnableInterp, CompAlg

dim DwellTime, IFBandwidth, ShowIterations, host, app

' GCA Settings/Values
'
' Acquisition Mode:
' naSmartSweep = 0
' naSweepPowerAtEachFreq2D = 1
' naSweepFreqAtEachPower2D = 2
' '
' Compression Algorithm
' naCompressionFromLinearGain = 0
' naCompressionFromMaximumGain = 1
' naBackoffCompression = 2
' naXYCompression = 3
```
EndOfSweepOperation
naDefaultPowerSet = 0
naSetToStringPower = 1
naSetToStopPower = 2
naSetRFOff = 3

CompLevel = 1 ' 1 dB compression level
Tolerance = 0.05 ' SMART Sweep tolerance
StartFreq = 1E9
StopFreq = 9E9
NumFreqs = 201
Scale = 0.1
LinearPower = -20
BackOff = 10 ' Not used for Deviation from linear gain
StartPower = -20
StopPower = 8
NumPowers = 60 ' Not used for SMART Sweep
DwellTime = 0.0005 ' Allow some time for DUT bias/thermal effects
IFBandwidth = 1000 ' Reasonable trace noise at -20 dBm
EnableInterp = False ' Disable interpolation
AcqMode = 0 ' Smart Sweep
CompAlg = 0 ' Deviation from linear gain
ShowIterations = False ' Configure SMART to not show iteration results

dim objargs

set objargs = wscript .Arguments

if (objArgs .Count = 1) then host = objargs (0)

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'' Create and Configuration GCA Channel:
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

set app = CreateObject("Agilentpna835x.application")

call SetupGCA ( app ,
    StartFreq ,
    StopFreq ,
    NumFreqs ,
    EnableInterp ,
    Scale ,
    CompLevel ,
    LinearPower ,
    AcqMode ,
    BackOff ,
    StartPower ,
    StopPower ,
    NumPowers ,
    CompAlg ,
    DwellTime )
sub SetupGCA ( app, StartFreq, StopFreq, NumFreqs, EnableInterp, Scale, CompLevel, LinearPower, AcqMode, BackOff, StartPower, StopPower, NumPowers, CompAlg, DwellTime, IFBAndwidth, ShowIterations )

dim chan, gca

app .reset

app .CreateCustomMeasurementEx 1, "Gain Compression", "S21", 1

set chan = app .channels (1)

chan .hold 1

app .CreateCustomMeasurementEx 1, "Gain Compression", "CompIn21", 1

app .CreateCustomMeasurementEx 1, "Gain Compression", "DeltaGain21", 1

app .nawindows (1).traces (3).YScale = Scale

app .nawindows (1).traces (3).ReferenceValue = -CompLevel

set gca = chan .CustomChannelConfiguration

gca .InputLinearPowerLevel = LinearPower

gca .AcquisitionMode = AcqMode

gca .CompressionLevel = CompLevel

gca .CompressionBackoff = BackOff
gca .CompressionDeltaX = BackOff

gca .CompressionDeltaY = BackOff - CompLevel

gca .CompressionAlgorithm = CompAlg

gca .NumberOfPowerPoints = NumPowers

gca .CompressionInterpolation = EnableInterp

gca .SmartSweepSettlingTime = DwellTime

gca .SmartSweepShowIterations = ShowIterations

chan .IFBandwidth = IFBandwidth

chan .DwellTime = DwellTime

chan .StartPower = StartPower

chan .StopPower = StopPower

chan .StartFrequency = StartFreq

chan .StopFrequency = StopFreq

chan .NumberOfPoints = NumFreqs

chan .single 1

end sub

'' GCA Calibration
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Configure GCA Guided Cal for the connector types and ECal module
that will be used:
GCACal .Initialize 1, 0
GCACal .ConnectorType (1) = "APC 3.5 female"
GCACal .ConnectorType (2) = "APC 3.5 male"
GCACal .CalKitType (1) = "N4691-60004 ECal"
GCACal .CalKitType (2) = "N4691-60004 ECal"
set GCACustomCal = GCACal .CustomCalConfiguration
GCACustomCal .PowerLevel = 0
0 dBm

' Set power level for source cal to

CalSteps = GCACal .GenerateSteps
for I = 1 to CalSteps
msgBox GCACal .GetStepDescription (I )
GCACal .AcquireStep (I )
next
CalSet = GCACal .GenerateErrorTerms ' Calculate error terms and
apply CalSet to GCA Channel
chan .CalSet .Save ("GCA 2P" ) ' Save CalSet
msgBox "Done"
end sub

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'' GCA Analysis (PNA Rev A.09.00 or later)

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
sub Analysis (app)
Dim meass
Dim ana

3186


Set meass = app.Measurements ' get the measurements

Set ana = meass(1).CustomMeasurementConfiguration ' get the measurement
ana.AnalysisEnable = true ' enable the analysis mode
ana.AnalysisCWFreq = 3e9 ' set the analysis cw frequency to 3GHz

Set ana = meass(2).CustomMeasurementConfiguration
ana.AnalysisEnable = true
ana.AnalysisCWFreq = 4e9
ana.AnalysisXAxis = naPsourceAsXAxis ' set the XAxis as the source power setting

Set ana = meass(3).CustomMeasurementConfiguration
ana.AnalysisEnable = true
ana.AnalysisIsDiscreteFreq = false ' turn off the discrete frequency option
ana.AnalysisCWFreq = 4.5e9

end sub
Create and Cal a GCX Measurement

This VBScript example creates and calibrates a GCX measurement and performs Compression analysis.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as GCX.vbs. Learn how to setup and run the macro.

See Gain Compression Object and Converter Object

```vbnet
option explicit

dim CompLevel, Tolerance, StartFreq, StopFreq, LOFreq, NumFreqs, Scale, LinearPower

dim AcqMode, BackOff, StartPower, StopPower, NumPowers, EnableInterp, CompAlg

dim DwellTime, IFBandwidth, ShowIterations, host, app, parser

CompLevel = 1 ' 1 dB compression level
Tolerance = 0.05 ' SMART Sweep tolerance
StartFreq = 2.5E9
StopFreq = 2.6E9
LOFreq = 1.7E9
NumFreqs = 21
Scale = 0.1
LinearPower = -10
BackOff = 10 ' Not used for Deviation from linear gain
StartPower = -20
StopPower = 8
NumPowers = 60 ' Not used for SMART Sweep
DwellTime = 0.0005 ' Allow some time for DUT bias/thermal effects
IFBandwidth = 1000 ' Reasonable trace noise at -20 dBm
EnableInterp = False ' Disable interpolation
AcqMode = 0 ' Smart Sweep
```
CompAlg = 0 ' Deviation from linear gain
ShowIterations = False ' Configure SMART to not show iteration results

dim objargs
set objargs = wscript.Arguments
if (objArgs.Count = 1) then host = objargs(0)

'----------------------------------------------------------
' Create and Configuration GCX Channel:
'----------------------------------------------------------

set app = CreateObject("Agilentpna835x.application")
call SetupGCAX( parser,_
    StartFreq,_
    StopFreq,_
    LOFreq,_
    NumFreqs,_
    EnableInterp,_
    Scale,_
    CompLevel,_
    LinearPower,_
    AcqMode,_
    BackOff,_
    StartPower,_
    StopPower,_
    NumPowers,_
    CompAlg,_
    DwellTime,_
    IFBAndwidth,_
    ShowIterations )
call CalGCAX( parser )
sub SetupGCAX( parser, StartFreq, StopFreq, LOFreq, NumFreqs, EnableInterp, Scale, CompLevel, LinearPower, _ AcqMode, BackOff, StartPower, StopPower, NumPowers, CompAlg, DwellTime, IFBAndwidth, _ ShowIterations )

dim chan, gca

app.reset

app.CreateCustomMeasurementEx 1, "Gain Compression Converters", "SC21", 1

set chan = app.channels(1)

dim converter

set converter = chan.Converter()

chan.hold 1

app.CreateCustomMeasurementEx 1, "Gain Compression Converters", "CompIn21", 1
app.CreateCustomMeasurementEx 1, "Gain Compression Converters", "DeltaGain21", 1

app.nawindows(1).traces(3).YScale = Scale

app.nawindows(1).traces(3).ReferenceValue = -CompLevel

set gca = chan.CustomChannelConfiguration

gca.InputLinearPowerLevel = LinearPower

gca.AcquisitionMode = AcqMode

gca.CompressionLevel = CompLevel

gca.CompressionBackoff = BackOff

gca.CompressionDeltaX = BackOff

gca.CompressionDeltaY = BackOff - CompLevel

gca.CompressionAlgorithm = CompAlg

gca.NumberOfPowerPoints = NumPowers
gca.CompressionInterpolation = EnableInterp

gca.SmartSweepSettlingTime = DwellTime

gca.SmartSweepShowIterations = ShowIterations

chan.IFBandwidth = IFBandwidth

chan.DwellTime = DwellTime

chan.StartPower = StartPower

chan.StopPower = StopPower

chan.TestPortPower(1) = LinearPower

chan.StartFrequency = StartFreq

chan.StopFrequency = StopFreq

chan.NumberOfPoints = NumFreqs

' set converter properties

converter.InputRangeMode = 0 ' swept

converter.LORangeMode(1) = 1 ' fixed

converter.OutputRangeMode = 0 ' swept

converter.InputStartFrequency = StartFreq

converter.InputStopFrequency = StopFreq

converter.LOFixedFrequency(1) = LOFreq

converter.LOName(1) = "Port 3"

converter.LOPower(1) = -10

converter.Calculate 2 ' calculateOutput

chan.Single 1

end sub

'----------------------------------------------------------

' GCAX Calibration

'----------------------------------------------------------

sub CalGCAX( parser )

Dim CalMgr
Set CalMgr = app.GetCalManager
Dim SMC
Set SMC = CalMgr.CreateCustomCal("SMC")
SMC.Initialize 1, 1
SMC.Do2PortEcal = 1 'specify 0 for mechanical cal, 1 for ecal
' use Factory Characterization
SMC.ECALCharacterization(1) = 0
SMC.OmitIsolation = 1
SMC.AutoOrient = 1
' 1- forward, 2-reverse, or Both
SMC.CalibrationPort = "1"
Dim steps
steps = SMC.GenerateSteps
Dim i
For i = 1 To steps
    MsgBox SMC.GetStepDescription(i)
    SMC.AcquireStep i
Next
Dim calset
calset = SMC.GenerateErrorTerms
Msgbox("SMC Cal Complete!")
end sub
Create and Cal a Noise Figure Measurement

This example program creates a Noise Figure measurement, then calibrates the measurement.

You MUST change the ECal Identification strings (in **Blue** font).

Optional: Uncomment the following lines (in **Blue** font) to change these settings:

- Noise Receiver = Noise Receiver to Std (VNA) Receiver
- Cal Method = "Vector" to "Scalar"
- Receiver Characterization Method = "NoiseSource" to "Power Meter"

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Noise.vbs. Learn how to setup and run the macro.

**See Also**

NoiseFigure Object.

Accessing the NoiseFigure object and NoiseCal object using C#

**See other COM Examples**

```vbnet
windowNum = 1
channelNum = 1
set pna=CreateObject("AgilentPNA835x.Application")
set scpi = pna.ScpiStringParser
pna.reset
' Create noise figure measurement
set noise = pna.createcustommeasurementex(channelNum, "NoiseFigure", "NF", windowNum)
set noisechan = pna.activechannel
' Create object to access noise-specific channel attributes
set noiseConfig = pna.activechannel.CustomChannelConfiguration
```
' Create guided noise calibration object on our channel
set noisecal = pna.GetCalmanager.CreateCustomCalEx(channelNum)
set noiseCalExtension = noisecal.CustomCalConfiguration
noiseCalExtension.NoiseSourceCold  = 300

' Substitute appropriate ECal identification strings here
tunerEcal = "N4691-60004 ECal 02821"
pullEcal = "N4691-60004 ECal 02297"

' configuration
ConfigureChannel
ConfigureNoiseSettings

' perform calibration
SetupCalAttributes_Insertable
SetupNoiseSource
FinishCalibration

' ----- Support subroutines -----
'
' Configure noise channel
sub ConfigureChannel
    noisechan.startfrequency = 500e6
    noisechan.stopfrequency = 5.0e9
    noisechan.numberofpoints = 201
    noisechan.IFBandwidth = 1.0E3
end sub

' Configure noise-specific channel settings
sub ConfigureNoiseSettings
    noiseConfig.NoiseReceiver = 1 'Noise Receiver
    noiseConfig.NoiseReceiver = 0 'Std VNA Receiver
    noiseConfig.noiseaveragestate = true
    noiseConfig.NoiseAverageFactor = 40
noiseConfig.NoiseTuner = tunerEcal
noiseConfig.NoiseTunerIn = "B"
noiseConfig.NoiseTunerOut = "A"
noiseConfig.NoiseBandwidth = 8e6

end sub

sub SetupCalAttributes_Insertable

noisecal.Initialize channelNum, true

noisecal.ConnectorType( 1 ) = "APC 3.5 female"
noisecal.ConnectorType( 2 ) = "APC 3.5 male"

noisecal.CalKitType (1) = pullEcal
noisecal.CalKitType (2) = pullEcal

noiseCalExtension.NoiseSourceConnectorType = "APC 3.5 male"
noiseCalExtension.NoiseSourceCalKitType = pullEcal
noiseCalExtension.CalMethod = "Vector"
' noiseCalExtension.CalMethod = "Scalar"
noiseCalExtension.RcvCharMethod = "NoiseSource" 'Can NOT be used with Std VNA Rcvr
' noiseCalExtension.RcvCharMethod = "PowerMeter"
end sub

sub SetupNoiseSource

' specify the ENR file for the noise source

noiseCalExtension.ENRFile = "c:\users\public\network analyzer\documents/346C_MY44420454.enr"

noiseCalExtension.NoiseSourceCold = 301.1
end sub

' Build the connection list and acquire the calibration

sub FinishCalibration
steps = noisecal.GenerateSteps
for i = 1 to steps
    str = noisecal.GetStepDescription( i )
    msgbox str
    noisecal.AcquireStep i
next

guid = noisecal.GenerateErrorTerms
wscript.echo "Calibration created calset guid: ",guid
end sub

**Bonus: Accessing the NoiseFigure object and NoiseCal object using C#**

Replace `<hostname>` with the full computer name of your VNA

```csharp
Type pna = Type.GetTypeFromProgID("AgilentPNA835x.Application", "<hostname>");

    AgilentPNA835x.Application app =
    (AgilentPNA835x.Application)Activator.CreateInstance(pna);

    app.Reset();

    app.CreateCustomMeasurementEx(1, "NoiseFigure", "NF", 1);

    AgilentPNA835x.ICalManager5 calManager =
    (AgilentPNA835x.ICalManager5)app.GetCalManager();

    AgilentPNA835x.IGuidedCalibration4 guidedCal4 =
    (AgilentPNA835x.IGuidedCalibration4)

    calManager.CreateCustomCalEx(1);

    AgilentPNA835x.INoiseCal noiseCal =
    (AgilentPNA835x.INoiseCal)guidedCal4.CustomCalConfiguration;
```
Create and Cal an NFX Measurement

This program does the following:

- Setup a Noise Figure SC21 Measurement
- Calibrate Noise Figure channel
- Optional - Configure for an Embedded LO

To run this program, make the following edits, highlighted in yellow:

- Set **host** to your VNA computer name
- Set **tunerECal** and **pullECal** to your ECal model and info
- Set **ENR** to correct file name and location
- Set **connector types** for ECal, power sensor, and noise source

This program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as NFX.vbs. Learn how to setup and run the macro.

**See Also**

CreateCustomMeasEX command
NoiseFigure Object
NoiseCal Object
Converter_Object
GuidedCal Object
EmbeddedLO Object

**Learn About...**

Noise Figure on Converters
Noise Cal

**See other COM Examples**
option explicit

' NFX sweep type
' naLinearSweep = 0
' naCWTimeSweep = 2

' Converter sweep mode
' naSwept = 0
' naFixed = 1

' Embedded LO tuning mode
' Broadband and precise = 0
' Precise only = 1
' None = 2

dim app

dim chan

dim nfx

dim host

host = "MyPNA"

set app = CreateObject("Agilentpna835x.application", host)

app.reset

call SetupNFX

' optional if not doing embedded LO
' call SetupEmbeddedLO

call CalNFX

sub SetupNFX

dim tunerEcal

tunerEcal = "N4691-60003 ECal 00591"

' create NFX traces
' set channel and application objects
set chan = app.ActiveChannel
set nfx = chan.CustomChannelConfiguration
dim converter
set converter = chan.GetConverter()

' Set channel properties
chan.single 1
chan.sweeptype = 0 'naLinearSweep
chan.numberofpoints = 201
chan.IFBandwidth = 1.e3

' Set nfx properties
nfx.noiseaveragestate = true
nfx.noiseaveragefactor = 10
nfx.noisetuner = tunerECal
nfx.NoiseTunerIn = "B"
nfx.NoiseTunerOut = "A"
nfx.NoiseBandwidth = 8e6
nfx.noisegain = 0 'low

' converter properties
converter.InputRangeMode = 0 ' swept
converter.LORangeMode(1) = 1 'fixed
converter.OutputRangeMode = 0 'swept
converter.InputStartFrequency = 8.0e8
converter.InputStopFrequency = 3.0e8
converter.LOFixedFrequency(1) = 1.5825e10
converter.LOPower(1) = -10
converter.Calculate 2 'calculateOutput
converter.LOName(1) = "Port 3"
converter.Apply
chan.Single 1
end sub

sub CalNFX
' Set ecal and noise tuner
dim SparamECal
SparamECal = "N4693-60001 User 2 ECal 00012"
chan.single 1
dim calMgr
set calMgr = app.GetCalManager
dim nfxCal
set nfxCal = CalMgr.CreateCustomCalEx(1)
dim nfxCalExt
dim nfxCalExt = nfxCal.CustomCalConfiguration
nfxCalExt.ENRFile = "C:\Program Files(x86)\Keysight\Network Analyzer\Noise\346C_44420601.enr"
' setup calibration
nfxCal.Initialize 1, true
'dut connector
nfxCal.ConnectorType(1) = "APC 3.5 female"
nfxCal.ConnectorType(2) = "APC 3.5 female"
nfxCal.CalKitType(1) = SparamECal
nfxCal.CalKitType(2) = SparamECal

' power sensor connector
nfxCal.PowerCalibrationPowerLevel(1) = -20
nfxCal.PowerSensorConnectorType(2) = "APC 3.5 male"
nfxCal.PowerSensorCalkitType(2) = SparamECal

' noise source connector
nfxCalExt.NoiseSourceConnectorType = "APC 3.5 male"
nfxCalExt.NoiseSourceCalkitType = SparamECal
nfxCalExt.CalMethod = "Vector"
nfxCalExt.EnableLOPowerCal(1) = False
nfxCalExt.ForceDeEmbedENRAdapter = False
nfxCalExt. ForceDeEmbedSensorAdapter = False

' step through calsteps
dim steps
steps = nfxcal.GenerateSteps
dim i , str
for i = 1 to steps
str = nfxcal.GetStepDescription(i)
msgbox str
nfxcal.AcquireStep i
next
dim guid
guid = nfxcal.generateerrorterms
wscript.echo "Calibration created calset guid: ", guid
chan.continuous
end sub
sub SetupEmbeddedLO

    ' Set embedded LO properties

dim ELO

    set ELO = converter.ConverterEmbeddedLO

    ELO.NormalizePoint = 101

    ELO.TuningMode = 0 ' Broadband and precise

    ELO.TuningIFBW = 3.0e4

    ELO.MaxPreciseTuningIterations = 5

    ELO.PreciseTuningTolerance = 1

    ELO.TuningSweepInterval = 1

    ELO.IsOn = true

    chan.Single 1

end sub
Create and Cal a Swept IMD Measurement

This VBScript example creates IMD power and IM3 measurements, sets sweep mode to Center Frequency Sweep, and performs an IMD cal.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as IMD.vbs.

You can see the VB Script program on the VNA that produces an IM spectrum channel from the Marker function at C:\Program Files(x86)\Keysight\Network Analyzer\Applications\IMD\IMD.VBS.

Learn how to setup and run the macro.

See SweptIMD Object.

```vbs
option explicit

' declare variables

Dim SweepMode, StartDeltaFreq, StopDeltaFreq, NumFreqs, TonePower, CWFreq

Dim app, hostname

' Sweep type:

' naIMDToneCWSweep = 0
' naIMDTonePowerSweep = 1
' naIMDToneCenterFreqSweep = 2
' naIMDDeltaFrequencySweep = 3
' naIMDToneSegmentSweep = 4

' init variables

SweepMode = 3        ' Sweep DeltaF
StartDeltaFreq = 100e3
StopDeltaFreq = 1e9
NumFreqs = 201
TonePower = -7
CWFreq = 5e9
```
' get host name from commandline
dim objargs
set objargs = wscript.arguments
if(objargs.Count = 1) then hostname = objargs(0)
set app = CreateObject("Agilentpna835x.application", hostname)
call SetupIMD
call CalIMD

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
chan.single 1
end sub
sub CalIMD
  dim chan, CalMgr, IMDCal, IMDCustomCal, CalSteps, I, CalSet
  set chan = app.ActiveChannel
  set CalMgr = app.GetCalManager
  set IMDCal = CalMgr.CreateCustomCalEx(1)

  'Configure IMD GuidedCal for the connector types and ECal module that will be used
  ' Substitute appropriate connector type and ECal identification strings here
  IMDCal.Initialize 1, true 'channel number is 1
  IMDCal.ConnectorType(1) = "APC 3.5 female"
  IMDCal.ConnectorType(2) = "APC 3.5 male"
  IMDCal.CalKitType(1) = "N4693-60001 User 2 ECal 00012"
  IMDCal.CalKitType(2) = "N4693-60001 User 2 ECal 00012"

  ' IMD Custom settings
  set IMDCustomCal = IMDCal.CustomCalConfiguration

  ' Set the Power Level at the power sensor to be used in calibration
  IMDCustomCal.PowerLevel = 0

  ' Specify the connector type of the power sensor. If there is an adapter between
  ' the input port and the power sensor, specify the connector type here, and set
  ' the appropriate cal kit type for the connector so that extra calibration can be
  ' performed. To skip the calibration for the adapter, set
PowerSensorConnectorType to "Ignored"

' i.e.: IMDCustomCal.PowerSensorConnectorType = "Ignored"

IMDCustomCal.PowerSensorConnectorType = "APC 3.5 female"

IMDCustomCal.PowerSensorCalKitType = "N4693-60001 User 2 ECal 00012"

' Set the Max product to calibrate, valid values are 3, 5, 7, and 9

IMDCustomCal.MaxProduct = 3

' Set the calibration Frequencies, can choose between calibrate only at center Frequencies (0)

' or calibrate at all frequencies (1).

IMDCustomCal.CalibrationFrequencies = 1

'Include 2nd order product in calibration

IMDCustomCal.Include2ndOrderProduct = true

CalSteps = IMDCal.GenerateSteps

for I = 1 to CalSteps

    msgBox IMDCal.GetStepDescription(I)

    IMDCal.AcquireStep(I)

next

CalSet = IMDCal.GenerateErrorTerms

msgBox "IMD Cal Done"

end sub
This VB Script program illustrates ENR file management using COM commands.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as Calibrate.vbs. Learn how to setup and run the macro.

See Other COM Example Programs

```
' Sample VBS program illustrating COM commands for ENR file management.

option explicit

dim pna   ' application

dim enr   ' ENRFile object

dim scpi, hostname

set pna=CreateObject("agilentpna835x.application")

set scpi = pna.ScpiStringParser

set enr = pna.ENRFile

' Generate data to put in ENR file

Dim vdata(3)

vdata(0) = 100E6    ' first frequency point

vdata(1) = 14.532   ' first ENR value

vdata(2) = 20E9     ' second frequency point

vdata(3) = 15.731   ' second ENR value

' send data to ENRFile object

enr.PutENRData(vdata)
```
' Set noise source serial number
enr.ENRSN = "ABCD1234"

' Write ENR file to disk
enr.SaveENRFile("C:\Program Files(x86)\Keysight\Network Analyzer\Documents\sample.enr")

The contents of the file created by this program are shown below.

[Filetype ENR]

[Version 1.0]

[Serialnumber ABCD1234]

! Frequency  ENR
! Hz   dB

  100000000  14.53200
  2e+010  15.73100
Events with C++

The following code, along with the Header file, shows how to use the VNA Events.

Download the Header file 'preventcatcher.h'

```cpp
#include <atlbase.h>
#include <atlcom.h>
#include <iostream>
#include "pnaeventcatcher.h"

inline void HR(HRESULT hr)
{
    if (FAILED(hr))
        throw hr;
}

class MyEventCatcher : public CPNAEventCatcher
{

    public:
        MyEventCatcher()
```
{  
    CoInitialize(NULL);
    CComPtr<IApplication> app;
    HR(app.CoCreateInstance(CLSID_Application));
    CPNAEventCatcher::SubscribeCatcher(app);
    HR(app->AllowAllEvents());
}
~MyEventCatcher()
{
    CPNAEventCatcher::Release();
    CoUninitialize();
}
virtual void OnMeasurementEvent(long eventID, long measurementNumber) {}
virtual void OnChannelEvent(long eventID, long ch)  
{
    if (eventID == 0x68070709L) // MSG_ALL_SWEEPS_COMPLETED_AND_PROCESSED
    {
        static int i = 0;
        ++i;
        std::cout << "Sweep:" << i << std::endl;
    }
}
In a .cpp file, (just like most ATL projects) you must have a declared an instance of CComModule. This will work:

CComModule _Module;

Remember that you are now the "Server" and the VNA is the Client. That makes DCOM a bit complicated.

This code was tested in VS2005 using a wizard generated MFC MDI project.
FOM Examples

All three VBScript examples in this topic create a FOM measurement with the following attributes:

- Sweep the Source (input) from 1 GHz to 2 GHz
- Sweep the Receivers (output) from 2 GHz to 3 GHz
- You provide an LO at 1 GHz

Learn more about Frequency Offset Mode

These programs can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as FOM.vbs. Learn how to setup and run the macro.

The following example will run on any VNA model with FOM (opt S93080A). However, these commands have no provisions for internal second source. It uses commands introduced before 'enhanced FOM' was released for the A.07.10 release.

```vbnet
set app = CreateObject("Agilentpna835x.application")
set chan = app.ActiveChannel
chan.startFrequency = 1e9
chan.StopFrequency = 2e9
' set the receiver frequencies to be 2e9->3e9
chan.FrequencyOffsetFrequency = 1e9
chan.FrequencyOffsetState = 1
```

The following example can be run ONLY on a VNA with revision A.07.10 or later and has FOM (opt S93080A). It uses new FOM commands. See FOMRange object.
The following example can be run ONLY on a VNA with a second internal source, has revision A.07.10 or later, and has FOM (opt S93080A). It uses the internal 2nd source for the fixed LO frequency.

```vba
set app = CreateObject("Agilentpna835x.application")
set chan = app.ActiveChannel
chan.startFrequency = 1e9
chan.StopFrequency = 2e9
' set the receiver frequencies to be 2e9->3e9
chan.fom("Receivers").Offset = 1e9
chan.fom.State = 1
```

```vba
chan.fom("Source2").Coupled = 0
chan.fom("Source2").StartFrequency = 1e9
chan.fom("Source2").StopFrequency = 1e9
' turn off port coupling
chan.coupleports = 0
' set LO to 10 dBm
chan.TestPortPower(3) = 10
'Turn ON port 3, our LO signal on our 2 source PNA
chan.SourcePortMode(3) = 1
chan.fom.State = 1
```
Intro to Examples

Limit Line Testing with COM

This Visual Basic program:

- Turns off existing Limit Lines
- Establishes Limit Lines with the following settings:
  - Frequency range - 4 GHz to 8 GHz
  - Maximum value - (10dB)
  - Minimum value - (-30dB)
- Turns on Lines, Testing, and Sound

If using Global Pass/Fail to report limit results, trigger the VNA after configuring and enabling Limit lines.

```vbnet
Public limts As LimitTest
Set limts = meas.LimitTest
'All Off
For i = 1 To 20
    limts(i).Type = naLimitSegmentType_OFF
Next i

'Set up Limit Lines
limts(1).Type = naLimitSegmentType_Maximum
limts(1).BeginResponse = 10
limts(1).EndResponse = 10
limts(1).BeginStimulus = 4000000000#
limts(1).EndStimulus = 8000000000#
limts(2).Type = naLimitSegmentType_Minimum
limts(2).BeginResponse = -30
limts(2).EndResponse = -30
limts(2).BeginStimulus = 4000000000#
limts(2).EndStimulus = 8000000000#

'Turn on Lines, Testing, and Sound
limts.LineDisplay = 1
limts.State = 1
limts.SoundOnFail = 1
```
Modify Display Colors

This VBScript example modifies display colors, modifies trace1 colors, then saves and recalls the theme.

These programs can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as Colors.vbs. Learn how to setup and run the macro.

```vbs
function RGB(R, G, B)
    RGB = R + G*(2^8) + B*(2^16)
end Function

shell.AppActivate "PNA Series Network Analyzer"
Set app = CreateObject("AgilentPNA835X.Application")
app.preset
Set colors = app.Preferences.DisplayColors
' Uncomment the following line to modify Print colors
'    Set colors = app.Preferences.PrintColors

colors.ResetTheme( )
colors.background = RGB(64,0,64) ' purple
displaycolors.grid = RGB(0,255,128) ' greenish
colors.activeLabels = RGB(0,0,255) ' blue
colors.inactiveLabels = RGB(255,0,0) ' red
colors.failedTraces = RGB(255,128,64) ' orange
dim Trace1
Set Trace1 = colors.Trace(1)
Trace1.DataAndLimits = RGB(1,251,1) ' green
Trace1.Memory = RGB(251,1,1) ' red
Trace1.Markers = RGB(251,251,251) ' white
```
Trace1.MemoryMarkers = RGB(1,251,251)    ' green + blue

colors.StoreTheme("c:\Program Files(86)\Keysight\Network Analyzer\Colors\Theme1.colors")

colors.LoadTheme("c:\Program Files(86)\Keysight\Network Analyzer\Colors\Theme1.colors")
The following VB Script example exercises the COM commands used to control the E5091A testset.

For a description of each command, see E5091Testsets collection.

```vbnet
Sub Main()
    Set pna = CreateObject("AgilentPNA835x.Application")
    Dim testsets As E5091Testsets
    Set testsets = pna.E5091Testsets
    Dim tset1 As E5091Testset
    Set tset1 = testsets(1)
    tset1.OutputPort(1, 3) = naE5091PortR2
    tset1.ControlLines(1) = 5
    tset1.ShowProperties = True
    tset1.Enabled = True
    MsgBox tset1.ID
    MsgBox tset1.Enabled
    MsgBox tset1.ShowProperties
    ' NumberOfPorts property returns 0 when testset not connected
    MsgBox tset1.NumberOfPorts
    MsgBox tset1.OutputPort(1, 3)
    MsgBox tset1.ControlLines(1)

    Dim tset2 As E5091Testset
    Set tset2 = testsets(2)
    tset2.Enabled = True
    tset2.ShowProperties = True
    MsgBox tset2.Enabled
    MsgBox tset2.ShowProperties
End Sub
```
Errors and the SCPIStringParser Object

This C++ program uses the SCPIStringParser.Parse command to detect the failed HRESULT and interrogate the errorInfo object for more details.

// scpierrors.cpp : Defines the entry point for the console application.

#include <iostream>
#include "afx.h"
#include "atlbase.h"
#include "C:\program files(x86)\Keysight\Network Analyzer\Automation\835x.tlb" raw_interfaces_only, no_namespace, named_guids

using namespace std;

HRESULT SendScpiCommand(IScpiStringParser* parser, CComBSTR& cmd, CComBSTR& response)
{
    CComBSTR bstr;
    HRESULT hr = parser->Parse(CComBSTR(cmd), &response);
    if (FAILED(hr))
    {
        // see if this interface supports ErrInfo
        CComPtr<ISupportErrorInfo> spSupportsErrInfo;
        if (SUCCEEDED(parser->QueryInterface(&spSupportsErrInfo)))
        {
            // it does, so let's get the errorinfo object
        }
    }
}
CComPtr<IErrorInfo> spErrorInfo;
if (SUCCEEDED(GetErrorInfo(0, &spErrorInfo)))
{
    CComBSTR errStr;
    spErrorInfo->GetDescription(&errStr);
    std::cout << "ERROR: " << CString(errStr) << std::endl;
}

return hr;

int main()
{
    CoInitialize(NULL);

    CComBSTR response;
    CComPtr<IApplication> spPNA;
    CComPtr<IScpiStringParser> spSCPI;
    if (SUCCEEDED(spPNA.CoCreateInstance(CLSID_Application)))
    {
        spPNA->get_ScpiStringParser(&spSCPI);
        SendScpiCommand(spSCPI, CComBSTR("SYSTEM:PRESET"), response);
        SendScpiCommand(spSCPI, CComBSTR("CALC:PAR:CAT?"), response);
        std::cout << CString(response) << std::endl;
    }
}
SendScpiCommand(spSCPI,
CComBSTR("THIS:IS:A:SYNTAX:ERROR"), response);

CoUninitialize();

return 0;
The following VB Script example exercises the COM commands used to control the Z5623AK64 testset.

For a description of each command, see TestsetControl Object

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as Testset.vbs. Learn how to setup and run the macro.

```vbs
' Demonstrate some COM commands for external testsets.
Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")
Sub DemoTestset(na)
Dim testsets, tset1
Dim portNum
Dim chNum, address
Set testsets = na.ExternalTestsets
chNum = 1

' Load a configuration file.
' NOTE: the K64 testset is only compatible with 4-port analyzers.
address = 0
testsets.Add "Z5623AK64", address

' Get the testset object
' in the testsets collection.
Set tset1 = testsets(1)
```
'Show the selections available for each port.

For portNum = 1 To 4
MsgBox("Port " & CStr(portNum) & " catalog: " & tset1.PortCatalog(portNum))
Next

'Set port mappings on channel 1.
tset1.OutputPorts(chNum) = "5 ext R,2 int R,3 int R,6 int R"

'Set control lines.
tset1.ControlLines(chNum) = 85

'Set label.
tset1.Label(chNum) = "Some label"

'Enable external testset control. This automatically enables status bar display as well.
tset1.Enabled = True
End Sub

'The testset used in this demo is only usable on 4-port analyzers
If (pna.NumberOfPorts <> 4) Then
MsgBox("This program only runs on 4-port analyzers.")
Else
DemoTestset(pna)
End If
RF PathConfiguration Example

Note: These commands are accessible only for PNA-X models.

These Visual Basic and C# examples exercise various commands on the:

- PathConfigurationManager Object
- PathConfiguration Object
- PathElement Object

See Also

IFPathConfiguration Setup example

VB Example

```
' Create / Get the VNA application
Dim app
Set app = CreateObject("AgilentPNA835x.Application")
' Preset the instrument
app.Preset
' Get a channel interface on which to operate
Dim chan
Set chan = app.ActiveChannel
' Modify the Default configuration, and save it as "My Config"
chan.PathConfiguration = "Default"
' Set the "Combiner" element to value "Reversed"
chan.PathConfiguration.Element("Combiner").Value = "Reversed"
' Set the "Src1" element to value "High Power"
chan.PathConfiguration.Element("Src1").Value = "High Power"
' Change the description text
```
Set the instrument's path config back to the default (req. 8)

app.PathConfigurationManager.Load 1, "My Config"

C# Example

```csharp
Type pnaType =
    Type.GetTypeFromProgID("AgilentPNA835x.Application", "PNA-NAME-HERE");

AgilentPNA835x.Application pna =
    (AgilentPNA835x.Application)Activator.CreateInstance(pnaType);

AgilentPNA835x.Channel chan =
    (AgilentPNA835x.Channel)pna.ActiveChannel;

    // Preset the Instrument
    pna.Preset();

    // Modify the Default configuration, and save it as "My Config"
    chan.set_PathConfiguration("Default");

    // Set the "Combiner" element to value "Reversed"
    chan.get_PathConfiguration().get_Element("Combiner").Value = "Reversed";

    // Change the description text
    chan.get_PathConfiguration().DescriptionText = "Connect J8 to J9."

    // Store the modified configuration
    chan.get_PathConfiguration().Store("My Config");

    // Set the instrument's path config back to the default (req. 8)
```
chan.set_PathConfiguration("Default");

// Load a previously saved configuration onto channel 2
pna.PathConfigurationManager.LoadConfiguration(1, "My Config");
Create a Narrowband Pulsed Measurement using the PNA-X or N522x

The following COM example demonstrates how to create a narrowband pulsed measurement using the Pulsed Application DLL on the PNA-X.

See the example program for wideband pulsed measurements on the PNA-X.

It first gets valid configuration settings and then uses those settings to configure the VNA and internal pulsed generators.

To run this program, you need:

- PNA-X or N522x
- Pulsed Application (Option H08)

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Pulse.vbs. Learn how to setup and run the macro.

**See Also**

- Learn how to install and register the pulsed .dll on your PC
- See the ConfigEnhancedNB2 method for sending and returning parameters to the .dll.
- See the ConfigEnhancedNBIFAtten method for setting the receiver IF gain.
- See the COM IF Configuration commands used in the program.
- See the equivalent SCPI IF Configuration commands.

' Interfaces

Dim OApp As AgilentPNA835x.application
Dim OIntPG As AgilentPNA835x.PulseGenerator
Dim OPathConf As AgilentPNA835x.PathConfiguration
Dim OFilter As AgilentPNA835x.SignalProcessingModuleFour
Dim OIF As AgilentPNA835x.IFConfiguration

' Pulsed parameters
Dim DPRF As Double
Dim DBW As Double
Dim DPhysicalIF As Double
Dim DNCO As Double
Dim DCF As Double
Dim DGD As Double
Dim DGW As Double
Dim DSWG As Double
Dim DSWG As Double
Dim DSWGR As Long
Dim LStage1TapArray() As Long
Dim LStage2TapArray() As Long
Dim LStage3TapArray() As Long
Dim BFixedPRF As Boolean
Dim IIFAtten As Integer
'pulsed DLL interface
Dim OPulsed As New AgilentPNAPulsed.application
'Pulsed settings
DPRF = 5000 'Hz
DBW = 500 'Hz
BFixedPRF = True
DNCO = 0#
DCF = 0#
DGD = 0#
DGW = 0.000001
DSWGR = 0#
'Send desired pulsed parameters to the pulsed configuration DLL. The DLL will return a new set of pulse parameters to send to the VNA.
OPulsed.ConfigEnhancedNB2 DPRF, DBW, DPhysicalIF, DNCO, DCF, LStage1TapArray, LStage2TapArray, LStage3TapArray, BFixedPRF, DGD, DGW, DSWGD, DSWGW, DSWGR

'Send configuration to VNA

'Connect to the VNA application
Set OApp = CreateObject("AgilentPNA835x.Application")

'Create instance of pulse generators on active channel
Set OIntPG = OApp.ActiveChannel.PulseGenerator

'Create instance of path configuration on active channel
Set OPathConf = OApp.ActiveChannel.PathConfiguration

'Create instance of digital filter on active channel
Set OIF = OApp.ActiveChannel.IFConfiguration

'Create instance of Hana digital filter on active channel
Set OFilter = OApp.ActiveChannel.SignalProcessingModuleFour

'Set up primary pulse period for internal pulse generators
OIntPG.Period = 1 / DPRF

'Set up internal pulse generator output #1 to drive internal source modulation
OIntPG.Width(1) = 0.0001 '100us
OIntPG.Delay(1) = 0.00001 '10us
OIntPG.State(1) = True

OPathConf.Element("PulseModDrive").Value = "Pulse1"

'Set up internal pulse generator output #2 to drive internal receiver gates for a 2 port PNA-X
OIntPG.Width(2) = 0.000001 '1us
OIntPG.Delay(2) = 0.00005 '50us
OIntPG.State(2) = True

OPathConf.Element("IFGateA").Value = "Pulse2"
OPathConf.Element("IFGateB").Value = "Pulse2"
Configure VNA in pulsed mode operation

'Turn off ALC and turn on modulator control

OApp.ActiveChannel.ALCLevelingMode(1) = naALCOpenLoop 'Source 1 output #1 ALC off

OPathConf.Element("Src1Out1PulseModEnable").Value = "Enable" 'Enable Source 1 pulse modulator

'Set path and enable IF gates

OApp.ActiveChannel.IFBandwidth = DBW

OPathConf.Element("IFSigPathAll").Value = "NBF"

'Set filter stages based on pulse parameters

OIF.IFFrequency = DPhysicalIF

OIF.IFFrequencyMode = naMANUAL

OFilter.Stage1Frequency = DNCO

OFilter.Stage1Coefficients = LStage1TapArray

OFilter.Stage2Coefficients = LStage2TapArray

OFilter.Stage3FilterType = "RECT"

OFilter.Stage3Parameter("C") = LStage3TapArray(0)

OFilter.FilterMode = naMANUAL

'Set receivers to auto gain setting

OPulsed.ConfigEnhancedNBIFAtten DPRF, DGW, IIFAtten '1us pulse width

OPathConf.Element("NBFATNA").Value = IIFAtten

OPathConf.Element("NBFATNB").Value = IIFAtten

OPathConf.Element("NBFATNR1").Value = IIFAtten

OPathConf.Element("NBFATNR2").Value = IIFAtten

MsgBox "Done"
Power Range Example

The following VB Script example exercises the COM commands used to set up the Power Range feature to access data sheet specified and typical max and min power levels (in dBm). Max power refers to the maximum leveled source power.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as RxLevel.vbs. Learn how to setup and run the macro.

See Also

About Receiver Leveling

Power Range Object

See other COM Examples

Example #1

Set Application = CreateObject("AgilentPNA835x.Application", "A-N5242A-90065")

Set PowerRange = application.Capabilities.PowerRange

powerRange.Reset

powerRange.PortNumber = 1

powerRange.PathElement("Src2Out1LowBand").Value = "HiPwr"

powerRange.RangeStartFrequency = 1e9

powerRange.RangeStopFrequency = 10e9

maxPower = powerRange.RangeGetMaxPower

minPower = powerRange.RangeGetMinPower

MsgBox("Max power: " & maxPower & " dBm" & Chr(10) & _
"Min power: " & minPower & " dBm")

End Function
Example #2

```vbscript
Set Application = CreateObject("AgilentPNA835x.Application", "A-N5242A-90065")

Set PowerRange = application.Capabilities.PowerRange
powerRange.Reset
powerRange.PortNumber = 3
powerRange.PathElement("Src2Out1LowBand").Value = "HiPwr"
powerRange.DiscreteFrequencies = Array(1e9,2e9,3e9,4e9)
maxPower = powerRange.DiscreteGetMaxPowerArray
minPower = powerRange.DiscreteGetMinPowerArray

MsgBox( "Frequency: " & ArrayToString(powerRange.DiscreteFrequencies) & " Hz" & _ Chr(10) & "Max power: " & ArrayToString(maxPower) & " dBm" & Chr(10) & _ "Min power: " & ArrayToString(minPower) & " dBm" )

Function ArrayToString(Arr)
ArrayToString = "[
If isArray(Arr) Then
ArrayToString = ArrayToString & CStr(Arr(0))
For i = 1 to ubound(Arr)
ArrayToString = ArrayToString & "," & CStr(Arr(i))
Next
End If
ArrayToString = ArrayToString & "]"
End Function
```
Setup Basic Measurements

This VBScript program sets up four basic s-parameter measurements in four windows, all in a single channel. Handles are created to the measurement, channel, and window objects so that subsequent settings can be made for each.

**Note:** This is only an example. This is not necessarily the most efficient way to make basic S-parameter measurements.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Basic.vbs. Learn how to setup and run the macro.

See VNA Object Model

See CreateSParameterEx

---

```vbs
Set pna = CreateObject("AgilentPNA835x.Application")
pna.Preset

' Get a handle to the preset channel 1, S11 meas, and window(1)
set meas1=pna.ActiveMeasurement
set chan1=pna.ActiveChannel
set win1=pna.ActiveNAWindow

' Creates a new S21 measurement in New window(2)
pna.CreateSParameterEx 1,2,1,2,2
set meas2=pna.ActiveMeasurement
set win2=pna.ActiveNAWindow

' Creates a new S12 measurement in New window(3)
pna.CreateSParameterEx 1,1,2,1,3
set meas3=pna.ActiveMeasurement
set win3=pna.ActiveNAWindow
```
' Creates a new S22 measurement in New window(4)

pna.CreateSParameterEx 1,2,2,2,4

set meas4=pna.ActiveMeasurement

set win4=pna.ActiveNAWindow

'Make settings

' set Stop Frequency for channel
chan1.StopFrequency=1e9

'set Display formats
meas1.format=1 'Lin Mag
meas2.format=2 'Log Mag
meas3.format=3 'Phase
meas4.format=4 'Smith

'Show title in all windows
win1.title="Win #1"

win2.title="Win #2"

win3.title="Win #3"

win4.title="Win #4"
Setup Compression Marker

This example program does the following:

- Creates a compression marker
- Queries the Power Out and Power In values

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as CompMkr.vbs. Learn how to setup and run the macro.

See the FIFO object.

See Other COM Example Programs

Set app = CreateObject("AgilentPNA835X.Application")
set meas = app.activemasurement
'get the COM marker object
'and create marker1
set mark = meas.marker(1)
' set the compression level
mark.compressionlevel = 1.5
'make it a compression marker
'and find the compression point
mark.searchcompressionpoint
'return power out and power in
'power in
dim answer
answer = mark.compressionpin
wscript.echo("pin: " & answer)
'power out
answer = mark.compressionpout
wscript.echo("pout: " & answer)
This VBScript example creates a Noise Figure Converter measurement for a converter with an Embedded LO.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Noise.vbs. Learn how to setup and run the macro.

```vbs
option explicit

dim app
dim chan
dim host
set app = CreateObject("Agilentpna835x.application")
app.reset

' create NFX traces
app.CreateCustomMeasurementEx 1, "Noise Figure Converters", "NF", 1
app.CreateCustomMeasurementEx 1, "Noise Figure Converters", "SC21", 1

' set channel and application objects
set chan = app.ActiveChannel

dim nfx
set nfx = chan.CustomChannelConfiguration
dim converter
set converter = chan.GetConverter()
dim calMgr
set calMgr = app.GetCalManager
dim nfxCal
set nfxCal = CalMgr.CreateCustomCalEx(1)
dim nfxCalExt
set nfxCalExt = nfxCal.CustomCalConfiguration
```
dim ELO

set ELO = converter.ConverterEmbeddedLO

' Set embedded LO properties
ELO.NormalizePoint = 101
ELO.TuningMode = 0 ' Broadband and precise
ELO.TuningIFBW = 3.0e4
ELO.MaxPreciseTuningIterations = 5
ELO.PreciseTuningTolerance = 1
ELO.TuningSweepInterval = 1
ELO.IsOn = true

' The following single sweep performs the same
' function as "Find Now" on the ELO dialog
chan.Single 1
Setup FastCW and FIFO

This example program does the following:

- Setup an A/R and B/R measurement
- Turn ON point averaging
- Set external edge triggering (commented out)
- Set FIFO and Fast CW
- Write data into FIFO data buffer
- Read FIFO data buffer

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as FIFO.vbs. Learn how to setup and run the macro.

See the FIFO object.

See Other COM Example Programs

```vbs
Dim app
set app = createObject("Agilentpna835x.application")
' Setup and measure A/R and B/R
app.Reset
app.CreateMeasurement 1,"A/R1",0
app.CreateMeasurement 1,"B/R1",0
' Set IFBW to 600khz (400thousand pts/second)
app.activeChannel.IFBandwidth = 600e3
' Point Averaging Count = 10
app.activeChannel.AverageMode = 0 ' point
app.activeChannel.averagingFactor = 10
app.ActiveChannel.averaging = 1 ' turn on
```
' Edge triggering - positive edge

'app.TriggerSetup.ExternalTriggerConnectionBehavior(1) = 2 ' BNC1 = trigger
positive edge

'app.TriggerSetup.Source = 2 ' external

'app.ActiveChannel.TriggerMode = 0 ' point

'Setup FIFO and Fast CW count

app.ActiveChannel.Hold 1 ' hold - synchronous

app.FIFO.State = 1' turn on FIFO

app.FIFO.Clear

app.activechannel.sweeptype = 3 ' CW sweep

app.activechannel.fastcwpointcount = 1000000' set the point count to 1million

app.activechannel.single 1 ' synchronous single

'the single will wait until the end of sweep.

'You do not have to wait until end of sweep to start emptying FIFO.

points = app.fifo.datacount

msgbox points

'points == 2000000 ' points = 2million. Took 5 seconds to acquire

For I = 0 to 1 ' 2 iterations (2 parameters * 2 sets of 1 million)

Dim data

Data = app.fifo.data(1000000)

Next

msgbox data(0)
Setup Noise Figure Port Mapping

This program demonstrates how to change source and receive ports when measuring noise figure. It assumes that option 029 ("Fully Corrected Noise Figure") is installed.

If only option 028 ("Noise figure measurements using standard receivers") is installed, switching ports is simpler, since only one noise receiver selection is available.

This program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as NF.vbs. Learn how to setup and run the macro.

See Also

Noise Figure Object

Create and Cal a NoiseFigure Measurement

See Other COM Example Programs

```vbs
option explicit

' Noise receiver enumerations
dim naStandardReceiver, naNoiseReceiver

' standard VNA receiver
naStandardReceiver = 0

' dedicated noise receiver (option 029 only)
naNoiseReceiver = 1

dim pna, windowNum, channelNum
set pna = CreateObject("Agilentpna835x.application")

windowNum = 1
channelNum = 1

pna.Reset

' Create Noise Figure measurement
dim noise, noiseChan, noiseConfig
```
set noise = pna.CreateCustomMeasurementEx(channelNum, "Noise Figure Cold Source", "NF", windowNum)

set noiseChan = pna.ActiveChannel

' provides access to noise-specific channel properties
set noiseConfig = noiseChan.CustomChannelConfiguration

' To change from the default input/output port settings of
' source port = VNA1, receive port = VNA2,
' you must first change the noise receiver,
' then select the desired ports.
dim srcPort, rcvPort

' set port mapping to source port = VNA3, receive port = VNA4
srcPort = 3
rcvPort = 4

' use VNA receiver for noise measurements
noiseConfig.NoiseReceiver = naStandardReceiver

noiseConfig.SetPortMap srcPort, rcvPort

' To revert back to using the noise receiver, the source
' and receive ports must be set to their default values
' BEFORE switching to the noise receiver.
' Otherwise, a COM exception will be thrown.
' restore defaults: source=VNA1, receiver=VNA2
noiseConfig.SetPortMap 1,2

' use dedicated noise receiver for noise measurements
noiseConfig.NoiseReceiver = naNoiseReceiver
Setup Phase Control

The following VB Script example exercises the COM commands used to setup and display Phase Sweep measurements.

See Also

About Phase Control

PhaseControl Object

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as RxLevel.vbs. Learn how to setup and run the macro.

```vbs
'Assume port 1 is connected to port 3
Set pna = CreateObject("AgilentPNA835x.Application")
pna.Preset
Set chan = pna.ActiveChannel
chanNum = chan.ChannelNumber

'Create 3 traces: S33, R3/C(amp),R3/C(phase)
pna.CreateMeasurement 1,"S33",3
Set meas1 = pna.ActiveMeasurement
meas1.Format = 4   'Smithchart format
pna.CreateMeasurement 1,"R3/C",3 'Log format
Set meas2 = pna.ActiveMeasurement
meas2.Format = 1   'Phase format
Set meas = pna.ActiveMeasurement
meas.Format = 2   'Phase format

'turn on 3 and 1
```
chan.SourcePortMode(1) = 1
chan.SourcePortMode(3) = 1
chan.SweepType = 5 'Phase sweep
Set phase = chan.PhaseControl
'set port3's control parameter to R3/C
phase.PhaseParameter(3) = "R3/C"
'notice the reference port should not included in the parameter
phase.PhaseReferencePort(3) = 1
'Set port3 to PAR mode
phase.PhaseControlMode(3) = 1 'PhaseControlParameter mode
phase.FixedRatioedPower(3) = 3
phase.StartPhase(3) = 0
phase.StopPhase(3) = 180
Setup PNOP and PSAT Marker Search

This example program does the following:

- Sets up measurement for either PNOP or PSAT marker search
- Sets parameters for search
- Reads a parameter for each

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as SearchMkr.vbs. Learn how to setup and run the macro.

```
Set app = CreateObject("AgilentPNA835X.Application")
app.Preset
set meas = app.activemeasurement
'veView Power Out vs Power In
meas.ChangeParameter "B",1
'veperform power sweep
set chan = app.ActiveChannel
chan.SweepType = 2
chan.StartPower = -5
chan.StopPower = 0
'-------------------
'Choose marker search
resp=Msgbox ("PNOP (yes) or PSAT (no)" , 4, "PNA Marker Search Demo")
if resp=6 then
    PNOP1()
Else
```

See Other COM Example Programs
Sub PSAT1()
    set psat = meas.PSaturation
    psat.PMaxBackOff = .3
    psat.SearchPowerSaturation
    'Read PSAT Parameter
    dim answer
    answer = psat.GainSaturation
    wscript.echo("Gain Sat: ", answer)
End Sub

Sub PNOP1()
    set pnop = meas.PNOP
    pnop.BackOff = 2
    pnop.PinOffset = 1
    pnop.SearchPowerNormalOperatingPoint
    'Read PNOP Parameter
    dim answer
    answer = pnop.Gain
    wscript.echo("PNOP Gain: ", answer)
End Sub
Setup Receiver Leveling

The following VB Script example exercises the COM commands used to setup Receiver Leveling.

See Also

About Receiver Leveling

RxLevelingConfiguration Object

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as RxLevel.vbs. Learn how to setup and run the macro.

```vbs
' Demonstrate some COM commands for Receiver Leveling.
Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")
Dim chan
Set chan = pna.ActiveChannel
Dim RxLevel
Set RxLevel = chan.GetRxLevelingConfiguration
Dim srcPort
srcPort = 1
pna.Preset
RxLevel.ReferenceReceiver(srcPort) = "R1"
RxLevel.Tolerance(srcPort) = 0.02
RxLevel.IterationNumber(srcPort) = 10
RxLevel.FastMode(srcPort) = True
RxLevel.LevelingIFBW(srcPort) = 100
RxLevel.PowerOffset(srcPort) = 0
RxLevel.PowerMax(srcPort) = 20
```
RxLevel.PowerMin(srcPort)= -50
RxLevel.SafeMode(srcPort)= True
RxLevel.State(srcPort)= True
This VBScript program shows how to send commands that allow you to view specific 'custom' windows, and sweep specific channels, during a UI (Cal Wizard) or remote calibration.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as CalWindow.vbs. Learn how to setup and run the macro.

These commands are used to show and sweep specific windows and channels:

- AllowChannelToSweepDuringCalAcquisition Method
- DisplayNAWindowDuringCalAcquisition Method
- DisplayOnlyCalWindowDuringCalAcquisition Method
- SweepOnlyCalChannelDuringCalAcquisition Method

The following command sweeps the Cal Windows before remote acquisition:

- SetupMeasurementsForStep Method

See Other COM Example Programs

```vbs
Set pna = CreateObject("AgilentPNA835x.Application")
pna.Preset

' get a handle to the preset channel 1 so that we can later cal it
set meas=pna.ActiveMeasurement

' Creates a new S21 measurement in channel 2 and New window(2)
' this will be the channel and window to show during cal
pna.CreateSParameterEx 2,2,1,2,2

Set calMgr = pna.GetCalManager
Set guidedCal = calMgr.GuidedCalibration

' show window 2 during cal
```
callMgr.DisplayNAWindowDuringCalAcquisition 2, True
' sweep channel 2 during calibration of chan 1

callMgr-AllowChannelToSweepDuringCalAcquisition 1, 2, True
' make Channel1 the active channel
' activating the measurement also activates the channel

meas.Activate

guidedCal.Initialize 1, True
' Do 2-port cal
' Select the connectors

guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 male"
' Select the Cal Kit for each port being calibrated.
guidedCal.CalKitType(1) = "85052D"
guidedCal.CalKitType(2) = "85052D"
' Initiate the calibration and query the number of steps

numSteps = guidedCal.GenerateSteps
' Measure the standards, compute and apply the cal

value = MsgBox("Number of steps is " + CStr(numSteps))
' Measure the standards

For i = 1 to NumSteps
    step = "Step " + CStr(i) + " of " + CStr(numSteps)
    strPrompt = guidedCal.GetStepDescription(i)
    ' Sweep the Cal window prior to standard acquisition

guidedCal.SetupMeasurementsForStep i
    ' prompt to connect standard

    value = MsgBox(strPrompt, vbOKOnly, step)
    ' measure standard

guidedCal.AcquireStep i
Next

' Conclude the calibration

guidedCal.GenerateErrorTerms

MsgBox ("Cal is complete!")

' clear the Cal window and channel flags

calMgr.DisplayOnlyCalWindowDuringCalAcquisition

calMgr.SweepOnlyCalChannelDuringCalAcquisition
Spectrum Analyzer

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as SA.vbs. Learn how to setup and run the macro.

See Also

Spectrum Analyzer Object

See other COM Examples

' Demonstration of basic Spectrum Analyzer measurement setup using COM.
set pna=CreateObject("AgilentVNA835x.Application","hostname")
set channel = Nothing
set sachannel = CreateSAMeasurement(pna)
SetupLinearSweep(sachannel)
ConfigureAdvancedSettings(sachannel)
' Create a Spectrum Analyzer measurement.
' Return the custom SA channel object.
Function CreateSAMeasurement(pna)
' Create a B measurement on channel 1
pna.Channels.RemoveChannelNumber(1)
call pna.CreateCustomMeasurementEx(1,"Spectrum Analyzer","B")
set channel = pna.Channels(1)
set sachannel = channel.CustomChannelConfiguration
set CreateSAMeasurement = sachannel
' Set frequency range
channel.centerFrequency = 3.0E9
channel.FrequencySpan = 2.0E9

' Center frequency step size

' Set to Auto mode with CenterFrequencyStepSizeMode = naMANUAL|naAUTO

channel.CenterFrequencyStepSize = 20E6

' RBW filter shape

' Choices are:
' 0 - naNoWindow
' 1 - naWindowFlatTop
' 2 - naWindowGaussian
' 3 - naWindowBlackman
' 4 - naWindowKaiser

sachannel.BandwidthShape = 4 ' naWindowKaiser

' Detector type

' Choices are:
' 0 - naDTAverage
' 1 - naDTSample
' 2 - naDTPeak
' 3 - naDTNormal
' 4 - naDTNegPeak
' 5 - naDTPeakSample
' 6 - naDTPeakAverage

sachannel.DetectorFunction = 2 ' naDTPeak

' Video averaging type

' Choices are:
' 0 - naPower
' 1 - naLog
' 2 - naVoltage 
' 3 - naVoltageMax 
' 4 - naVoltageMin 

sachannel.VideoAveragingType = 3 ' naVoltageMax 
avgcount = sachannel.VideoAveragingCount 
' ADC Filter 
sachannel.ADCFilter = 38E6 
' RBW and VBW values 
sachannel.ResolutionBW = 100E3 
sachannel.VideoBW = 10E3 
' RBW/VBW and Span/RBW ratios 
sachannel.ResolutionBWVideoBWRatio = 1.23 
sachannel.SpanResolutionBWRatio = 134 
End Function 
' Configure a Spectrum Analyzer measurement for Linear sweep mode on Port 1. 
Sub SetupLinearSweep(sa) 
portnum = 1 
portlabel = "Port " & CStr(portnum) 
' Turn Port 1 ON 
channel.SourcePortMode(portnum) = 1 ' naSourcePortOn 
' Set Port 1 sweep type to Linear 
' Choices are: 
' 0 - naLinearSweep 
' 3 - naCWTimeSweep 
sa.SourceSweepType(portlabel) = 0 ' naLinearSweep
' Set start and stop frequencies
sa.SourceStartFrequency(portlabel) = 2E9
sa.SourceStopFrequency(portlabel) = 4E9

' Set 'Source Number of Steps'. This is the number of frequencies
to use between start and stop (inclusive).
' This setting is channel-wide.
sa.SourcePointCount = 5

' Set 'SA Sweeps per Source Steps'. This is the number of sweeps
to take at each measurement frequency.
' This setting is also channel-wide.
sa.SourceRepeatCount = 2
End Sub

' Configure a few of the Advanced Settings for SA.
Sub ConfigureAdvancedSettings(sa)
' Set the 'Image Reject' selection.
' Choices are:
'   0 - naIRNoneHigh
'   1 - naIRNoneLow
'   2 - naIRMin
'   3 - naIRNormal
'   4 - naIBetter
'   5 - naIRMax
sa.ImageRejectMethod = 5 ' naIRMax

' Enable display of ImageReject traces.
sa.EnableImageRejectTraces = 1 ' naON

' Enable point mode.
' This forces the number of display points to match the FFT point
count.

sa.EnableDetectorBypass = 1 ' naON

End Sub
This Visual Basic program shows how to monitor the end of sweep. The program will set sweep time to various amounts and BEEPs when sweep is completed. This method allows other processes to continue while waiting for end-of-sweep. This program stops after 10 loops.

**Note:** To avoid Permission Denied problems, this should be run on the VNA and not a PC. To run it from a PC both units must be "trusted" and on the same domain/workgroup.

```vba
Option Explicit
Dim na As AgilentPNA835x.Application
Dim WithEvents naEvnt As AgilentPNA835x.Application
Dim ch As AgilentPNA835x.Channel
Dim sweepComplete As Boolean

Private Sub Form_Load()
    Dim N As Integer
    Set na = CreateObject("AgilentPNA835x.application")
    na.preset
    Set ch = na.ActiveChannel
    na.DisallowAllEvents ' Turn off all events
    Set naEvnt = na ' Enable event interrupts
    Do
        N = N + 1 ' Loop counter
        ch.sweepTime = 1 + (Rnd * 9) ' Set random sweep-time from 1-10 sec
        sweepComplete = False ' Trigger sweep
        ch.Single False ' Trigger sweep
        naEvnt.AllowEventCategory naEventCategory_CHANNEL, True ' Enable Channel event
        DoEvents ' Allows other processes to continue
        Loop Until sweepComplete = True
        naEvnt.AllowEventCategory naEventCategory_CHANNEL, False ' Disable event until ready for next one
        Beep ' Do end-of-sweep processing here;
    Loop Until N > 10
End
End Sub

Private Sub naEvnt_OnChannelEvent(ByVal eventID As Variant, ByVal chNumber As Variant)
    ' In this example we don't care about the channel info
```
If eventID = naEventID_CHANNEL_TRIGGER_COMPLETE Then sweepComplete = True
End Sub
Uncertainty on Power Meter

The following program is an example of setting up power uncertainty on a power meter.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as a <filename>.vbs file. Learn how to setup and run the macro.

See Also

ExternalDevices Collection

ExternalDevice Object

PowerSensor Object

PowerSensorAsReceiver Object

PowerSensorCalFactorSegmentPMAR Object

PowerLossSegmentsPMAR_Collection

PowerLossSegmentPMAR Object

User Defined Power Meter Uncertainty File

Power Meter Uncertainty dialog description

Application Note (5988-9215EN)

'  
'  Keysight Technologies 2018 
'  
'  Uncertainty on power meter COM example  
'  
'  This script execute and print the result of all the 
'  commands related to the uncertainty on power meter  
'  by using COM  
'  
' .................................
' Create the Application object

Dim app

Set app = CreateObject("AgilentPNA835x.Application")

' Get the external devices

dim externalDevices

Set externalDevices = app.ExternalDevices

' Get the specific device in this example is Device2

dim myExternalDevice

Set myExternalDevice = externalDevices.Item("Device2")

' print the selected external device type

msgbox("External Device type:" & myExternalDevice.DeviceType)

' Get the External device extended properties specific

Dim PwrasRec

Set PwrasRec = myExternalDevice.ExtendedProperties

' Get the uncertainty object

dim PowerSensorUncertainty

Set PowerSensorUncertainty = PwrasRec.PowerSensorUncertainty

' Get the list of all the model available for uncertainty

dim catalog

catalog = PowerSensorUncertainty.UncertaintyModelCatalog
' Print the first element of the catalog
'
msgbox("First Element of Power Meter with uncertainty catalog: ",&catalog(1))
'
' Get the selected uncertainty model
Dim pwrmodel
pwrmodel=PowerSensorUncertainty.UncertaintyModel
' print it
msgbox("Selected Power Meter Model: ",&pwrmodel)
'
' Get the custom uncertainty file
' If the model is not a CustomFile this will return Undefined as file name
Dim pwruncfile
pwruncfile=PowerSensorUncertainty.UncertaintyFile
' print it
msgbox("Selected Power Meter Custom File: ",&pwruncfile)
'
' Read the pwrmtr uncertainty value for a specific frequency and pwrlevel
' in this example 10GHz and 0.0 dBm
Dim PwrMtrReadingUncertianty
PwrMtrReadingUncertianty=PowerSensorUncertainty.PowerMtrReadingUncertainty(10e9,0.0)
msgbox("10GHz and 0.0dBm nominal value Power Meter reading uncertainty variance: ",&PwrMtrReadingUncertianty&" [W^2]")
'
' Read the power value which offer the best accuracy for the power meter
Dim PowerForBestAcc
PowerForBestAcc=PowerSensorUncertainty.PowerForBestAccuracy
msgbox("Power Meter power for best accuracy: "]PowerForBestAcc" [dBm]")
Upload a Segment Table using C++

This example program shows the Variant method for uploading a segment sweep to the VNA using the `SetAllSegments` method.

```cpp
#include "stdafx.h"
#include <stdio.h>
#include "atlbase.h"
#include "objbase.h"

// import the VNA type library
#include "C:\Program Files(x86)\Common Files\Keysight\PNA\835x.tlb" no_namespace, named_guids

int _tmain(int argc, _TCHAR* argv[])
{
    // interface pointers to retrieve COM interfaces
    HRESULT hr;
    IUnknown* pUnk = 0;
    IApplication* pNA = 0;
    IChannel* pChan = 0;
    IMeasurement* pMeas = 0;
    IMeasurement5* pMeas5 = 0;
    IArrayTransfer* pTrans = 0;
    ISegments* pSeg = 0;
    ISegments2* pSeg2 = 0;

    // code...
}
```
//Variables for X and Y data read portion
SAFEARRAY* sArray;
_variant_t vXVals;
double HUGEP* xVals;
float* pScalarData;

//Variables for Segment portion
double Fstart, Fstop, SegWidth;
long i[2];
int num_points = 11;
SAFEARRAY* pSA;
VARIANT vSeg;
VARIANT v;

int NUM_SEGS = 10;
int SEG_SIZE = 7;

//Create SafeArray to hold the segment data
SAFEARRAYBOUND aDim[2]; //This must be 2 the VNA expects to see a 2 dimensional array
aDim[0].lLbound = 0;
aDim[0].cElements = SEG_SIZE; //This will be set to 7 unless port power is uncoupled
aDim[1].lLbound = 0;
aDim[1].cElements = NUM_SEGS;

pSA= SafeArrayCreate(VT_VARIANT,2,aDim); //The cDim parameter must be set to 2 as the VNA expects a 2D array
// Init Variant to set values in Safearray

VariantInit(&vSeg);

Fstart=10e6;
Fstop=3e9;
SegWidth=(Fstop-Fstart)/NUM_SEGS;

// Loop to write segment data
for(int j=0; j<NUM_SEGS; ++j)
{
    i[1]=j; // Set Segment #

    // Segment Definition
    i[0] = 0;
    vSeg.vt = VT_BOOL; // First parameter is Boolean
    vSeg.boolVal = VARIANT_TRUE; // Segment State
    SafeArrayPutElement(pSA, i, &vSeg);
    i[0] += 1;
    vSeg.vt = VT_I4; // Second parameter is an integer
    vSeg.intVal = num_points; // Number of Points
    SafeArrayPutElement(pSA, i, &vSeg);
    i[0] += 1;
    vSeg.vt = VT_R8; // Remaining parameters are of type double
    vSeg.dblVal = Fstart+j*SegWidth; // Start Frequency
    SafeArrayPutElement(pSA, i, &vSeg);
    i[0] += 1;
    vSeg.dblVal=vSeg.dblVal+SegWidth; // Stop Frequency
SafeArrayPutElement(pSA, i, &vSeg);
  i[0] += 1;
vSeg.dblVal = 1.0e3; //IF Bandwidth
SafeArrayPutElement(pSA, i, &vSeg);
  i[0] += 1;
vSeg.dblVal = 0.0; //Dwell time
SafeArrayPutElement(pSA, i, &vSeg);
  i[0] += 1;
vSeg.dblVal = -5.0; //Power
SafeArrayPutElement(pSA, i, &vSeg);
}

//vSeg no longer needed, clean up
VariantClear(&vSeg);

//Declare Variant to use with Segment data
VariantInit(&v);
v.vt = VT_ARRAY|VT_VARIANT;
v.parray = pSA; //write safearray to variant

// Initialize the COM subsystem
CoInitialize(NULL);
CoInitializeSecurity(NULL, //security descriptor
  -1, // authn svc entries
  NULL, // authn svcs
  NULL, // reserved
  RPC_C_AUTHN_LEVEL_NONE,
RPC_C_IMP_LEVEL_IMPERSONATE,
NULL, // authn info
0, // capabilities
NULL); // reserved

// Create an instance of the network analyzer
// Request the NA's IUnknown interface
hr = CoCreateInstance(CLSID_Application, 0, CLSCTX_ALL, IID_IUnknown, (void**) &pUnk);
if (!FAILED(hr))
{
    // QueryInterface for the INetworkAnalyzer interface of the NetworkAnalyzer object
    hr = pUnk->QueryInterface(IID_IApplication, (void**) &pNA);

    if (!FAILED(hr))
    {
        // Reset the analyzer to instrument preset
        pNA->Reset();

        // Create S11 measurement
        pNA->CreateSParameter(1, 1, 1, 1);

        // Set pChan variable to point to the active channel
        pNA->get_ActiveChannel(&pChan);

        // Show Segment table
pNA->NAWindows->Item(1)->ShowTable((NATableType)2);

//Get handle to ISegments Interface
pChan->get_Segments(&pSeg);

//Get handle to ISegments2 Interface
hr = pSeg->QueryInterface(IID_ISegments2, (void**)&pSeg2);

//Set Segment Sweep Options
pSeg2->IFBandwidthOption = VARIANT_TRUE;
pSeg2->SourcePowerOption = VARIANT_TRUE;

//Push segments to VNA
pSeg2->SetAllSegments(v);

//Set Sweep Type to Segment Sweep
pChan->SweepType = naSegmentSweep;

if (pChan)
{
    // Set pMeas variable to point to the active measurement
    pNA->get_ActiveMeasurement(&pMeas);

    if(pMeas)
    {
        // Setup the channel for a single trigger
        pChan->Hold(true);
    }
}
pNA->TriggerSignal = naTriggerManual;
pChan->TriggerMode = naTriggerModeMeasurement;

// Make the VNA application visible
pNA->put_Visible(true);

// Send a manual trigger to initiate a single sweep
pChan->Single(true);

// QueryInterface for the IArrayTransfer interface of the
NetworkAnalyzer object
hr = pMeas->QueryInterface(IID_IArrayTransfer, (void**)&pTrans);

// Get handle for IMeasurement5 interface
hr = pMeas->QueryInterface(IID_IMeasurement5, (void**)&pMeas5);

if (!FAILED(hr))
{
    int val = num_points*NUM_SEGS;

    // Store the data in the "result" variable
    pScalarData = new float[val];
    xVals = new double[val*2];

    // Get X axis values
    vXVals = pMeas5->GetXAxisValues();
// Convert _variant_t array to a SAFEARRAY

sArray = vXVals.parray;

// Convert data from SAFEARRAY to double array. Each SAFEARRAY value
// is 16 bytes so it takes up 2 floats so the xVals size is double
// the number of points. This also means that every other data point
// in the resulting array can be discarded.

hr = SafeArrayAccessData(sArray, (void HUGEP**)&xVals);

// Get Measurement Values

pTrans->getScalar(naRawData, naDataFormat_LogMag, (long *)&val, pScalarData);

// Display the result

printf("S11(dB) - Visual C++ COM Example for PNA operating in segment sweep mode/n/n");

for (int j = 0; j < val; j++)
{
    // Write value...the xVals array is offset by 1 in each data point since
    // the return data is 16 bytes and each double is 8.

    printf("%.3lf GHz, %.4f/n", xVals[2*j+1]/1e9,

    pScalarData[j]);
}

}
}
else
{
    printf("Programmed failed to connect to the PNA.");
}
}
CoUninitialize();

system("PAUSE");
return 0;
Using C#

The following are common C# examples:

**Connecting to a specific VNA via DCOM:**

```csharp
public AgilentPNA835x.Application Connect(string hostname)
{
    AgilentPNA835x.Application pna = null;
    try
    {
        Type t = Type.GetTypeFromProgID("AgilentPNA835x.Application", hostname, true);
        pna = (AgilentPNA835x.Application) Activator.CreateInstance(t);
    }
    catch (Exception e)
    {
        HandleExceptions(e);
    }
    return pna;
}
```

**Using the GetData Interface**
AgilentPNA835x.IMeasurement meas = app.ActiveMeasurement;

    object[] dataArrayAsObj;

    dataArrayAsObj = (object[])(AgilentPNA835x.NADataStore.naMeasResult, AgilentPNA835x.NADataFormat.naDataFormat_LogMag);

    float[] dataArray = new float[dataArrayAsObj.Length];
    for (int j = 0; j < dataArrayAsObj.Length; j++)
    {
        dataArray[j] = (float)dataArrayAsObj[j];
    }

2-dimensional GetData

AgilentPNA835x.IMeasurement meas = app.ActiveMeasurement;

    app.ActiveChannel.Single(true);

    object[,] dataArrayAsObj;

    dataArrayAsObj = (object[,])(AgilentPNA835x.NADataStore.naRawData, AgilentPNA835x.NADataFormat.naDataFormat_Smith);

    float[,] dataArray = new float[dataArrayAsObj.Length, 2];
    for (int j = 0; j < dataArrayAsObj.Length; j++)
    {
        dataArray[j,0] = (float)dataArrayAsObj[j,0];
        dataArray[j,1] = (float)dataArrayAsObj[j,1];
    }

Other C# / .NET Topics

Perform a Guided Cal with CSharp

Getting a handle to the Noise Figure Cal object.

Using .NET
Before developing or running a COM program, you should first establish communication between your PC and the analyzer. This process is referred to as gaining **Access** to the analyzer. You should then register the VNA type library on your PC.

**DCOM** (Distributed Component Object Model) refers to accessing the VNA from a remote PC.

**COM** refers to accessing the VNA application from the analyzer PC.

- **Access Concepts**
- **Access Procedures**
- **Register the VNA Type Library on Your PC**
- **Problems?**

**Note:** After performing a **Firmware Upgrade** you must copy the new type library to your development PC to get access to new COM commands. See **Register the analyzer on your PC**.

### Other Topics about COM Concepts

For detailed information on this subject, see [http://na.support.keysight.com/pna/DCOMSecurity.html](http://na.support.keysight.com/pna/DCOMSecurity.html)

**Note:** 64-bit compiler option is supported.

#### Access Concepts

VNAs are shipped from the factory such that **Everyone** has permission to launch and access the VNA application via COM/DCOM. The term **Everyone** refers to a different range of users depending on whether the VNA is a member of a **Domain** or **Workgroup** (it must be one or the other; not both). By default, the VNA is configured as members of a workgroup. Therefore, **Everyone** includes only those users who have been given logon accounts on the VNA.

**Workgroup**

A workgroup is established by the **VNA administrator** declaring the workgroup name and declaring the VNA as a member of the workgroup. A workgroup does not require a network administrator to create it or control membership.
**Everyone** includes only those users who have been given logon accounts on the VNA.

By default, the VNA is configured as members of a workgroup named WORKGROUP.

**Note:** To setup a logon account for a new user, see [Additional Users](#).
The easiest method of gaining DCOM access, is to make the user's account name and password on the VNA to EXACTLY match their PC logon account name and password.

---

**Domain**

A domain is typically a large organizational group of computers. Network administrators maintain the domain and control which machines have membership in it.

**Everyone** includes those people who have membership in the domain. In addition, those with logon accounts can also access the analyzer.

---

**Summary**

- A **Workgroup** requires no maintenance, but allows DCOM access to only those users with a log-on account for the VNA.

- A **Domain** requires an administrator, but all members of the domain and those with logons to the analyzer are allowed DCOM access to the VNA.

The following section "Access Procedures" provides a tighter level of security allowing only selected (not **Everyone**) domain and workgroup users DCOM **Access** and **Launch** capability of the VNA.

---

**Access Procedures**

Perform this procedure for the following reasons:

- To allow only selected users (not everyone) remote Access and remote Launch capability to the VNA. Launch capability is starting the VNA application if it is not already open.

- To verify that you have DCOM access to the analyzer.

**Note:** Before doing this procedure, you must first have a logon account on the VNA. See [VNA User Accounts](#).

The following procedure grants specific users DCOM access and launch capability of the VNA application:
To perform this procedure, you must first **minimize the VNA application**.

**How do I know which Operating System I have?**

<table>
<thead>
<tr>
<th><strong>Windows 7</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>On the VNA, click the Windows <strong>Start</strong> button</strong></td>
</tr>
<tr>
<td><strong>In the <em>Search programs and files</em> text field, type <strong>cmd</strong> then press <strong>Enter</strong></strong></td>
</tr>
<tr>
<td><strong>In the cmd: window, type <strong>dcomcnfg</strong></strong></td>
</tr>
<tr>
<td><strong>Open the following folder sequence:</strong></td>
</tr>
<tr>
<td>Component Services</td>
</tr>
<tr>
<td>Computers</td>
</tr>
<tr>
<td>My Computer</td>
</tr>
<tr>
<td>DCOM Config</td>
</tr>
<tr>
<td>Right click <strong>Keysight PNA Series</strong></td>
</tr>
<tr>
<td>Click <strong>Properties</strong></td>
</tr>
<tr>
<td>Click the <strong>Security</strong> tab</td>
</tr>
<tr>
<td>Under <strong>Access Permissions</strong>, click <strong>Customize</strong>, then click <strong>Edit</strong></td>
</tr>
<tr>
<td>Select <strong>Everyone</strong>, then click <strong>Remove</strong></td>
</tr>
<tr>
<td>Click <strong>Add</strong></td>
</tr>
<tr>
<td>Type a group name or user account name</td>
</tr>
<tr>
<td>Click <strong>OK</strong></td>
</tr>
</tbody>
</table>

**Under **Launch Permissions**, click **Customize**, then click **Edit**** |
| Select **Everyone**, then click **Remove** |
| Click **Add** |
| Type a group name or user account name |
| Click **OK** |
On the VNA, click the **Type here to search** icon

In the **Type here to search** text field, type `cmd` then press **Enter**

In the **cmd**: window, type `dcomcnfg`

Open the following folder sequence:

- Component Services
- Computers
- My Computer
- DCOM Config

Right click **Keysight PNA Series**

Click **Properties**

Click the **Security** tab

Under **Access Permissions**, click **Customize**, then click **Edit**

Select **Everyone**, then click **Remove**

Click **Add**

Type a group name or user account name

Click **OK**

Under **Launch and Activation Permissions**, click **Customize**, then click **Edit**

Select **Everyone**, then click **Remove**

Click **Add**

Type a group name or user account name

Click **OK**

---

**Register the VNA Type Library on Your PC**

The type library contains the VNA object model. On your PC, there is a Registry file that keeps track of
where object models are located. Therefore, you must register the type library on the PC that will be used to develop code and run the program. It is much more efficient to have the type library registered at design time (BEFORE running your COM program).

Do the following two items before proceeding:

1. Connect your PC and the VNA to LAN.
2. Either map a drive to the analyzer or copy the type library files on a floppy disk or other media. See Drive Mapping.

**Note:** To register the type library on your PC, you must be logged on as an administrator of your PC. Learn about User Accounts.

This procedure will do the following:

- Register the Network Analyzer application on your PC.
- Copy and register the proxystub (835xps.DLL) onto the PC.
- Copy and register the VNA type library (835x.tlb) onto the PC.
- Copy and register the FCA type library (fca.tlb) onto the PC.

1. Using Windows Explorer on your PC, find the Analyzer's C: drive. The drive will not be named "C:" on your PC, but a letter you assigned when mapping the drive.
2. Navigate to Program Files / Keysight / Network Analyzer / Automation
3. Double-click pnaproxy.exe and follow the prompts to Install PNA Proxy. If the installation offers a choice of Modify, Repair, or Remove, then select Remove. Then double-click on pnaproxy.exe again.
4. When prompted, type the Computer name of the VNA (Learn how to find this).
5. After the install program runs, the VNA and FCA type library should be registered on your PC.
6. Your programming environment may require you to set a reference to the VNA type library now located on your PC. In Visual Basic, click Project, References. Then browse to C:/Program Files/Common Files/Keysight/PNA Select 835x.tlb

**Problems?**

- These procedures will fail if there are any programs using the VNA type library (for example: Visual basic, VEE, Visual Studio, or any other application program that may communicate with the VNA).
Perform the following procedure if the previous procedure did not return an error, but you cannot connect to the VNA.

If you received an error, check that both the account name and password used on both the VNA and PC match EXACTLY.

If you still get errors, see http://na.support.keysight.com/pna/DCOMSecurity.html.

1. Map a drive from your remote PC to the VNA. Note the drive letter your PC assigns to the VNA. Substitute this drive letter for VNA in the following procedure.

2. On your PC, go to a DOS prompt c:>

3. Type PNA: (for example o:)

4. Type cd program files/Keysight/network analyzer/automation

5. Type copy 835xps.dll c:/program files/common files/Keysight/pna

6. Type copy 835x.tlb c:/program files/common files/Keysight/pna

If you will NOT be using FCA commands, skip steps 7, 8, and 9.

7. Type cd..

8. Type cd extensions/fca

9. Type copy fca.tlb c:/program files/common files/Keysight/pna

10. If it is not already there, copy regtlib.exe from PNA:/WINNT to your C:/<windows>/system32 directory (<windows> is OS-dependent- it is either windows or WINNT)

11. Type regtlib C:/program files/common files/Keysight/pna/835x.tlb

12. Type regsvr32 C:/program files/common files/Keysight/pna/835xps.dll

13. Type regtlib C:/program files/common files/Keysight/pna/fca.tlb

Perform the Access Procedure after doing these steps.
COM Fundamentals

The following terms are discussed in this topic:

- Objects
- Interfaces
- Collections
- Methods
- Properties
- Events
- Visual Basic Syntax

**Note:** The information contained in this topic is intended to help an experienced SCPI programmer transition to COM programming. This is NOT a comprehensive tutorial on COM programming.

**Other Topics about COM Concepts**

**Visual Basic Syntax**

The examples in VNA Help use Visual Basic as the programming environment for COM, which uses 'dot' notation.

To set a property, follow the object reference with:

- a period (.)
- property or method
- an equal sign (=)
- the new value

For example:

```vbnet
object.property = value 'This Green text following an apostrophe (') is a comment.
```
To read a property, a variable to contain the returned value is followed with:

- an equal sign (=)
- an object, or reference to an object
- a period (.)
- property

For example:

```
variable = object.property
```

To execute a method, an object, or reference to an object is followed with:

- a period (.)
- the method
- a blank space
- any required parameters

For example:

```
object.method parameters
```

Some methods return values, such as methods that return data. To return data from a method, a variable to contain the returned data is followed with:

- an equal sign (=)
- an object, or reference to an object
- a period (.)
- the method
- any required parameters enclosed in parenthesis

```
variable = object.method (parameters)
```
**Objects**

The objects of the Network Analyzer (Application) are arranged in a hierarchical order. The **VNA object model** lists the objects and their relationship to one another.

In SCPI programming, you must first select a measurement before making settings. With COM, you first get a handle to the object (or collection) and refer to that object in order to change or read settings (properties).

For more information on working with objects, see *Getting a Handle to an Object*.

**Interfaces**

A COM Interface is the connection to an object. When you get a handle to an object, you are actually using an interface to an object. This is important if you are developing VNA code that will run on multiple code versions. For more information, see *VNA Interfaces*.

**Collections**

A collection is an object that contains several other objects of the same type. For example, the **Channels** collection contains all of the channel objects.

*Note:* In the following examples, the collections are referred to as a variable. Before using a collection object, you must first get an instance of that object. For more information, see *Getting a Handle to an Object*.

Generally, items in a collection can be identified by **number** or by **name**. The order for objects in a collection cannot be assumed. They are always unordered and begin with 1. For example, in the following procedure, chans(1) is used to set averaging on the first channel in the Channels collection (not necessarily channel 1).

```
Sub SetAveraging()
    chans(1).AveragingFactor = 10
End Sub
```

The following procedure uses the measurement string name to set the display format for a measurement in the measurements collection.

```
meass("CH1_S11_1").Format = 1
```

You can also manipulate an entire collection of objects if the objects share common methods. For example, the following procedure sets the dwell time on all of the segments in the collection.

```
Sub setDwell()
    For Each seg In segs
        segs.DwellTime = 0.03
    Next
```

The objects of the Network Analyzer (Application) are arranged in a hierarchical order. The **VNA object model** lists the objects and their relationship to one another.

In SCPI programming, you must first select a measurement before making settings. With COM, you first get a handle to the object (or collection) and refer to that object in order to change or read settings (properties).

For more information on working with objects, see *Getting a Handle to an Object*.

A COM Interface is the connection to an object. When you get a handle to an object, you are actually using an interface to an object. This is important if you are developing VNA code that will run on multiple code versions. For more information, see *VNA Interfaces*.

A collection is an object that contains several other objects of the same type. For example, the **Channels** collection contains all of the channel objects.

*Note:* In the following examples, the collections are referred to as a variable. Before using a collection object, you must first get an instance of that object. For more information, see *Getting a Handle to an Object*.

Generally, items in a collection can be identified by **number** or by **name**. The order for objects in a collection cannot be assumed. They are always unordered and begin with 1. For example, in the following procedure, chans(1) is used to set averaging on the first channel in the Channels collection (not necessarily channel 1).

```
Sub SetAveraging()
    chans(1).AveragingFactor = 10
End Sub
```

The following procedure uses the measurement string name to set the display format for a measurement in the measurements collection.

```
meass("CH1_S11_1").Format = 1
```

You can also manipulate an entire collection of objects if the objects share common methods. For example, the following procedure sets the dwell time on all of the segments in the collection.

```
Sub setDwell()
    For Each seg In segs
        segs.DwellTime = 0.03
    Next
```
Methods

A method is an action that is performed on an object. For example, `CreateSParameter` is a method on the `Application` object. The following procedure uses that method to create a new S21 measurement in channel 1 in a new window.

```vba
Sub CreatMeas
    app.CreateSParameter 1, 2, 1, 1
End Sub
```

Properties

A property is an attribute of an object that defines one of the object's characteristics, such as size, color, or screen location. A property can also change an aspect of the object's behavior, such as whether the object is visible. In either case, to change the characteristics of an object, you change the values of its properties.

For example, the following statement sets the IF Bandwidth of a channel to 1 KHz.

```vba
Chan.IFBandwidth = 1e3
```

You can also read the current value of a property. The following statement reads the current IF Bandwidth of a channel into the variable `Ifbw`.

```vba
Ifbw = Chan.IFBandwidth
```

Some properties cannot be set and some cannot be read. The Help topic for each property indicates if you can:

- Set and read the property (Write/Read)
- Only read the property (Read-only)
- Only set the property (Write-only)

Events

A COM event is an action recognized by an object, such as clicking the mouse or pressing a key. Using events, your program can respond to a user action, program code, or triggered by the analyzer.

The SCPI equivalent of an event is a Service Request (SRQ).

For example:

```vba
OnChannelEvent
```
For more information, see *Working with the Analyzer's Events*.
Getting a Handle to an Object

The following are discussed in this topic:

- What Is a Handle
- Declaring an Object Variable
- Assigning an Object Variable
- Navigating the Object Hierarchy
- Getting a Handle to a Collection

Other Topics about COM Concepts

What Is a Handle

In SCPI programming, you must first select a measurement before changing or reading settings. With COM, you first get a handle to the object (or collection) and refer to that object to change or read its settings. The following analogy illustrates this:

A CAR could be called an object. More precisely, CAR is a class of objects. For example, one of the properties of the CAR class is "Color". You can read (by looking) or set (by painting) the color property of a car object. In other words, you can only read or set properties of a specific car object; not the entire car class. Therefore, to read or set a property, you need to get "a handle", or an instance of the object.

This process is also called "accessing an object", "getting an instance of an object", "returning an object", or "referring to an object". You can have handles to many instances of an object at the same time.

Accessing VNA Objects

The VNA Application object is the highest object in the VNA object model hierarchy. Because of that, it is the only object that must be 'created' before it, or any other objects, can be accessed and used. During the creation process, the application object is assigned to a variable name, or handle. Throughout your program, that object is used by referring to that variable. All VNA objects can be assigned to a variable, and subsequently referred to, in this same manner.

The following example shows how to create the VNA Application object, as well as illustrate the general steps of get a handle to an object.

There are two steps in the process of getting a handle to analyzer objects:
1. Declaring a Variable

**Note:** The examples in these topics use the Visual Basic Programming Language. See the short section regarding Visual Basic syntax.

Use the Dim statement or one of the other declaration statements (Public, Private, or Static) to declare a variable. The type of variable that refers to an object must be a Variant, an Object, or a specific type of object. Some programming languages, such as VBScript and Keysight VEE, do not allow you to specify variable types.

The following examples ALL declare the variable **VNA**. Each subsequent statement is more specific than the previous:

- `Dim pna 'Variant data type.``
- `Dim pna As Object 'Object data type.``
- `Dim pna As AgilentPNA835x.Application ' Specific Application type``
- `Dim pna As AgilentPNA835x.IApplication ' Interface type``

1. If you use a variable without declaring it first, the data type of the variable is Variant. If you don't care about using automatic type checking, and willing to run code less efficiently, this method is very safe and is useable on all programming environments.

2. If you know the specific object type, and your programming environment allows it, you can declare the variable as an object.

3. Declaring a specific object type provides automatic type checking (Intellisense), faster code, and improved readability.

4. Declaring the interface is the most specific way and is beneficial when developing code for multiple firmware revisions. Learn more about Interfaces.

2. Assigning an Object to a Variable

To assign an object instance to a variable, use the **Set** keyword before the object variable that was declared previously. In the following line of code, we SET the current AgilentPNA835x Application to "pna".

```
Set pna = AgilentPNA835x.Application
```

As mentioned earlier, the AgilentPNA835x object is unique because it is the highest level of object in
the VNA object model hierarchy. Therefore, we must use the `CreateObject` keyword with the `(classname,server name)` parameters.

- The `classname` for the analyzer object is always "AgilentPNA835x.Application".
- To find your analyzer's `server name`, see View or change full computer name

The following statements create an instance of the Analyzer object.

```vba
Dim pna As AgilentPNA835x.Application
Set pna = CreateObject("AgilentPNA835x.Application", "Analyzer46")
```

**Note:** These statements will start the VNA application if it is not already running on your instrument.

### Navigating the Object Hierarchy

Once an instance of the VNA Application is "created", you access all of the VNA objects by navigating the object hierarchy. Navigating the object model hierarchy can be tricky. In addition, you also need to know how to refer to a specific instance of that object. For example, if you have three measurements present on the VNA, how do you refer to the channel 1 measurement? Each object on the VNA Object Model image is linked to an object page. At the top of each object page is a Description section and another called "Accessing the ... Object". These sections together explain how to navigate the VNA hierarchy to access a specific instance of that object.

From the previous discussion, you may think that you must always declare and assign variables to an object before setting or reading its properties. While this method is best for objects that you will continue to reuse, such as a measurement, it is not always necessary. You can also refer to an object directly.

The `TriggerSetup` object, which is a child of the Application object. Because we will only need to refer to this object once to set a couple of properties, and it is easy to access, we will refer to it directly. From the previous example, we already have a handle to the Application object in the variable `VNA`. The following example uses Visual basic 'dot' notation to refer to the TriggerSetup object, and then the Scope property.

```vba
pna.TriggerSetup.Scope = naChannelTrigger
```

By referring to the TriggerSetup object directly, we must type the same path whenever we refer to properties on the TriggerSetup object. The following method assigns the `VNA.TriggerSetup` object to a variable that can be reused.

```vba
Dim trig As Object
Set trig = pna.TriggerSetup
```

Once created, you can treat an object variable exactly the same as the object to which it refers. For
example:

```vbnet
trig.Scope = naChannelTrigger
trig.Source = naTriggerSourceInternal
```

**Getting a Handle to a Collection**

The analyzer has several collections of objects which provide a convenient way of setting or reading all of the objects in the collection with a single procedure. Also, there are objects (limit lines for example) that can only be accessed through the collection.

To get a handle to an item in a collection, you can refer to the object by item number or sometimes by name. However, you first have to get a handle to the collection. To assign the collection to a variable, use the same two step process (1. declare the variable, 2. assign the variable using 'Set').

```vbnet
Dim meass As Measurements
Dim meas As Measurement
```

You can then iterate through the entire collection of measurements to read or set properties

```vbnet
Sub setFormat()
    For Each meas In meass
        meas.Format = naDataFormat_LinMag
    Next
End Sub
```

Or you can read or set a property on an individual object in the collection:

```vbnet
meass(1).Format = naLinMag
```

**Note:** Each object and collection has its own unique way of dealing with item names, and numbers. Refer to the [Analyzer Object Model](#) for details.
Collections in the Analyzer

Collections are a gathering of similar objects. They are a convenience item used primarily to iterate through the like objects in order to change their settings. Collections generally provide the following generic methods and properties:

<table>
<thead>
<tr>
<th>Item(n)</th>
<th>Count</th>
<th>Add(n)</th>
<th>Remove(n)</th>
</tr>
</thead>
</table>

where \((n)\) represents the number of the item in the collection. Some collections may have unique capabilities pertinent to the objects they collect.

Other Topics about COM Concepts

Collections are Dynamic

A collection does not exist until you ask for it. When you request a Channels object (see Getting a Handle to an Object / Collection), handles to each of the channel objects are gathered and placed in an array.

For example, if channels 2 and 4 are the only channels that exist, then the array will contain only 2 items. The command 'channels.Count' will return the number 2, and:

- Channels(1) will contain the channel 2 object.
- Channels(2) will contain the channel 4 object.

The ordering of objects within the collection should not be assumed. If you add a channel to the previous example, as in:

\[\text{Pna.Channels.Add(3)}\]

'channels.Count' will now return 3 and:

- Channels(1) will contain the channel 2 object.
- Channels(2) will contain the channel 3 object.
- Channels(3) will contain the channel 4 object.

Primarily, collections are useful for making this type of iteration possible:
Dim ch as Channel
For each ch in pna.Channels
  Print ch.Number
  Print ch.StartFrequency
  Print ch.StopFrequency
Next ch

As soon as this for-each block has been executed, the Channels object goes out of scope.
The VNA uses several data types to communicate with the host computer. Before using a variable, it is best to declare the variable as the type of data it will store. It saves memory and is usually faster to access. The following are the most common data types:

- **Long Integer**
- **Single Precision (Real)**
- **Double Precision (Real)**
- **Boolean**
- **String**
- **Object**
- **Enumeration**
- **Variant**

### Other Topics about COM Concepts

**Long** (long integer) variables are stored as signed 32-bit (4-byte) numbers ranging in value from -2,147,483,648 to 2,147,483,647.

**Double** (double-precision floating-point) variables are stored as IEEE 64-bit (8-byte) floating-point numbers ranging in value from -1.79769313486232E308 to -4.94065645841247E-324 for negative values and from 4.94065645841247E-324 to 1.79769313486232E308 for positive values.

**Single** (single-precision floating-point) variables are stored as IEEE 32-bit (4-byte) floating-point numbers, ranging in value from -3.402823E38 to -1.401298E-45 for negative values and from 1.401298E-45 to 3.402823E38 for positive values.

**Boolean** variables are stored as 16-bit (2-byte) numbers, but they can only be True or False. Use the keywords True and False to assign one of the two states to Boolean variables.

When other numeric types are converted to Boolean values, 0 becomes False and all other values become True. When Boolean values are converted to other data types, False becomes 0 and True becomes -1.
The following properties return True rather than 1 to conform with this definition. This may affect the functionality of your COM program:

- Bandwidth Tracking Property
- ErrorCorrection Property
- Interpolate Correction Property
- LimitTestFailed Property

**String** variables hold character information. A String variable can contain approximately 65,535 bytes (64K), is either fixed-length or variable-length, and contains one character per byte. Fixed-length strings are declared to be a specific length. Variable-length strings can be any length up to 64K, less a small amount of storage overhead.

**Object** variables are stored as 32-bit (4-byte) addresses that refer to objects within the analyzer or within some other application. A variable declared as Object is one that can subsequently be assigned (using the Set statement) to refer to any actual analyzer object.

**Enumerations (Enum)** are a set of named constant values. They allow the programmer to refer to a constant value by name instead of by number. For example:

```vbnet
Enum DaysOfWeek
  Sunday = 0
  Monday = 1
  Tuesday = 2
  Wednesday = 3
  Thursday = 4
  Friday = 5
  Saturday = 6
End Enum
```

Given this set of enumerations, the programmer can then pass a constant value as follows:

```vbnet
SetTheDay(Monday)
```

rather than

```vbnet
SetTheDay(1)
```
where the reader of the code has no idea what the value 1 refers to.

However, the analyzer RETURNS a long integer, not the text.

```vbnet
Day = DaysofWeek(today) 'Day = 1
```
**Variant** - If you don't declare a data type ("typed" data) the variable is given the Variant data type. The Variant data type is like a chameleon - it can represent many different data types in different situations.

The VNA provides and receives Variant data because there are programming languages that cannot send or receive "typed" data. Variant data transfers at a slower rate than "typed" data.
A COM interface is the connection to an object. When you get a handle to an object, you are actually using an interface to an object. This subtle distinction is relevant to the COM programmer for the following two reasons:

- Interface Inheritance (Coding for Multiple VNA Versions)
- Custom Interfaces.

**Interface Inheritance (Coding for Multiple VNA Versions)**

The VNA continues to evolve and release new firmware / software versions that provide more functionality and features. New commands are added to existing objects, and with them new interfaces are added to support those commands. For example, new commands were added to the Measurement object in VNA release 3.0. These commands are accessible from the new IMeasurement2 interface. This can be important if you develop code using the type library in release 3.0, and run the code on a VNA with an older release, such as 2.0

When you use a command that was new with release 3.0, and you run that code on a VNA with release 2.0 firmware, errors will occur because that VNA does not recognize the new commands. However, even if you do NOT utilize new commands, errors can still occur. The following example shows how this occurs and how to avoid it.

The following Visual Basic statement dimensions the `meas` variable as an object.

```dim meas As Measurement```

When the program compiles, Visual Basic figures out what interface to use to access that object. When dimensioning as an object, VB will use the default interface. As new interfaces are added to an object, they become the default interface. If this program was developed and compiled using the VNA 3.0 type library, the default Interface of the Measurement Object was IMeasurement2. However, if this program is run on an instrument with VNA 2.0 firmware, there was no IMeasurement2 Interface, and an E_NOINTERFACE error will occur.

Therefore, the more robust approach would be to specify the interface instead of the object when declaring a variable.

```dim meas As IMeasurement```
This code will ONLY use the IMeasurement interface; not the default interface.

However, regardless of how you declare a variable, errors will always occur if you use new commands, and run the code on an older instrument.

**Custom Interfaces**

The VNA object model contains three "custom" interfaces use "typed" variables, which is more efficient than using variant type variables. However, these interfaces are only usable from VB6, C, & C++. All other programming languages must use the other standard interfaces.

The custom interfaces are:

- **IArrayTransfer** - Measurement object
- **ICalData** - Calibrator object
- **ISourcePowerCalData** - Channel object
What are Events?

Windows applications work from user-initiated events such as mouse moves and mouse clicks. A mouse-click produces an event that the programmer can either ignore or "handle" by providing an appropriate subroutine like this:

```vba
Sub DoThis_onClick
    Perform something
End Sub
```

If this subroutine were in your program and the mouse-click event occurs on your PC, it would generate a "Callback" to the client and interrupt whatever it was doing and handle the event.

A more practical example of an event in the analyzer is Limit test. If limit test is on and the measurement fails, the analyzer produces a "Limit-failed" event. If the measurement passed, the analyzer produces a "Limit-succeeded" event.

The Analyzer has a very sophisticated Event structure. Your program **CAN** be notified when one or more events occur. However, it may not be necessary.

For example, the analyzer has an event that will notify your program when a sweep is complete. A
A simpler alternative is to use a synchronous command which waits for the sweep to complete.

```plaintext
sync = True
app.ManualTrigger sync
chan.StartFrequency = 4.5E6
```

This would NOT work if you want the controller to do other things while waiting, like setup a power meter or sort some data. In this case you would like a "callback" from the analyzer to let your program know that the sweep has completed. For an example of this see Events Example.

Another reason to use events is when you want to be notified of several conditions when they occur, such as errors or source unlock conditions. It would not be practical to routinely poll these conditions while executing your program.

### Using Events

If you decide to use the COM events to get a callback, your program must do two things:

1. **Subscribe to events:**

   All events in the analyzer are a child of the Application object through the INetworkAnalyzerEvents Interface. You must tell the Application object that you are interested in receiving event callbacks. This process is called subscription.

   In Visual Basic, this is done by including " WithEvents " in the declaration statement. The declaration below dimensions an Application object (myPNA) and subscribes to the events produced by the Application.

   ```vb
   Dim WithEvents myPNA as AgilentPNA835x.Application
   ```

   In C++, this is a bit more involved. You must queryInterface for the IconnectionPointContainer interface, locate the InetworkAnalyzerEvents interface via a call to FindConnectionPoint and call Advise().

2. **Implement the Event Handler**

   When an event occurs, the Application object will "callback" to the client through the INetworkAnalyzerEvents interface.

   In VB, click on the object window (upper left pane). Find the Application object and click it. The event interfaces will appear in the upper right pane. As you click on them, VB supplies the first line of code. You fill in the rest of the handler routine to service the event. The following is an example of a event handler subroutine.

   ```vb
   Note: In C++, you must type the callback.
   ```
Private Sub OnChannelEvent( eventID as Variant, channelNumber as Variant)
Select Case (eventID)
    Case naEventID_CHANNEL_TRIGGER_COMPLETE:
        GetData( channelNumber )
    Case naEventID_CHANNEL_TRIGGER_ABORTED:
        MsgBox( "Hey don't touch the front panel!"")
End Select
End Sub

When the trigger is complete, the application object "fires" the event by making a callback to the event handler Sub OnChannelEvent().

---

### Event IDs

<table>
<thead>
<tr>
<th>Sev</th>
<th>CR</th>
<th>Facility</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

---

### Filtering Events

There are over 140 different events that you subscribe to when you "Dim WithEvents..." (or the equivalent in your programming language). Monitoring all of these conditions slows the speed of the analyzer significantly. The following methods allow you to filter the events so that you only monitor specific conditions.

- **AllowEventMessage** - monitor a specific event
- **AllowAllEvents** - monitor ALL events
- **DisallowAllEvents** - monitor NO events
- **AllowEventCategory** - monitor specific event categories (discussed later)
- **AllowEventSeverity** - monitor events having one or more of the following severity levels associated with them.

### Severity Enumeration

<table>
<thead>
<tr>
<th>Code</th>
<th>Severity Enumeration</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>naEventSeveritySUCCESS - the operation completed successfully</td>
</tr>
<tr>
<td>01</td>
<td>naEventSeverityINFORMATIONAL - events that occur without impact on the measurement integrity</td>
</tr>
<tr>
<td>10</td>
<td>naEventSeverityWARNING - events that occur with potential impact on measurement integrity</td>
</tr>
<tr>
<td>11</td>
<td>naEventSeverityERROR - events that occur with serious impact on measurement integrity</td>
</tr>
</tbody>
</table>
List of Events

The following is a list of categories and the general types of events they include. Click the link view the event details.

<table>
<thead>
<tr>
<th>Category Enumeration</th>
<th>Callback</th>
</tr>
</thead>
<tbody>
<tr>
<td>naEventCategory_PARSER</td>
<td>OnSCPIEvent</td>
</tr>
<tr>
<td>naEventCategory_MEASURE</td>
<td>OnMeasurementEvent</td>
</tr>
<tr>
<td>naEventCategory_CHANNEL</td>
<td>OnChannelEvent</td>
</tr>
<tr>
<td>naEventCategory_HW</td>
<td>OnHardwareEvent</td>
</tr>
<tr>
<td>naEventCategory_CAL</td>
<td>OnCalEvent</td>
</tr>
<tr>
<td>naEventCategory_USER</td>
<td>OnUserEvent</td>
</tr>
<tr>
<td>naEventCategory_DISPLAY</td>
<td>OnDisplayEvent</td>
</tr>
<tr>
<td>naEventCategory_GENERAL</td>
<td>OnSystemEvent</td>
</tr>
</tbody>
</table>

**Note:** Use the MessageText Method to get a text message describing the event.

Out of Range Errors

When you attempt to set a value on an active function that is beyond the range (min or max) of the allowable values, the analyzer limits that value to an appropriate value (min or max) and sets the function to the limited value. From the front panel controls this is visually evident by the limited value in the edit box or by the annotation on the display. An example would be attempting to set the start frequency below 300kHz. The edit control doesn't allow the number to fall below 300kHz.

When the automation user programs a setting (such as start frequency below the allowable limits) the same behavior takes place. The analyzer accepts the limited value. However, in order to learn what setting took place, you have to read the HRESULT.

All automation calls return HRESULTs. By default the HRESULT returned when an overlimit occurs is S_NA_LIMIT_OUTOFRANGE. This value is a success code, meaning that bit 31 in this 32 value is 0. Programmers should check the return code from all automation calls to determine success or failure.

Some C++ macros (like SUCCEEDED(hr) or FAILED(hr)) only check bit 31. So if you are interested in trapping this outOfRange error you will have to check for S_NA_LIMIT_OUTOFRANGE explicitly.

Alternatively, you can configure the analyzer to report outOfRange conditions with an error code. Use the method: App.SetFailOnOverRange (true). With this method set TRUE, any overrange error will return E_NA_LIMIT_OUTOFRANGE_ERROR.

This method is provided for the benefit of VB clients. VB users can't detect specific success codes...
because the VB runtime strips off the HRESULT and only raises a run time error if bit 31 is set, indicating a fail code.

**Troubleshooting Problems with Callbacks**

When you do callbacks, the client PC becomes the server and the analyzer (server) becomes the client. Callbacks can only take place when both server and client are in the same workgroup or in the same domain. See [Configure for COM](#).
Read and Write Calibration Data using COM

Calibration data in the VNA is stored in Cal Sets. Learn more about Cal Sets

You can read or write two types of Calibration data:

- **Error Terms** - calculated data using standard measurement data and the algorithms for the specified cal type.

- **Standard Measurement data** - raw data resulting from the measurement of a calibration standard.

Each of these data are available in the VNA in either variant data or typed data. Learn more about variant and typed data

Other Topics about COM Concepts

**Calibration / Cal Set Interfaces**

There are several interfaces associated with Calibration.

*ICalibrator*

This interface is the original interface provided with the first version of the VNA. It provides remote access to the "Unguided" Calibration wizard. This interface can perform 1 and 2 port calibrations as well as response cals.

This interface can also read and write error terms from/to a Cal Set. However, ICalibrator is NOT recommended for this purpose. The ICalSet2 Interface is better suited for reading and writing error terms.

See a vbscript example of how to perform a 2-port Cal and read the cal data.

*IGuidedCalibration*

This interface provides the methods and properties used by the Guided Calibration wizard. With this interface you can perform multi-port calibrations (1 to 4 port cals), but no response cals.

*ICalSet2 and ICalData3*

These interfaces provide access to the Cal Set contents. You can read and write error terms with both of these interfaces.
• ICalSet2 uses Variant data, which means it is usable from vbscript.

• ICalData3 uses "typed" data, which means it can be used from any automation engine that can read the type library (VEE, VB, C++, etc.). Typed arguments (such as float or single) are more efficient than variants, so use the ICalData3 interface where better performance is needed.

See a vbscript example of how to read Cal Set data.

ICalSet3

This interface provides access to the stimulus attributes of the Cal data: frequency, power, number of points. These are the stimulus conditions under which the Cal Set was created.
Programming the VNA with C++

The programming information contained in this Help system is aimed at the Visual Basic programmer. VB does a lot of work for the programmer when it comes to managing and accessing components. Using a lower level language like C++ requires a more thorough understanding of the underlying tenets of COM. It is not the intent of this section to teach COM programming. The following is intended to acquaint you with some of the basic concepts you need to know in order to program against COM.

- Initializing COM
- Importing the Type Library
- Creating the Application Object
- Errors
- Events
- Additional Reading
- Example

**Note:** The information in this section assumes development on a Windows OS using Microsoft tools.

### Other Topics about COM Concepts

#### Initializing COM

The first thing you must do before performing any COM transactions is to initialize the COM library. You can do this in a number of ways. The most basic of these is a call to `CoInitialize()` or `CoInitializeEx()`. Alternatively you can use the MFC (Microsoft Foundation Classes) `AfxOleInit()`. Conversely, before your program exits you must uninitialize COM. You can accomplish this with `CoUninitialize()` or the MFC routine `AfxOleTerm()`.

#### Importing the Type Library

To make a component available to the client, the server exports what is called the type library. For the VNA, this file is 835x.tlb. It is located on the VNA's hard drive at `C:\Program Files(x86)\Keysight\Network Analyzer\Automation`. See Configure for COM-DCOM Programming.

The type library can be read and deciphered using another COM interface called ITypeLib. VB uses this interface to present, for example, its object browser. Visual C++ can also read type libraries. This
is done by importing the type library into your project with a compiler directive:

```csharp
#import "C:\Program Files\Common Files\Keysight\Pna\835x.tlb", named_guids
```

When you compile your program with this statement in it, the compiler creates two other files: `835x.tlh` and `835x.tli`. The first is a header file that contains the type definitions for the VNA's COM interfaces and their methods. The second file contains inline functions that wrap the VNA's interface methods. The wrappers are beneficial in that they contain error reporting for each of the method calls.

The `.tlh` file defines a smart pointer which you can use to access the VNA's objects. The smart pointer definition looks like this:

```csharp
_com_smartptr_typedef(Iapplication, _uuidof(Iapplication))
```

A smart pointer is a term used for a C++ object that encapsulates a pointer used to refer to a COM object. All COM objects derive from the interface `IUnknown`. This interface has three methods: `QueryInterface()`, `AddRef()`, and `Release()`. The function of the `AddRef` and `Release` methods is to maintain a reference count on the object and thus control the object's lifetime. Anytime you copy or create a reference to a COM object, you are responsible for incrementing its reference count. And likewise, when you are finished using that reference, it is your responsibility to Release it. Smart pointers do this work for you, as shown in the example program. In addition, smart pointers will also perform the `QueryInterface` call when required. `QueryInterface` is a method that requests a specific interface from an object. In the example program we gain access to the `IArrayTransfer` interface of the `Measurement` object. In the `ReadMethod` routine, we see this:

```csharp
PTransferData = pMeas;
```

The assignment operator is overloaded for the smart pointer and in reality, this simple statement does this:

```csharp
HRESULT hr = pMeas->QueryInterface( IID_IArrayTransfer,(void**)&pTransferData);
```

Using the existing interface pointer (`pMeas`) to the object, this call asks the object if it supports the `IArrayTransfer` interface, and if so to return a pointer to it in `pTransferData`. Smart pointer makes life easier for the C++ programmer. Read more about smart pointers in Microsoft Developer's Network Library (MSDN).

---

Creating the Application Object

The only createable object exported by the VNA is the Application object. Typically this would be done with a call to `CoCreateInstance`:

```csharp
STDAPI CoCreateInstance( 
    CLSID__IApplication, //Class identifier (CLSID) of the object 
    NULL, //Pointer to controlling IUnknown 
    CLS_CTX_SERVER, //Context for running executable code 
```
With the smart pointer, this is taken care of with the following call:

```c++
IApplicationPtr pNA; // declare the smart pointer
pNA = IApplicationPtr("AgilentPNA835x.Application.1");
```

### Errors

All COM method calls are required to return an HRESULT. This is 32 bit long with a specific format.

- The most significant bit indicates success(0) or failure(1).
- The lower 16 bits indicate the specific failure.

Visual Basic strips off the returned HRESULT and raises an error object for non-successful returns. The C++ programmer must himself be diligent about handling errors. You must check the return value of each COM call to ensure its success.

### Events

The Application object sources the INetworkAnalyzerEvents interface. This object is the source for all events. To use events in C++, you must do two things:

1. Implement the INetworkAnalyzerEvents interface - derive an object from INetworkAnalyzerEvents and implement the methods described there.
2. Subscribe to the IconnectionPoint interface of the Application object. - obtain a pointer to the IConnectionPointContainer interface of the Application object and making the following request:

   ```c++
   FindConnectionPoint( IID_InetworkAnalyzerEvents, &pConnection );
   ```

A successful call to this interface will return a valid pointer in pConnection. Use this pointer to subscribe to the Application object:

```c++
pConnect->Advise( IUnknown* punk, DWORD dwCookie);
```

This call provides the server object with a callback address. The IUnknown pointer in this call is the IUnknown pointer of the object that implements the INetworkAnalyzerEvents interface. This is the event sink. The application object needs a pointer to this object in order to call your interface when an event occurs. The `dwCookie` is your subscription key. Use it to unsubscribe (see Unadvise( ) ).

### Additional Reading
Example

The example uses the smart pointer created by Microsoft Visual Studio. The calls to CoInitialize and CoUninitialize open and close the COM libraries. In the example, notice that the pointers local to the main routine are explicitly released. When smart pointers go out of scope, they will perform this duty implicitly. However, we are calling CoUninitialize before they have the chance to be destroyed, so we are obliged to release them.

See the example program.
Using COM from .NET

To communicate with the VNA from Microsoft .NET enabled languages such as C# and Visual Basic.NET perform the following steps:

1. Configure your PC and VNA for COM-DCOM Programming.

2. Reference the type library within the development environment (see the following exception for managed C++ projects.) In the process of referencing the type library, a .NET assembly is created that wraps the VNA type library with a .NET friendly interface. This .NET assembly is called an Interop Assembly.

**Note:** ONLY 32-bit compiler option is supported (64-bit is NOT supported).

**Exception for managed C++ projects:** To generate the Interop Assembly for managed C++ projects, you must use the tlbimp.exe utility. This utility is described in the MSDN documentation. On your PC, click Start then Run then type: tlbimp.exe 835x.tlb and click OK. After doing this you can use the #using directive to include the Interop Assembly on managed C++ projects.

**Example: Creating a .NET object from C#**

The following is an example that shows how to create a .NET object that connects to the VNA over DCOM. In this example, `machineName` is either the DNS name or the IP address of the VNA to connect with.

```csharp
Type pna = Type.GetTypeFromProgID("AgilentPNA835x.Application", machineName);
AgilentPNA835x.IApplication app = (AgilentPNA835x.IApplication)Activator.CreateInstance(pna);
```

**See C# Example Programs:**

Perform a Guided Cal with C#

Using C#

**Registering the VNA Primary Interop Assembly (PIA) (OPTIONAL)**

The PIA is NOT necessary to communicate with the VNA. The following procedure is useful only when there are two .NET programs that want to share the same VNA interface definitions. Without the PIA, each .NET application would use its own Interop Assembly.

To register the PIA on a machine, you need to have the common language runtime (CLR) installed. This is included with Visual Studio.NET. Then perform the following steps:

**Note:** In the following steps, replace `<local directory>` with the full path name of the specified file on
1. Run the PNAPProxy.exe program as described in Configure for COM-DCOM Programming.

2. On the VNA, copy C:/Program Files/Keysight/Network Analyzer/Automation/AgilentPNA835x.dll to a local directory on your PC. Make a note of this directory.

3. On your PC, click Start, then Run, then type: regasm <local directory>/AgilentPNA835x.dll and click OK to register the dll.

4. Again, click Start, then Run, then type: gacutil /i <local directory>/AgilentPNA835x.dll and click OK to add the assembly to the Global Assembly Cache (GAC).

To Uninstall the PIA, perform the following:

1. On your PC, click Start, then Run, then type: gacutil /u <local directory>/AgilentPNA835x.exe and click OK to remove the assembly from the GAC.

2. On your PC, click Start, then Run, then type: regasm/unregsiter <local directory>/agilentpna835x.dll and click OK to unregister the assembly.

3. To uninstall PNA Proxy.exe use the Add/Remove Programs utility in the control panel.
See Also

- Example Programs
- Find commands using a simulated VNA UI
- See list of all SCPI Errors.
- See Calibrating the VNA Using SCPI
- Synchronizing the Analyzer and Controller
- IEEE-488.2 Common Commands
- Local Lockout

ABORt
Stops all sweeps

AFR
Automatic Fixture Removal

+ CALCulate Click to hide and show CALC SCPI command branches

CALPod
Controls CalPod units

CONTrol
Interface control, ECal module state control, and Rear-panel connector control.

CONTrol:MULTiplexer
Control the E5092A Configurable Multiport Test Set.

CSET
Work with a Cal Set without having to select it into that channel.

DISPLAY
Display settings

FORMAT
Format for data transfer

HCOPY
Hardcopy printing

INITiate
Continuous or manual triggering

LLO
Local Lockout

LXI
LXI communications

MMEMory
Saves and recalls instrument states

OUTPUT
Turns RF power ON and OFF

ROUTE
Controls internal switch to reference receiver. (Opt 81)

+ SENSE Click to hide and show SENSE SCPI command branches

SERVICE
Service commands

+ SOURce Click to hide and show SOURce SCPI command branches
STATus
Reads the VNA status registers

+ SYSTem  Click to hide and show SYSTem SCPI command branches

TRIGger
Trigger measurements
### IEEE 488.2 Common Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*CLS</td>
<td>Clear Status</td>
</tr>
<tr>
<td>*ESE</td>
<td>Event Status Enable</td>
</tr>
<tr>
<td>*ESE?</td>
<td>Event Status Enable Query</td>
</tr>
<tr>
<td>*ESR?</td>
<td>Event Status Enable Register - See *ESR? programming example</td>
</tr>
<tr>
<td>*IDN?</td>
<td>Identify</td>
</tr>
<tr>
<td>*OPC</td>
<td>Operation complete command</td>
</tr>
<tr>
<td>*OPC?</td>
<td>Operation complete query</td>
</tr>
<tr>
<td>*OPT?</td>
<td>Identify Options Query</td>
</tr>
<tr>
<td>*RST</td>
<td>Reset</td>
</tr>
<tr>
<td>*SRE</td>
<td>Service Request Enable</td>
</tr>
<tr>
<td>*SRE?</td>
<td>Service Request Enable Query</td>
</tr>
<tr>
<td>*STB?</td>
<td>Status Byte Query</td>
</tr>
<tr>
<td>*TST?</td>
<td>Result of Self-test Query</td>
</tr>
<tr>
<td>*WAI</td>
<td>Wait</td>
</tr>
</tbody>
</table>

**See Also**

- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

---

**CLS - Clear Status**

Clears the instrument status byte by emptying the error queue and clearing all event registers. Also cancels any preceding *OPC command or query. See [Status Commands](#) and [Reading the Analyzer's Status Registers](#).

---

**ESE - Event Status Enable**

Sets bits in the standard event status enable register. See [Status Commands](#) and [Reading the Analyzer's Status Registers](#).

---

**ESE? - Event Status Enable Query**

Returns the results of the standard event enable register. The register is cleared after reading it. See [Status Commands](#) and [Reading the Analyzer's Status Registers](#).
*ESR - Event Status Enable Register
Reads and clears event status enable register. See Status Commands and Reading the Analyzer’s Status Registers.

*IDN? - Identify
Returns a string that uniquely identifies the analyzer. The string is of the form "Keysight Technologies",<model number>,<serial "number>,<software revision>".

Note: Beginning with Rev 6.01, this command now returns the software revision with 6 digits instead of 4. For example, A.06.01.02.

For 4 state CALPod support the following is required.

- CCT (configurable command table) version 1.4
- Controller FPGA version 8.4

*OPC - Operation complete command
Generates the OPC message in the standard event status register when all pending overlapped operations have been completed (for example, a sweep, or a Default). See Understanding Command Synchronization.

*OPC? - Operation complete query
Returns an ASCII "+1" when all pending overlapped operations have been completed. See Understanding Command Synchronization

*OPT? - Identify Options Query
Returns a comma-separated string identifying the analyzer option configuration.

See a list of VNA options. Refer also to the option number differences between the common option numbers and those returned using this command.

See also SYST:CAP:LIC:CAT? for the installed product license.

*RST - Reset
Executes a device reset and cancels any pending *OPC command or query, exactly the same as a SYSTem:PRESet with one exception: Syst:Preset does NOT reset Calc:FORMAT to ASCII. The contents of the analyzer’s non-volatile memory are not affected by this command.
**SRE - Service Request Enable**

Before reading a status register, bits must be enabled. This command enables bits in the service request register. The current setting is saved in non-volatile memory. See Status Commands and Reading the Analyzer’s Status Registers.

**SRE? - Service Request Enable Query**

Reads the current state of the service request enable register. The register is cleared after reading it. The return value can be decoded using the table in Status Commands. See also Reading the Analyzer’s Status Registers.

**STB? - Status Byte Query**

Reads the value of the instrument status byte. The register is cleared only when the registers feeding it are cleared. See Status Commands and Reading the Analyzer’s Status Registers.

**TST? - Result of Self-test Query**

Returns the result of a query of the analyzer hardware status. An 0 indicates no failures found. Any other value indicates one or more of the following conditions exist. The value returned is the Weight (or sum of the Weights) of the existing conditions. For example:

- If 4 is returned from *TST?, an Overpower condition exists.
- If 6 is returned, both Unleveled and Overpower conditions exist.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>Phase Unlock</td>
<td>The source has lost phaselock. This could be caused by a reference channel open or a hardware failure.</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>Unleveled</td>
<td>The source power is unleveled. This could be a source is set for more power than it can deliver at the tuned frequency. Or it could be caused by a hardware failure.</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>Not used</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>EE Write Failed</td>
<td>An attempted write to the EEPROM has failed. This is possibly caused by a hardware failure.</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>YIG Cal Failed</td>
<td>The analyzer was unable to calibrate the YIG. Either the phaselock has been lost or there has been a hardware failure.</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>Ramp Cal Failed</td>
<td>The analyzer was unable to calibrate the analog ramp generator due to a possible hardware failure.</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>Not used</td>
<td></td>
</tr>
</tbody>
</table>
*WAI - Wait
Prohibits the instrument from executing any new commands until all pending overlapped commands have been completed. See Understanding Command Synchronization
About Triggering

Abort Command

ABORt

Applicable Models: All

(Write-only) Stops all sweeps - then resume per current trigger settings. This command is the same as INITiate:IMMediate (restart) except if a channel is performing a single sweep, ABORt will stop the sweep, but not initiate another sweep.

Examples

<table>
<thead>
<tr>
<th>Examples</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ABOR</td>
</tr>
<tr>
<td></td>
<td>abort</td>
</tr>
</tbody>
</table>

Query Syntax

Not applicable

Default

Not applicable
These commands are used to control the Automatic Fixture Removal (AFR) capabilities in the VNA.

AFR:

FIXTure:

| ADVanced

| MCONversion

| [:STATe]

| RESet

| TIME

| STARt

| STOP

| WINDow

| COEFFicient

| MANual

| [:STATe]

| BLIMited[:STATe]

| CDUT[:STATe]

| CLENght[:STATe]

| CMATch[:STATe]

| INPuts

| MEASurement

| PREView

| DATA

| [:IMPedance]?
M A R K e r : Y ?
R E F Z
S E T
S Y S Z
[ : S T A T e ]
U S E
T H R U s
[ : S T A T e ]
I N I T i a l i z e
S A V E :
F I L e n a m e
I M P e d a n c e
N O R M a l i z e
[ : S T A T e ]
P O R T s
T Y P E
S T A N d a r d :
A L L O p e n [ : S T A T e ]
A L L S h o r t [ : S T A T e ]
D A T A
[ : I M P e d a n c e ]?
M A R K e r : Y ?
E D I T :
F L E N g t h
AFR:ADVanced:MCONversion[:STATe] <bool>

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-Write)* This command turns ON or OFF AFR mode conversion.

**Parameters**

<bool> Choose from:

- **ON (or 1)** - Turns mode conversion on.
- **OFF (or 0)** - Turns mode conversion off.

**Examples**

AFR:ADVanced:MCONversion ON

**Query Syntax**

AFR:ADVanced:MCONversion[:STATe]?

**Return Type**

Boolean

**Default**

OFF

---

**See Also**

- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree
AFR:ADVanced:RESet

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Write-only)* This command resets the AFR configuration.

**Parameters** None

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:ADVanced:RESet</td>
</tr>
</tbody>
</table>

**Query Syntax** Not Applicable

**Default** Not Applicable

AFR:ADVanced:TIME:STARt <num>

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-Write)* This command gets or sets the manual start time in the AFR configuration.

**Parameters**

- `<num>` Start time in ns.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:ADVanced:TIME:STARt -0.2</td>
</tr>
</tbody>
</table>

**Query Syntax**

:AFR:ADVanced:TIME:STARt?

**Return Type** Numeric

**Default** Not Applicable

AFR:ADVanced:TIME:STOP <num>

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-Write)* This command gets or sets the manual stop time in the AFR configuration.

**Parameters**

- `<num>` Stop time in ns.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:ADVanced:TIME:STOP 5.0</td>
</tr>
</tbody>
</table>

**Query Syntax**

:AFR:ADVanced:TIME:STOP?

**Return Type** Numeric

**Default** Not Applicable

AFR:ADVanced:WINDow:COEFFicient <char>
**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-Write)* This command gets or sets the manual window type in AFR configuration.

**Parameters**

<char> Window coefficient. Choose from:

- **MAXWin**
- **WIDWin**
- **NORMWin**
- **MINWin**

**Examples**

AFR:ADVanced:WINDow:COEFFicient WIDWin

**Query Syntax**
AFR:ADVanced:WINDow:COEFFicient?

**Return Type** String

**Default** NORMWin

---

**AFR:ADVanced:WINDow:MANual[:STATe] <bool>**

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-Write)* This command turns ON or OFF the manual window coefficient of the AFR.

**Parameters**

<bool> Choose from:

- **ON (or 1)** - Turns manual window coefficient on.
- **OFF (or 0)** - Turns manual window coefficient off.

**Examples**

AFR:ADVanced:MANual:STATe ON

**Query Syntax**
AFR:ADVanced:WINDow:MANual[:STATe]?

**Return Type** Boolean

**Default** OFF

---

**AFR:FIXTure:BLIMited[:STATe] <bool>**
Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command selects whether the fixture is band limited or not.

Parameters

<bool> Band limited or not. Choose from:

ON (or 1) - Band limited.

OFF (or 0) - Not band limited.

Examples AFR:FIXTure:BLIMited ON

Query Syntax AFR:FIXTure:BLIMited?

Return Type Boolean

Default OFF

AFR:FIXTure:CDUT[:STATe] <bool>

Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command selects whether to use DUT correction or not when the characterization fixture is not equal to the DUT measurement fixture.

Parameters

<bool> DUT correction state. Choose from:

ON (or 1) - Use DUT correction.

OFF (or 0) - Do not use DUT correction.

Examples AFR:FIXTure:CDUT ON

Query Syntax AFR:FIXTure:CDUT?

Return Type Boolean

Default OFF

AFR:FIXTure:CLENgth[:STATe] <bool>
**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-Write)* This command selects Fixture Length A not equal to B correction.

**Parameters**

<bool> Correction match state. Choose from:

- **ON (or 1)** - Correct.
- **OFF (or 0)** - Do not correct.

**Examples**

<table>
<thead>
<tr>
<th>AFR:FIXTure:CLENgth</th>
<th>ON</th>
</tr>
</thead>
</table>

**Query Syntax**

AFR:FIXTure:CLENgth?

**Return Type**

Boolean

**Default**

ON

---

**AFR:FIXTure:CMATch[:STAte] <bool>**

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-Write)* This command selects Fixture Match A not equal to B correction.

**Parameters**

<bool> Correction match state. Choose from:

- **ON (or 1)** - Correct.
- **OFF (or 0)** - Do not correct.

**Examples**

<table>
<thead>
<tr>
<th>AFR:FIXTure:CMATch</th>
<th>ON</th>
</tr>
</thead>
</table>

**Query Syntax**

AFR:FIXTure:CMATch?

**Return Type**

Boolean

**Default**

ON

---

**AFR:FIXTure:INPuts <char>**

3324
Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command describes the fixture inputs (single ended or differential).

Parameters

<char> Choose from:

SENDed - Single ended fixture inputs.

DIFFerential - Differential fixture inputs.

Examples

AFR:FIXTure:INPuts SENDed

Query Syntax

AFR:FIXTure:INPuts?

Return Type

String

Default

SEND

AFR:FIXTure:MEASurement <num>

Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command selects the number of fixtures to be characterized.

Parameters

<num> Choose from 1, 2, or 4.

Examples

AFR:FIXTure:MEASurement 2

Query Syntax

AFR:FIXTure:MEASurement?

Return Type

Numeric

Default

2

AFR:FIXTure:PREView
Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Write-only) This command refreshes preview data. It is required that this command be sent before a query gate, fixture length, and impedance data.

Parameters
None

Examples
AFR:STANDARD:EDIT:GATE AFIX, 0.485
AFR:FIXTURE:PREVIEW
AFR:STANDARD:EDIT:GATE? AFIX

Query Syntax
Not Applicable

Default
Not Applicable

AFR:FIXTURE:PREVIEW:DATA[:IMPedance]? <char_param>

Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-only) This command reads the impedance profile of the calculated fixture model.

Parameters

<char_param> Selected impedance term. Choose from:

ASENded - Single-EndedZA
BSENded - Single-Ended ZB
ADIFf - Differential ZA
BDIFf - Differential ZB
ACOMm - Common mode ZA
BCOMm - Common mode ZB

Examples
AFR:FIXTURE:PREVIEW:DATA:IMPedance? ADIF

Return Type
Block Data

Default
Not Applicable

AFR:FIXTURE:PREVIEW:DATA[:IMPedance]:MARKer:Y? <char_param>,<num_x>
**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-only)* This command reads the impedance profile of the calculated fixture model at a specified position.

**Parameters**

<char_param> Selected impedance term. Choose from:

- **ASENded** - Single-Ended ZA
- **BSENded** - Single-Ended ZB
- **ADIFF** - Differential ZA
- **BDIFF** - Differential ZB
- **ACOMm** - Common mode ZA
- **BCOMm** - Common mode ZB

<num_x> The X-axis position (ns), where the Y-axis value will be returned.

**Examples**

```
AFR:FIXTure:PREView:DATA:IMPedance:MARKer:Y? ADIF,0.5
```

**Return Type** Numeric

**Default** Not Applicable

---

**AFR:FIXTure:REFZ <char>[,<num>]**

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-Write)* This command chooses the calibration reference Z0 after fixture removal.

**Parameters**

<char> Choose from:

- **SYSZ** - System Z0.
- **MEAZ** - Measured fixture Z0.
- **CUSTom** - User input.

<num> Optional argument. If <char> is set to CUST, this is the user input value for Z0.

**Examples**

```
AFR:FIXTure:REFZ SYSZ
AFR:FIXTure:REFZ CUST,52.0
```
AFR:FIXTure:SET:SYSZ[:STATe] <bool>

Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command sets "System Z0" to Calibration Reference Z0.

Parameters

<bool> Choose from:

ON (or 1) - Sets "System Z0" to Calibration Reference Z0.

OFF (or 0) - Do not set "System Z0" to Calibration Reference Z0.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:FIXTure:SET:SYSZ ON</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

AFR:FIXTure:USE:THRUs[:STATe] <bool>

Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command specifies whether thrus are used in case of multi-port fixtures.

Parameters

<bool> Choose from:

ON (or 1) - Use thrus.

OFF (or 0) - Do not use thrus.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:FIXTure:USE:THRUs ON</td>
<td>Boolean</td>
</tr>
</tbody>
</table>
AFR:INITialize

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Write-only)* Restores the default AFR settings.

**Parameters** None

**Examples** `AFR:INIT`

**Query Syntax** Not Applicable

**Default** Not Applicable

---

AFR:SAVE:FILename <string>[,<string>]

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-Write)* This command specifies the file paths of saved fixture data.

**Parameters**

- `<string>` Fixture A path.
- `<string>` Fixture B path.

**Examples** `AFR:SAVE:FILename 'C:\fixA.s2p','C:\fixB.s2p'`

**Query Syntax** AFR:SAVE:FILename?

**Return Type** String

**Default** "C:\fixA.s2p,C:\fixB.s2p"

---

AFR:SAVE:IMPedance:NORMalize[:STATe]
**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Write-only)* This command specifies whether the port impedances are normalized in saving the AFR fixture files.

**Parameters**

<bool> Choose from:

**ON (or 1)** - Normalize the port impedances when they are not equal.

**OFF (or 0)** - Do not normalize the port impedances.

**Examples**

AFR:SAVE:IMPedance:NORMalize 1

**Return Type** Not Applicable

**Default** OFF

---

**AFR:SAVE:PORTs <char>**

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-Write)* This command assigns the ports for saved fixture data in several formats.

**Parameters**

<char> Port assignment. Choose from:

**PLTS** - PLTS format.

**VNA** - VNA format.

**ADS** - ADS format.

**Examples**

AFR:SAVE:PORTs VNA

**Query Syntax**

AFR:SAVE:PORTs?

**Return Type** String

**Default** VNA

---

**AFR:SAVE:TYPE <char>**
Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command sets the file type to save fixture data.

Parameters

<char> Impedance method. Choose from:

TSOnE - Touchstone file type.
TSTwo - Touchstone 2 file type.
CITifile - Citifile file type.

Examples

AFR:SAVE:TYPE \texttt{TSOnE}

Query Syntax

AFR:SAVE:TYPE?

Return Type

String

Default

TSOnE

AFR:STANdard:ALLOpen[:STATe] \texttt{<bool>}

Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command selects all OPEN standards.

Parameters

<bool> Select all OPEN standards or not. Choose from:

\textbf{ON (or 1)} - Use all OPEN standards.

\textbf{OFF (or 0)} - Do not use all OPEN standards.

Examples

AFR:STANdard:ALLOpen \texttt{ON}

Query Syntax

AFR:STANdard:ALLOpen?

Return Type

Boolean

Default

OFF

AFR:STANdard:ALLShort[:STATe] \texttt{<bool>}

3331
**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command selects all SHORT standards.

**Parameters**

<bool> Select all SHORT standards or not. Choose from:

**ON (or 1)** - Use all SHORT standards.

**OFF (or 0)** - Do not use all SHORT standards.

**Examples**

```
AFR:STANdard:ALLShort ON
```

**Query Syntax**

AFR:STANdard:ALLShort?

**Return Type**

Boolean

**Default**

OFF

---

**AFR:STANdard:DATA[:IMPedance]? <char_std>,<char_param>**

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-only) This command reads the impedance profile of the measured standard.

**Parameters**

<char_std> Selected standard. Choose from:

**THRU** - 2X Thru

**STHRu** - Second 2X Thru

**FDUT** - Fixtured DUT

**AOPen** - Fixture A Open

**BOPen** - Fixture B Open

**ASHort** - Fixture A Short

**BSHort** - Fixture B Short

<char_param> Selected impedance term. Choose from:

**ASENded** - Single-Ended ZA

**BSENded** - Single-Ended ZB

**ADIFf** - Differential ZA
**Examples**

AFR:STANdard:DATA:IMPedance? AOP,ADIF

**Return Type**

Block Data

**Default**

Not Applicable

A FR:STANdard:DATA[:IMPedance]:MARKer:Y? <char_std>,<char_param>,<num_x>

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-only)* This command reads the impedance of the measured standard at a specified position.

**Parameters**

- **<char_std>** Selected standard. Choose from:
  - THRU - 2X Thru
  - STHRu - Second 2X Thru
  - FDUT - Fixture DUT
  - AOPen - Fixture A Open
  - BOPen - Fixture B Open
  - ASHort - Fixture A Short
  - BSHort - Fixture B Short

- **<char_param>** Selected impedance term. Choose from:
  - ASENded - Single-Ended ZA
  - BSENDed - Single-Ended ZB
  - ADIFf - Differential ZA
  - BDIFf - Differential ZB
  - ACOMm - Common mode ZA
  - BCOMm - Common mode ZB
The X-axis position (ns), where the Y-axis value will be returned.

**Examples**

```
AFR:STANdard:DATA:IMPedance:MARKer:Y? AOP,ADIF, 0.5
```

**Return Type** Numeric

**Default** Not Applicable

---

**AFR:STANdard:EDIT:FLENgth <char>,<num>**

**Applicable Models:** All with Automatic Fixture Removal Option (S9x007A/B, 007)

*(Read-Write)* For the selected fixture, this command reads the fixture length for both 1X or 2X AFR, or sets the fixture length for 1X AFR.

**Note:** After setting fixture length to a new value, the preview command must be sent to apply the value.

**Parameters**

- `<char>` Selected fixture. Choose from:
  - `AFIXture` - Fixture A.
  - `BFIXture` - Fixture B.
- `<num>` Fixture length value. Unit is ns.

**Examples**

```
AFR:STANdard:EDIT:FLENgth AFIx,0.3273
AFR:FIXTure:PREView
```

**Query Syntax**

```
AFR:STANdard:EDIT:FLENgth? AFIx
```

**Return Type** Numeric

**Default** Not Applicable

---

**AFR:STANdard:EDIT:GATE <char>,<num>**
Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command sets or reads the gate position for the selected fixture.

**Note:** After setting a new gate position, the preview command must be sent to apply the new position.

**Parameters**

- `<char>` Selected fixture. Choose from:
  - **AFIXture** - Fixture A.
  - **BFIXture** - Fixture B.

- `<num>` Gate position value. Unit is ns.

**Examples**

```
AFR:STANDARD:EDIT:GATE AFIX,0.485
AFR:FIXTURE:PREVIEW
```

**Query Syntax**

```
AFR:STANDARD:EDIT:GATE? AFIX
```

**Return Type** Numeric

**Default** Not Applicable

---

**AFR:STANDARD:EDIT:IMPedance <char>,<num>**

Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Write-only) This command sets the impedance for the selected term.

**Parameters**

- `<char>` Impedance term. Choose from:
  - **ASENded** - Single-Ended ZA.
  - **BSENded** - Single-Ended ZB.
  - **ADIFF** - Differential ZA.
  - **BDIFF** - Differential ZB.
  - **ACOMm** - Common mode ZA.
  - **BCOMm** - Common mode ZB.

- `<num>` Impedance value. Unit is Ohms.

**Examples**

```
AFR:STANDARD:EDIT:IMPedance ASEN,52.5
```

---

3335
AFR:STANdard:EDIT:IMPedance:METHod <char>

Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command sets the impedance method.

Parameters

<char> Impedance method. Choose from:

DEFault
AUTO
USER

Examples AFR:STANdard:EDIT:IMPedance:METHod AUTO

Query Syntax AFR:STANdard:EDIT:IMPedance:METHod?

Return Type Character

Default DEFault

AFR:STANdard:LOAD <char>,<string>

Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command loads the calibration standards data from a file.

Parameters

<char> Standards type. Choose from:

THRU- 2X Thru.
STHRu - Second 2X Thru.
FDUT - Fixtured DUT.
AOPen - Fixture A Open.
BOPen - Fixture B Open.
ASHort - Fixture A Short.
BSHort - Fixture B Short.
<string> File path of existing measurement data (touchstone 1.0, touchstone 2.0, or citifile).

Examples

AFR:STANdard:LOAD AOPen,'C:\open.s1p'

Query Syntax

AFR:STANdard:LOAD? AOPen

Return Type

String

Default

Not Applicable

AFR:STANdard:MEASure <char>[,<p1>][,<p2>][,<p3>][,<p4>]

Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Write-only) This command measures the calibration standards.

Parameters

<char> Standards type. Choose from:

THRU - 2X Thru.
STHRu - Second 2X Thru.
FDUT - Fixture DUT.
AOPen - Fixture A Open.
BOPen - Fixture B Open.
ASHort - Fixture A Short.
BSHort - Fixture B Short.

<p1> - Mapped PNA Port 1. Optional parameter.
<p2> - Mapped PNA Port 2. Optional parameter.
<p3> - Mapped PNA Port 3. Optional parameter.
<p4> - Mapped PNA Port 4. Optional parameter.

Examples

AFR:STANdard:MEASure AOPen

Default

Not Applicable

AFR:STANdard:THRU <char>[,<num>]
Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command specifies fixture thru settings.

Parameters

<char> Thru type. Choose from:

**KNOWn** - Known thru, flush thru, or user input length.

**UNKNown** - Unknown thru length computed using reflects.

<num> User input thru length. Default is 0.

Examples

```
AFR:STANdard:THRU KNOW,0
```

Query Syntax  AFR:STANdard:THRU?

Return Type  String

Default  "KNOW,0"

AFR:STANdard:USE <char>[,<bool>]

Applicable Models: All with Automatic Fixture Removal Option (S9x007A/B, 007)

(Read-Write) This command chooses the calibration standards.

Parameters

<char> Choose from:

**THRU** - 2X Thru.

**STHRu** - Second 2X Thru.

**FDUT** - Fixtured DUT.

**AOPEN** - Fixture A Open.

**BOPEN** - Fixture B Open.

**ASHort** - Fixture A Short.

**BSHort** - Fixture B Short.
Use calibration standards or not. Choose from:

**ON (or 1)** - Use calibration standards.

**OFF (or 0)** - Do not use calibration standards.

**Examples**

```
AFR:STANdard:USE AOPen,ON
```

**Query Syntax**

AFR:STANdard:USE?

**Return Type**

String of used standards, separated by commas.

For example, use AOPen and BOPen returns:

"AOPen,BOPen"

**Default**

All standards - OFF
Calculate:Correction Commands

Controls error correction functions.

These commands are **Superseded** by the `CALCulate:MEASure:CORRection` commands.

```
CALCulate:CORRection

  EDELay
    | DISTance
    | TIME
    | MEDium
    | UNIT
    | WGCutoff

  ERRor
    | [:STATe]
    | TYPE

  [STATe]
    | INDicator?

  TYPE

  OFFSet
    | [MAGNitude]
    | PHASE
```

Click on a keyword to view the command details.

**Blue** keywords are superseded.

See Also

- Example Programs
- Calibrating with SCPI
- Synchronizing the Analyzer and Controller
Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

CALCulate<cnum>:CORRection:EDELay:DISTance <num>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the electrical delay in physical length (distance) for the selected measurement.

See Critical Note

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Electrical delay in distance.

First Specify units using CALC:CORR:EDEL:UNIT

Use SENS:CORR:RVEL:COAX <num> to set Velocity factor.

This parameter supports MIN and MAX as arguments. Learn more.

Examples

- CALC1:CORR:EDEL:DIST 5
- calculate2:correction:distance .003

Query Syntax

CALCulate:CORRection:EDELay:DISTance?

Return Type

Numeric

Default

0

CALCulate<cnum>:CORRection:EDELay:MEDium <char>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the media used when calculating the electrical delay.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<num>` Choose from: COAX for coaxial medium, WAVEguide for waveguide medium.

Examples

- `CALC:CORR:EDEL:MED COAX`
- `calc3:corr:edelay:medium waveguide`

Query Syntax

CALCulate<cnum>:CORRection:EDELay:MEDium?

Return Type

Character

Default

COAX

CALCulate<cnum>:CORRection:EDELay:UNIT <char>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the units for specifying electrical delay in physical length (distance).

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<char>` Units for delay in distance. Choose from:
  
  - METer
  - FEET
  - INCH

Examples

- `CALC:CORR:EDEL:UNIT MET`
- `calc3:corr:edelay:unit inch`

Query Syntax

CALCulate<cnum>:CORRection:EDELay:UNIT?

Return Type

Character

Default

METer

CALCulate<cnum>:CORRection:EDELay[:TIME] <num>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the electrical delay for the selected measurement.

See Critical Note

**Parameters**

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- **<num>** Electrical delay in seconds. Choose any number between **-10.00** and **10.00**
  
  Use `SENS:CORR:RVEL:COAX <num>` to set Velocity factor.

This parameter supports MIN and MAX as arguments. Learn more.

**Examples**

```
CALC1:CORR:EDEL:TIME 1NS
```

```
calculate2:correction:time 0.5e-12
```

**Query Syntax**

```
CALCulate:CORRection:EDELay[:TIME]?
```

**Return Type**

Numeric

**Default**

0 seconds

**CALCulate<cnum>:CORRection:EDELay:WGCutoff <num>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the waveguide cutoff frequency used when the electrical delay media is set to WAVEguide. (See `CALCulate:CORRection:EDELay:MEDium <char>`.)

**Parameters**

- **<cnum>** Any existing channel number. If unspecified, value is set to 1.
- **<num>** Waveguide cutoff frequency used with the electrical delay calculation.

This parameter supports MIN and MAX as arguments. Learn more.

**Examples**

```
CALC:CORR:EDEL:WGC 18.067 GHz
```

```
calculate3:correction:edelay:wgcutoff 14.047 ghz
```

**Query Syntax**

```
CALCulate<cnum>:CORRection:EDELay:WGCutoff?
```

**Return Type**

Numeric

**Default**

45 MHz
CALCulate<cnum>:CORRection:ERRor[:STATe] <bool>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Turns error correction ON or OFF on the specified channel.

To turn error correction ON or OFF for a channel, use `SENS:CORR:STATe`.

See Critical Note

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` Correction state. Choose from:
  - 0 - Correction OFF
  - 1 - Correction ON

**Examples**

- `CALC:CORR:ERR ON`
- `calculate:correction:error:state off`

**Query Syntax**

CALCulate<cnum>:CORRection:ERRor:STATe?

**Return Type**

Boolean

**Default** Not Applicable

---

CALCulate<cnum>:CORRection:ERRor:TYPE <string>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the Cal Type on the specified channel. This is used when a Cal Set is applied. Learn more about applying Cal Types.

- Use `SENS:CORR:TYPE:CAT?` to list the Cal Types in the VNA.
- Use `SENS:CORR:CSET:TYPE:CAT?` to list the Cal Types contained in the active Cal Set for the channel.
- Use `SENS:CORR:COLL:METH` to set the Cal type to perform a new Unguided calibration,

See Critical Note

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<string>` *(String)* Cal type. Case sensitive. Use one of the following:

  **For Full Calibrations (NO Power Cal included):**
This command does not distinguish between TRL and SOLT. The same number of error terms is applied for both Cal Types.

"Full <n> Port(x,y,z...)"

where

<n> = the number of ports to calibrate
x,y,z = the port numbers to calibrate

For example:

"Full 4 Port(1,2,3,4)"

**For Full Calibrations (including Power Cal):**

After the Full <n> port, include the string, "with power"

For example:

"Full 4 Port with power(1,2,3,4)"

**For Response Calibrations:**

"Response(param)" OR

"ResponseAndIsolation(param)"

Where param =

- S-parameter. For example"
  - "Response(S21)"
  - "ResponseAndIsolation(A/R)"

- Single or ratioed receivers using either logical receiver notation or physical receiver notation. For example:
  - "Response(A)"
  - "ResponseAndIsolation(a3/b4)"

**For Enhanced Response Calibrations:**

"EnhancedResp(sourcePort, recPort)"

Where:
- sourcePort = stimulus port number
- recPort = receiver port number

For FCA Calibrations:

Learn more about this setting.

- "SMC_2P" (Response + Input + Output) All four sweeps required. Most accurate.
- "SMCRsp+IN" No Output match. All four sweeps required.
- "SMCRsp+OUT" No Input match. All four sweeps required.
- "SMCRsp" No Input or Output match. Saves two sweeps.

For VMC, multiple Cal types are not available.

For Gain Compression Cal

where r = receive port; s = source port

- "GCA 2P (r,s)" - full 2-port cal
- "GCA Enh Resp (r,s)" - Enhanced Response Cal

For Noise Figure Cal

- VNC_2P - full 2-port Vector Noise Figure correction (requires vector noise figure calset).
- SNC_2P - full 2-port Scalar Noise Figure correction (requires either scalar or vector noise figure calset).

### Examples

CALC:CORR:ERR:TYPE "Scalar Mixer Cal"

### Query Syntax

CALCulate<cnun>:CORRection:ERRor:TYPE?

### Return Type

String

**Default** Not Applicable

CALCulate<cnun>:CORRection[:STATe] <bool>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Turns error correction ON or OFF for the selected measurement on the specified channel.

To turn error correction ON or OFF for a channel, use `SENS:CORR:STATe`.

**See Critical Note**

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` Correction state. Choose from:
  - 0 - Correction OFF
  - 1 - Correction ON

**Examples**

```
CALC:CORR ON
```

```calculate:correction:state off```

**Query Syntax**

CALCulate<cnum>:CORRection:STATe?

**Return Type**

Boolean

**Default**

Not Applicable

---

**CALCulate<cnum>:CORRection[:STATE]:INDicator?**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-only)* Returns the error correction state for the selected measurement on the specified channel.

To turn error correction ON or OFF for a channel, use `SENS:CORR:STATe`.

**See Critical Note**

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Examples**

```
CALC:CORR:IND?
```

```calculate:correction:state:indicator?
```

**Return Type**

Character

- **NONE** - No error correction
- **MAST** (Master) - Original error correction terms
INT - Error terms are interpolated. Learn more.

DELT - Delta Match calibration terms. Learn more.

INV - Error terms are not valid

**Default** - NONE

**CALCulate<cnum>:CORRection:TYPE <string>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the Cal Type for the selected measurement on the specified channel. This is used when a Cal Set is applied. Learn more about applying Cal Types.

- Use SENS:CORR:TYPE:CAT? to list the Cal Types in the VNA.
- Use SENS:CORR:CSET:TYPE:CAT? to list the Cal Types contained in the active Cal Set for the channel.
- Use SENS:CORR:COLL:METH to set the Cal type to perform a new Unguided calibration,

**See Critical Note**

**Parameters**

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<string>`  *String* Cal type. Case sensitive. Use one of the following:

**For Full Calibrations (NO Power Cal included):**

This command does not distinguish between TRL and SOLT. The same number of error terms is applied for both Cal Types.

"Full <n> Port(x,y,z..."

where

<n> = the number of ports to calibrate

x,y,z = the port numbers to calibrate

For example:

"Full 4 Port(1,2,3,4)"

**For Full Calibrations (including Power Cal):**

After the Full <n> port, include the string, "with power"
For example:

"Full 4 Port with power(1,2,3,4)"

For Response Calibrations:

"Response(param)" OR

"ResponseAndIsolation(param)"

Where param =

- S-parameter. For example"
  - "Response(S21)"
  - "ResponseAndIsolation(A/R)"
- Single or ratioed receivers using either logical receiver notation or physical receiver notation. For example:
  - "Response(A)"
  - "ResponseAndIsolation(a3/b4)"

For Enhanced Response Calibrations:

"EnhancedResp(recPort, sourcePort)"

Where:

- recPort = receiver port number
- sourcePort = stimulus port number

For FCA Calibrations:

Learn more about this setting.

- "SMC_2P" (Response + Input + Output) All four sweeps required. Most accurate.

Note: The “SMC_2P” is the only correction type that automatically forces the reverse measurements (SC12, RevOPwr, RevIPwr). If you require any of the reverse measurements while using a lesser correction type, either include those parameters in your channel or use the SENSE:MIXer:REVerse command. Keep in mind that adding the reverse measurements or forcing the reverse sweep
using the SENSE:MIXer:REVerse command will increase the number of sweeps required in the channel.

- "SMCRsp+IN" No Output match. Saves two sweeps.
- "SMCRsp+OUT" No Output match. Saves one sweep.
- "SMCRsp" No Input or Output match. Saves two sweeps.

For VMC, multiple Cal types are not available.

For Gain Compression Cal

where $r = \text{receive port}; s = \text{source port}$

- "GCA 2P (r,s)" - full 2-port cal
- "GCA Enh Resp (r,s)" - Enhanced Response Cal

Examples

```
CALCULATE:CORR:TYPE "Scalar Mixer Cal"
```

Query Syntax

CALCulate<cnum>:CORRection:TYPE?

Return Type

String

Default

Not Applicable

CALCulate<cnum>:CORRection:OFFSet[:MAGNitude] <num> Superseded

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Note: This command is replaced with SENS:CORR:RPOWer:OFFSet[:AMPLitude].

To set data trace magnitude offset, use CALC:OFFS:MAGN

This command does NOT function for FCA measurements.

See an example of a Receiver Power Calibration.

(Read-Write)

For Receiver Power Calibration, specifies the power level to which the selected (unratioed) measurement data is to be adjusted. This command applies only when the selected measurement is of unratioed power.

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
<num> Cal power level in dBm. No limits are enforced on this value, but the VNA receivers themselves have maximum and minimum power specifications (that may differ between VNA models) which this value must comply with for a valid receiver power cal.

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:CORR:OFFS 10DBM</td>
<td>Calculate correction offset magnitude maximum</td>
</tr>
</tbody>
</table>

Query Syntax

CALCulate<cnum>:CORRection:OFFSet[:MAGNitude]?

Return Type

Numeric

Default

0dBm

CALCulate<cnum>:CORRection:OFFSet:PHASe <num>[<char>] Superseded

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Note: This command is replaced with CALC:OFFS:PHASE

(Read-Write) Sets the phase offset for the selected measurement.

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<num> Offset phase value. Choose any number between:

-360 and 360

<char> Units for phase. OPTIONAL. Choose either:

DEG - Degrees (default)
RAD - Radians

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:CORR:OFFS:PHAS 10</td>
<td>Calculate correction offset phase 20rad</td>
</tr>
</tbody>
</table>

Query Syntax

CALCulate:CORRection:OFFSet:PHASE?

Return Type

Numeric, returned value always in degrees

Default

0 degrees
Calculate:Custom Commands

Creates and modifies application measurements.

These commands are **Superseded** by the CALCulate:MEASure:DEFine and CALCulate:MEASure:PARameter commands.

**Note:** For setting up a TDR measurement class, use SYSTem:TDR:INITialize.

### CALCulate:CUSTom:

- DEFine
- MODify

**See Also**

- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

### CALCulate<cnum>:CUSTom:DEFine <Mname>, <type> [,param]

**Applicable Models:** All

*(Write-only)* Creates a custom measurement depending on the configurations and options. The custom measurement is not automatically displayed. You must also do the following:

- Use DISP:WIND:STATe to create a window if it doesn't already exist.
- Use DISP:WIND:TRAC:FEED to display the measurement
- Select the measurement (CALC:PAR:SEL) before making additional settings.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<Mname>` Name of the measurement. Any non-empty, unique string, enclosed in quotes.
- `<type>` *(string)* - Choose from the following (click or scroll down to view valid `<params>` for each type)
- "Standard"
- "Active Hot Parameters"
- "Vector Mixer/Converter"
- "Scalar Mixer/Converter"
- "Gain Compression"
- "Gain Compression Converters"
- Modulation Distortion
- "Noise Figure Cold Source"
- "Noise Figure Converters"
- "Swept IMD"
- "IM Spectrum"
- "Swept IMD Converters"
- Modulation Distortion
- Phase Noise
- "IM Spectrum Converters"
- "Differential I/Q"
- "Spectrum Analyzer"

Measurement names to create:

<table>
<thead>
<tr>
<th>Meas Class</th>
<th>Measurement Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Standard&quot;</td>
<td>&quot;S11&quot;, &quot;S21&quot;, and so forth</td>
<td>S-parameter name</td>
</tr>
<tr>
<td></td>
<td>&quot;A_1&quot;, &quot;A_2&quot;, and so forth</td>
<td>Unratioed parameter names with notation: &quot;receiver_source port&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See <a href="#">Balanced S-parameter measurement names</a></td>
</tr>
<tr>
<td>Port 1 is the Source Port (DUT input). Port 3 or Port 2 can be chosen as the output of the DUT.</td>
<td>Learn about Active Hot parameters</td>
<td></td>
</tr>
</tbody>
</table>
### Active Hot Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;HotS11&quot;</td>
</tr>
<tr>
<td>&quot;HotS31&quot;</td>
</tr>
<tr>
<td>&quot;HotS13&quot;</td>
</tr>
<tr>
<td>&quot;HotS33&quot;</td>
</tr>
<tr>
<td>&quot;IPwr&quot;</td>
</tr>
<tr>
<td>&quot;OPwr&quot;</td>
</tr>
<tr>
<td>&quot;Gamma&quot;</td>
</tr>
<tr>
<td>&quot;Pmax&quot;</td>
</tr>
<tr>
<td>&quot;Xs(3,3)&quot;</td>
</tr>
<tr>
<td>&quot;Xt(3,3)&quot;</td>
</tr>
<tr>
<td>&quot;Xf(3,1)&quot;</td>
</tr>
<tr>
<td>&quot;DeltaOPwr&quot;</td>
</tr>
</tbody>
</table>

### Vector Mixer/Converter

**For output port Y (input port must be 1):**

- "S11"
- "VCY1"
- "SYY"

**Learn about VMC parameters**

**Note:** Input and output ports are set up using the Mixer Setup dialog. If the ports are not set up using the Mixer Setup dialog, then ports 1 and 2 are the default input and output ports and the only ports that can be used.

### Scalar Mixer

**For input port X and output port Y:**

- "SCXY"
- "SCYX"
- "SXX"

**Learn about SMC parameters**

**Note:** Input and output ports are set up using the Mixer Setup dialog. If the ports are not set up using the Mixer Setup dialog, then ports 1 and 2 are the default input and output ports and the only ports that can be used.
### Gain Compression

- "Gain Compression" Learn more
- "Gain Compression Converters" Learn more

### GCA and GCX:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;CompIn21&quot;</td>
<td>Input power at the compression point.</td>
</tr>
<tr>
<td>&quot;CompOut21&quot;</td>
<td>Output power at the compression point.</td>
</tr>
<tr>
<td>&quot;CompGain21&quot;</td>
<td>Gain at the compression point.</td>
</tr>
<tr>
<td>&quot;CompS11&quot;</td>
<td>Input Match at the compression point</td>
</tr>
<tr>
<td>&quot;RefS21&quot;</td>
<td>Linear Gain</td>
</tr>
<tr>
<td>&quot;DeltaGain21&quot;</td>
<td>CompGain21 - Linear Gain</td>
</tr>
<tr>
<td>&quot;S11&quot;, &quot;S21&quot;, &quot;S12&quot;, &quot;S22&quot;</td>
<td>Standard S-parameters; measured at port 1 and port 2</td>
</tr>
</tbody>
</table>

### GCX - All Gain Compression parameters (except S21 and S12) plus the following:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;S11&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;SC21&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;SC12&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;S22&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;Ipwr&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;RevIPwr&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;Opwr&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;RevOPwr&quot;</td>
<td></td>
</tr>
</tbody>
</table>

### Modulation Distortion and Modulation Distortion Converters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;PIn1&quot;</td>
<td>Power In</td>
</tr>
<tr>
<td>&quot;POut1&quot;</td>
<td>Reflected Power In</td>
</tr>
<tr>
<td>&quot;POut2&quot;</td>
<td>Power Out</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>PModFile</td>
<td>Power of modulation file</td>
</tr>
<tr>
<td>MSig2</td>
<td>Modulation Signal Out</td>
</tr>
<tr>
<td>MDist2</td>
<td>Modulation Distortion Out</td>
</tr>
<tr>
<td>MDistIR2</td>
<td>Modulation Distortion Input-referred</td>
</tr>
<tr>
<td>MGain21</td>
<td>Modulation Gain</td>
</tr>
<tr>
<td>MComp21</td>
<td>Modulation Compression</td>
</tr>
<tr>
<td>PGain21</td>
<td>Power Gain</td>
</tr>
<tr>
<td>LMatch2</td>
<td>Load match of VNA port</td>
</tr>
<tr>
<td>CarrIn1</td>
<td>Input Band Power</td>
</tr>
<tr>
<td>CarrOut2</td>
<td>Output Band Power</td>
</tr>
<tr>
<td>CarrGain21</td>
<td>Band Power Gain</td>
</tr>
<tr>
<td>NPRIn1</td>
<td>NPR at Input</td>
</tr>
<tr>
<td>NPROut2</td>
<td>NPR at Output</td>
</tr>
<tr>
<td>NPRDist21</td>
<td>NPR Distortion, Added by DUT</td>
</tr>
<tr>
<td>NPRPwrIn1</td>
<td>NPR Input Power</td>
</tr>
<tr>
<td>NPRPwrOut2</td>
<td>NPR Output Power</td>
</tr>
<tr>
<td>ACPIn1</td>
<td>ACP at input</td>
</tr>
<tr>
<td>ACPOut2</td>
<td>ACP at output</td>
</tr>
<tr>
<td>ACPDist21</td>
<td>ACP distortion, Added by DUT</td>
</tr>
<tr>
<td>ACPPwrIn1</td>
<td>ACP Input Power</td>
</tr>
<tr>
<td>ACPPwrOut2</td>
<td>ACP Output Power</td>
</tr>
<tr>
<td>EVMDistEq21</td>
<td>EVM Equalized Distortion, Added by DUT</td>
</tr>
<tr>
<td>EVMDistUn21</td>
<td>EVM Unequalized Distortion, Added by DUT</td>
</tr>
<tr>
<td>EVMPwrIn1</td>
<td>EVM Input Power</td>
</tr>
<tr>
<td>EVMPwrOut2</td>
<td>EVM Output Power</td>
</tr>
<tr>
<td>ModFilter</td>
<td>Measurement Modulation Filter</td>
</tr>
<tr>
<td>A, b1</td>
<td>Port 1 test port receiver</td>
</tr>
<tr>
<td>B, b2</td>
<td>Port 2 test port receiver</td>
</tr>
<tr>
<td>C, b3</td>
<td>Port 3 test port receiver</td>
</tr>
<tr>
<td>D, b4</td>
<td>Port 4 test port receiver</td>
</tr>
<tr>
<td>R1, a1</td>
<td>Port 1 reference receiver</td>
</tr>
<tr>
<td>R2, a2</td>
<td>Port 2 reference receiver</td>
</tr>
</tbody>
</table>
"R3", "a3"  Port 3 reference receiver
"R4", "a4"  Port 4 reference receiver

**Modulation Distortion ONLY - NOT Modulation Distortion Converters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;S11&quot;</td>
<td>Linear Input Match</td>
</tr>
<tr>
<td>&quot;S21&quot;</td>
<td>Linear Gain</td>
</tr>
<tr>
<td>&quot;LPIn1&quot;</td>
<td>Linear Input Match</td>
</tr>
<tr>
<td>&quot;LPOut1&quot;</td>
<td>Linear Reflected Power In</td>
</tr>
<tr>
<td>&quot;LPOut2&quot;</td>
<td>Linear Power Out</td>
</tr>
</tbody>
</table>

**Noise Figure AND NFX:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;NF&quot;</td>
<td>Noise figure</td>
</tr>
<tr>
<td>&quot;ENR&quot;</td>
<td>Validate noise source measurements.</td>
</tr>
<tr>
<td>&quot;T-Eff&quot;</td>
<td>Effective noise temperature.</td>
</tr>
<tr>
<td>&quot;DUTRNP&quot;</td>
<td>DUT noise power ratio. (Noise power expressed in Kelvin divided by 290).</td>
</tr>
<tr>
<td>&quot;DUTRNPI&quot;</td>
<td>System noise power ratio</td>
</tr>
<tr>
<td>&quot;SYSRNP&quot;</td>
<td>System noise power ratio</td>
</tr>
<tr>
<td>&quot;SYSRNPI&quot;</td>
<td>System noise power ratio</td>
</tr>
<tr>
<td>&quot;DUTNPD&quot;</td>
<td>DUT noise power density. (Noise power expressed in dBm/Hz).</td>
</tr>
<tr>
<td>&quot;DUTNPDI&quot;</td>
<td>System noise power density.</td>
</tr>
<tr>
<td>&quot;SYSNPD&quot;</td>
<td>System noise power density.</td>
</tr>
<tr>
<td>&quot;SYSNPDI&quot;</td>
<td>System noise power density.</td>
</tr>
<tr>
<td>&quot;OvrRng&quot;</td>
<td>Indication that the noise receiver is being over powered. (Opt 029 Only)</td>
</tr>
<tr>
<td>&quot;T-Rcvr&quot;</td>
<td>Temperature reading (in Kelvin) of the noise receiver board. (Opt 029 Only)</td>
</tr>
</tbody>
</table>

**Noise Figure ONLY - NOT NFX:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;S11&quot;, &quot;S21&quot;, &quot;S12&quot;, &quot;S22&quot;</td>
<td>Standard S-parameters; measured with the port1 and port2 noise switches set for noise</td>
<td></td>
</tr>
</tbody>
</table>
"Noise Figure Converters"

Learn more

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;A_1&quot;, &quot;A_2&quot;, ...</td>
<td>Unratioed parameters; with notation: &quot;receiver, source port&quot;</td>
</tr>
<tr>
<td>&quot;GammaOpt&quot;</td>
<td>Optimum Complex Reflection Coefficient</td>
</tr>
<tr>
<td>&quot;Rn&quot;</td>
<td>Noise Resistance</td>
</tr>
<tr>
<td>&quot;NFMin&quot;</td>
<td>Minimum noise figure that occurs at GammaOpt</td>
</tr>
</tbody>
</table>

**NFX ONLY:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;S11&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;SC21&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;SC12&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;S22&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;Ipwr&quot;</td>
<td>Mixer parameters</td>
</tr>
<tr>
<td>&quot;RevIPwr&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;Opwr&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;RevOPwr&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;ALO1&quot;, &quot;BLO1&quot;, ...</td>
<td>Test port receiver at LO1 frequency</td>
</tr>
<tr>
<td></td>
<td>Unratioed parameters with notation: &quot;receiver_source port&quot;</td>
</tr>
</tbody>
</table>

**Phase Noise**

Learn more

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PN</td>
<td>Phase Noise</td>
</tr>
<tr>
<td>AM</td>
<td>AM Noise</td>
</tr>
</tbody>
</table>

There are over 150 possible Swept IMD parameters, too many to list here.

Build the parameters with the Swept IMD Parameter dialog, then copy the parameter name to the remote command.

The following are a few example parameters:
| **"Swept IMD"** | **"PwrMainLo"** | Absolute power of the Low tone at the DUT output. |
| "Swept IMD Converters" | **"IM3"** | Power of the third product relative to the average power of the f1 and f2 tones measured at the DUT output. |
| Learn more | **"OIP3"** | Theoretical power level at which the third product will be the same power level as the average of the main tones at the output of the DUT. |
| **"IM Spectrum"** | **"Output"** | View signals OUT of the DUT and into VNA port 2 (B receiver). |
| Learn more | **"Input"** | View signals IN to the DUT (R1 receiver). |
| **"Reflection"** | **"Output"** | View signals reflected off the DUT input and back into VNA port 1 (A receiver). |
| **"IMx Spectrum Converters"** | **"Output"** | View signals OUT of the DUT and into VNA port 2 (B receiver). |
| Learn more | | |
| **"Differential I/Q"** | Create custom parameters using Sens:DIQ:Par:Def, then specify your custom parameter name here. |
| Learn more | The following are default parameters: |
| **"Input Power over F1 range"** | **"IPwrF1"** | Input Power over F1 range |
| **"Output Power over F1 range"** | **"OPwrF1"** | Output Power over F1 range |
| **"Gain over F1 range"** | **"GainF1"** | Gain over F1 range |
"Spectrum Analyzer"

Learn more

"a<n>"

Reference receiver

"b<n>"

Test port receiver

where <n> is the port number to measure

"ImageReject<n>"

where <n> is the image reject trace

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:CUST:DEF 'My VC21', 'Vector Mixer/Converter','S22'</td>
<td>Calculate a custom definition for a vector mixer/converter measurement</td>
</tr>
<tr>
<td>calculate2:custom:define 'MyNF', 'NoiseFigure', 'NF'</td>
<td>Define a noise figure measurement</td>
</tr>
<tr>
<td>CALC1:CUST:DEF 'MyAM', 'Phase Noise', 'AM' 'Defines an AM Noise measurement but doesn't display.'</td>
<td>Define an AM noise measurement</td>
</tr>
<tr>
<td>DISP:MEAS:FEED 1 'Displays AM Noise measurement in window number 1.'</td>
<td>Display the AM noise measurement in a specific window</td>
</tr>
</tbody>
</table>

Query Syntax

Not applicable

Overlapped?

No

Default

Not applicable

CALCulate<cnum>:CUSTom:MODify <param>

Applicable Models: All

(Write-only) Changes the selected custom measurement to a different parameter. This is dependent upon the configurations and options.

See an example using this command for a VMC and SMC measurement

Parameters

<cnum> Channel of the custom measurement to be changed. First, select the measurement using CALC:PAR:SEL.

@param Parameter to change the custom measurement to. Select a parameter that is valid for the type of measurement. Choose from the same arguments as Calc:Cust:Def.

Examples

SYST:PRES

CALC2:CUST:DEF 'My VC21', 'Vector Mixer/Converter'
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:PAR:SEL 'My VC21'</td>
<td></td>
</tr>
<tr>
<td>CALC2:CUST:MOD 'S22'</td>
<td></td>
</tr>
<tr>
<td>CALC:CUST:DEF 'MyPN','Phase Noise','PN'</td>
<td>Defines a Phase Noise measurement but doesn't display.</td>
</tr>
<tr>
<td>DISP:WIND1:TRAC1:FEED 'MyPN'</td>
<td>Displays Phase Noise measurement in window number 1.</td>
</tr>
<tr>
<td>CALC:PAR:SEL 'MyPN'</td>
<td>Selects the 'MyPN' measurement.</td>
</tr>
<tr>
<td>CALC:CUST:MOD 'AM'</td>
<td>Changes the Phase Noise parameter to AM.</td>
</tr>
</tbody>
</table>

**Query Syntax**

- **Overlapped?**
  - No

- **Default**
  - Not applicable
Calculate:Data Commands

Controls writing and reading VNA measurement data.

These commands are **Superseded** by the `CALCulate:MEASure:DATA` commands.

<table>
<thead>
<tr>
<th>CALCulate:DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTom</td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>MFData?</td>
</tr>
<tr>
<td>MSData?</td>
</tr>
<tr>
<td>SNP?</td>
</tr>
<tr>
<td>PORTs?</td>
</tr>
<tr>
<td>SAVE</td>
</tr>
</tbody>
</table>

Click on a red keyword to view the command details.

Red is a superseded command.

**See Also**

- Example Programs
- Data Access Map
- Synchronizing the Analyzer and Controller
- To read receiver data, use `CALC:RDATA?`
- To read error terms, use `SENS:CORR:CSET:DATA`
- To read SnP measurement data, use `CALC:DATA:SNP?`
- SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using `Calc:Par:MNUM` or `Calc:Par:Select`. Learn more.

(Write) `CALCulate<cnum>:DATA <char>,<data>`

(Read) `CALCulate<cnum>:DATA? <char>`
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Reads or writes Measurement data, Memory data, or Normalization Divisor data from the Data Access Map location.

- For Measurement data, use FDATA, RDATA, or SDATA
- For Memory data, use FMEM or SMEM. When querying memory, you must first store a trace into memory using CALC:MATH:MEMorize.
- For Normalization Divisor (Receiver Power Cal error term) data, use SDIV
- Use FORMat:DATA to change the data type (<REAL,32>, <REAL,64> or <ASCii,0>).
- Use FORMat:BORDer to change the byte order. Use “NORMal” when transferring a binary block from LabView or Vee. For other programming languages, you may need to “SWAP” the byte order.

Equation Editor Notes:

- When equation editor is active on a trace in a standard S-parameter channel, Calc:Data returns the data from the parameter on the trace that was measured last. For example, for the equation "S22 + S33 + S11", then S33 is the last measured parameter because it uses source port 3.
- In applications, if equation editor is active and the original parameter for the trace is not requested anywhere in the channel, then zeros are returned. If the original parameter is being measured within the channel, then data for the original parameter is returned.
- In general, if an equation contains no measurement parameters, then data for the original parameter is returned.

Note: The Calc:Data SCORR command to read / write error terms is Superseded with SENS:CORR:CSET:DATA. SCORR commands do NOT accommodate greater than 12 error terms.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>` FDATA Formatted measurement data to or from Data Access Map location Display (access point 2).

Note: When querying FDATA, data is received in degrees. When setting phase using FDATA, the command expects the data in radians.

- Corrected data is returned when correction is ON.
- Uncorrected data is returned when correction is OFF.
- Returns TWO numbers per data point for Polar and Smith Chart format.
- Returns one number per data point for all other formats.
- Format of the read data is same as the displayed format.

**RDATA**  Complex measurement data.

**Writes** data to Data Access Map location **Raw Measurement** (access point 0), same as SDATA.

**Note:** Write access is forbidden on the standard channel if factory calibration is on (applies to ENA/PXI/Streamline, not to PNA); to disable factory calibration use `SYST:FCOR:CHAN:COUP:STAT OFF` command.

**Reads** data from Data Access Map location **Raw Measurement** (access point 0).

- Returns TWO numbers per data point.
- Returned numbers are uncorrected (regardless of correction state)

**SDATA**  Complex measurement data.

**Writes** data to Data Access Map location **Raw Measurement** (access point 0).

**Note:** Write access is forbidden on the standard channel if factory calibration is on (applies to ENA/PXI/Streamline, not to PNA); to disable factory calibration use `SYST:FCOR:CHAN:COUP:STAT OFF` command.

**Reads** data from **Apply Error Terms** (access point 1).

**Note:** Writing data to this access point, and then reading the data from this access point, will result in different values if correction or factory cal is ON.

- Returns TWO numbers per data point.
- Corrected data is returned when correction is ON.
- Uncorrected data is returned when correction is OFF.

**FMEM**  Formatted memory data to or from Data Access Map location **Memory result** (access point 4).

- Returns TWO numbers per data point for Polar and Smith Chart format.
- Returns one number per data point for all other formats.
- Format of the read data is same as the displayed format.
- Returned data reflects the correction level (On|OFF) when the data was stored into memory.

**SMEM**  Complex measurement data to or from Data Access Map location Memory (access point 3).

- Returns TWO numbers per data point.
- Returned data reflects the correction level (On|OFF) when the data was stored into memory.
- Returned data reflects the correction level (On|OFF) when the data was stored into memory.

**SDIV**  Complex data from Data Access Map location Normalization (5).

- Returns TWO numbers per data point.
- If normalization interpolation is ON and the number of points changes after the initial normalization, the divisor data will then be interpolated.
- When querying the normalization divisor, you must first store a divisor trace using CALC:NORMalize[:IMMediate].

The following Calc:Data SCORR command to read / write error terms is Superseded with SENS:CORR:CSET:DATA. These SCORR commands do NOT accommodate greater than 12 error terms.

<table>
<thead>
<tr>
<th>2-Port SOLT and TRL calibrations</th>
<th>Specify this &lt;char&gt; to get or put this Error Term...</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCORR1</td>
<td>Forward Directivity</td>
</tr>
<tr>
<td>SCORR2</td>
<td>Forward Source Match</td>
</tr>
<tr>
<td>SCORR3</td>
<td>Forward Reflection Tracking</td>
</tr>
<tr>
<td>SCORR4</td>
<td>Forward Isolation</td>
</tr>
<tr>
<td>SCORR5</td>
<td>Forward Load Match</td>
</tr>
<tr>
<td>SCORR6</td>
<td>Forward Transmission Tracking</td>
</tr>
<tr>
<td>SCORR7</td>
<td>Reverse Directivity</td>
</tr>
<tr>
<td>SCORR8</td>
<td>Reverse Source Match</td>
</tr>
<tr>
<td>SCORR9</td>
<td>Reverse Reflection Tracking</td>
</tr>
<tr>
<td>SCORR10</td>
<td>Reverse Isolation</td>
</tr>
<tr>
<td>SCORR11</td>
<td>Reverse Load Match</td>
</tr>
<tr>
<td>SCORR12</td>
<td>Reverse Transmission Tracking</td>
</tr>
</tbody>
</table>
EXAMPLE

CALC:DATA FDATA, Data(x)
calculate2: data sdata, data(r, i)

See another example using this command.

Return Type: Block data

Default - Not Applicable

CALCulate<cnum>:DATA:CUSTom <name>, <data> Superseded

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Note: This command has been replaced by CALC:DATA: which can now be used with all VNA applications.

(Read-Write) Reads or writes data from a custom-named measurement buffer.

See Critical Note

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <name> Name of the buffer to be read or written
- <data> Data to be read or written to the custom buffer. Format as one number per data point.

Examples

CALC:DATA:CUST 'VectorResult0', 0, 1, 2, 3, 4, 5 'Write
CALC:DATA:CUST? 'VectorResult0' 'Read

Query Syntax

CALCulate:DATA:CUSTom? <name>

Return Type

Depends on Form: Data

Default Not Applicable

CALCulate<cnum>:DATA:CUSTom:CATalog? Superseded
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Note: This command has been replaced by `CALC:DATA:CAT` which can now be used with all VNA applications.

(Read-only) Reads the list of buffer names (comma separated list of string values) available from the selected parameter. Specify the measurement using `CALCulate:PARameter:SELection`.

See Critical Note

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
</tbody>
</table>

**Examples**

```
CALC:DATA:CUST:CAT?
calculate: data: custom: catalog?
```

**Return Type** String

**Default** Not Applicable

**CALCulate<cnum>:DATA:MFD? <measList>**

Applicable Models: N522xB, N523xB, N524xB, E5080A, M948xA, P937xA

(Read-only) Gets the formatted data array of multiple traces (traces-n, m, .... to l) of the selected channel.

This command gets multiple trace data with one command, while `CALC:MEAS:DATA:FDAT` returns only one trace with one command.

Note: If valid data is not calculated because of the invalid measurement, “1.#QNB” is read out.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
<tr>
<td>&lt;measList&gt;</td>
<td>Trace number. &quot;n, m, l, ...&quot; where n, m, l are 1 to the maximum trace number.</td>
</tr>
</tbody>
</table>

Note: Use comma for separator of trace number.

**Examples**

```
CALC:DATA:MFD? "1,2"
```

**Return Type** Data Block

**Default** Not Applicable
CALCulate<cnum>:DATA<data>:MSData? <measList>

**Applicable Models:** N522xB, N523xB, N524xB, E5080A, M948xA, P937xA

*(Read-only)* Gets the corrected data array of multiple traces (traces-n, m, ..., to l) of the selected channel.

This command is allows to get several corrected data with one command, while CALC:MEAS:DATA:SDAT returns only one corrected data with one command.

**Note:** If valid data is not calculated because of the invalid measurement, “1.#QNB” is read out.

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<measList>`: Trace number. "n, m, l,..." where n, m, l are 1 to the maximum trace number.

**Examples**

```
CALC:DATA:MSD? "1,2"
```

**Return Type**

- **Data Block**
- **Default**
  - Not Applicable

---

CALCulate<cnum>:DATA:SNP? <n> **Superseded**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

**Note:** This command has been replaced by CALC:DATA:SNP:PORTs?

*(Read-only)* Reads SnP data from the selected measurement. Learn more about SnP data.

This command is valid ONLY with standard S-parameter measurements.

**Notes**

- This command returns SNP data without header information, and in columns, not in rows as .SnP files. This means that the data returned from this command sends all frequency data, then all Sx1 magnitude or real data, then all Sx1 phase or imaginary data, and so forth.

- To avoid frequency rounding errors, specify FORM:DATA <Real,64> or <ASCii, 0>

**See Critical Note**

**Parameters**
<cnum>  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<n>   Amount of data to return. If unspecified, <n> is set to 2. The number you specify must be less than or equal to the number of available ports on the VNA.

Choose from:

1 (S1P) returns 1-Port data for the active measurement if the active measurement is a reflection parameter such as S11 or S22. The behavior is UNDEFINED if the active measurement is a transmission parameter such as an S21.

2 (S2P) returns data for the four 2 port parameters associated with the current measurement. Default. Data that is not available is zero-filled.

3 (S3P) returns data for the nine 3 port parameters associated with the current measurement. Data that is not available is zero-filled.

4 (S4P) returns data for the sixteen 4 port parameters associated with the current measurement. Data that is not available is zero-filled.

SnP data can be output using several data formatting options. See MMEM:STOR<file>.<snp>

See also MMEM:STOR <file>.<snp>

Examples

CALC:PAR:DEF MyMeasurement, S11
CALC:PAR:SEL MyMeasurement
CALC:DATA:SNP? 1

Return Type

Default: Not Applicable

CALCulate<cnum>:DATA:SNP:PORTs? <"x,y,z">[, FAST]

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Note: This command replaces CALC:DATA:SNP?. This command is more explicit regarding the data to be returned, and works for VNAs with multiport test sets.

(Read-only) Reads SNP data from the selected measurement for the specified ports. Learn more about SNP data.
This command is valid **ONLY** with standard S-parameter measurements.

**Notes**

- This command returns SNP data without header information, and in columns, not in rows as .SnP files. This means that the data returned from this command sends all frequency data, then all Sx1 magnitude or real data, then all Sx1 phase or imaginary data, and so forth.
- To avoid frequency rounding errors, specify FORM:DATA <Real,64> or <ASCii, 0>
- Data that is not available is zero-filled.
- For sweeps with a large number of data points, always follow this command with *OPC?. Learn more.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<"x,y,z">` Comma or space delimited port numbers for which data is requested, enclosed in quotes.

SNP data can be output using several data formatting options. See MMEM:STOR:e:TRACe:FORMat:SNP.

[, FAST] Reduce the saving time

:SENS:CORR:CACH:MODE should be set at ON.

The correction must cover all the ports of the SNP port list.

The active measurement must be a corrected S Parameter defined by the port list for the SNP requests. EG: "S33" is not a proper selected measurement with CALC:DATA:SNP:PORTS? <1,2>

**Examples**

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depends on FORMat:DATA</td>
<td></td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:DATA:SNP:PORTs:SAVE <"x,y,z">,<"filename">[, FAST]**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

**Note:** This command replaces MMEM:STOR sNp for standard channels (only). This command is more explicit regarding the data to be saved, and works for VNAs with multiport test sets.
(Write-only) Saves SNP data from the selected measurement for the specified ports. Learn more about SNP data.

- The Normal vs Mixed Mode selection is NOT used as it is in the Choose Ports dialog. Instead, data is returned as it is displayed on the trace. If the selected measurement is Mixed Mode (balanced), then balanced data is returned. If the selected measurement is an S-parameter, then S-parameter data is returned.

- This command is valid ONLY with the Standard measurement class (NOT applications).

- Data that is not available is zero-filled.

- For sweeps with a large number of data points, always follow this command with *OPC? Learn more.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

- `<"x,y,z">` *String* Comma or space delimited port numbers for which data is requested, enclosed in quotes.

- `<filename>` *String* Path, filename, and suffix of location to store the SNP data, enclosed in quotes. The suffix is not checked for accuracy. If saving 2 ports, specify "filename.s2p"; If saving 4 ports, specify "filename.s4p.", and so forth.

SNP data can be output using several data formatting options. See MMEM:STORe:TRACe:FORMat:SNP.

[, FAST] Reduce the saving time

:SENS:CORR:CACH:MODE should be set at ON.

The correction must cover all the ports of the SNP port list.

The active measurement must be a corrected S Parameter defined by the port list for the SNP requests. EG: "S33" is not a proper selected measurement with CALC:DATA:SNP:PORTS? <1,2>

### Examples

CALC:DATA:SNP:PORTs:Save '1,2,4','D:\MyData.s3p';*OPC?

### Return Type

- **Default** Not Applicable

### CALCulate<cnum>:MEASURE<nunum>:RDATa? <char>
(Read-only) Returns receiver data for the selected measurement. To query measurement data, see CALC:DATA?

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mnum>` Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, `<mnum>` is set to 1.
- `<char>` Choose from any physical receiver in the VNA.

For example: "A"

Also, **REF** - returns data for either R1 or R2 data depending on the source port of the selected measurement.

See the **block diagram** showing the receivers in VNA.

**Note:** Logical receiver notation is NOT allowed with this command. Learn more.

**Example**

```
INITiate:CONTinuous OFF
INITiate:IMMediate;*wai
CALC:MEAS:RDATA? A
CALCulate:RDATA? REF
```

**Return Type**

- **Default** Not Applicable

**Notes:**

Generally when you query the analyzer for data, you expect that the number of data values returned will be consistent with the number of points in the sweep.

However, if you query receiver data while the instrument is sweeping, the returned values may contain zeros. For example, if your request for receiver data is handled on the 45th point of a 201 point sweep, the first 45 values will be valid data, and the remainder will contain complex zero.

This can be avoided by synchronizing this request with the end of a sweep or putting the channel in hold mode.

Learn about Unratioed Measurements
**CALCulate<cnum>:DTOPoIogy <device>,<topology>**

**Applicable Models:** Multi-port systems with > 4 ports

*(Read-Write)* Maps the physical VNA ports to a device of balanced and single-ended logical ports for multi-port systems with greater than 4 ports. The device type is selected using `CALCulate:FSIMulator:BAlun:DEViCe`.

**See Also:**

`CALC:FSIM:BAL:PAR:CAT?` - returns the list of measurement parameters available for the currently selected topology.

`CALC:FSIM:BAL:PAR:CUST:DEFine` and `CALC:MEAS:PAR` - defines measurement parameter corresponding to a custom topology for systems where the port count is expandable beyond 4 ports.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<device>` (String) Device type for the balanced measurement. ‘B’ means the Balanced port; ‘S’ means the Single-ended port. Choose from:
  - B – 1 port balanced device (2 ports)
  - BB – Balanced - Balanced device (4 ports)
  - BS – Balanced - Single-ended device (3 ports)
  - BSS – Balanced - Single-ended - Single-ended device (3 ports)
  - SB – Single-ended - Balanced device (3 ports)
  - SSB – Single-ended - Single-ended - Balanced device (4 ports)
- `<topology>` (Int array) Physical port numbers mapped to the logical ports, separated by ‘,’.
  - ‘B’ (Balanced) requires 2 physical port numbers: `<nPos>`, `<nNeg>`.
  - ‘S’ (Single-ended) requires 1 physical port number.

**Examples**

'The following example sets up 6 physical ports into 5 logical ports:

'Logical port 1 is a single ended port mapped to physical port 1
'Logical port 2 is a single ended port mapped to physical port 2
'Logical port 3 is a balanced port mapped to physical ports 4 and 5
'Logical port 4 is a single ended port mapped to physical port 3
Logical port 5 is a single ended port mapped to physical port 6

Example 1
CALC:FSIM:BAL:DEV CUST
CALC:DTOP "SSBSS",1,2,4,5,3,6
CALC:MEAS:PAR "SDD33"

Example 2
CALC:PAR:COUN 1
CALC:FSIM:BAL:DEV CUST
CALC:FSIM:BAL:PAR:STATE ON
CALC:DTOPology "SSBSS",1,2,4,5,3,6
CALC:FSIM:BAL:PAR:CUST:DEF "SDD33"

Query Syntax  CALCulate<enum>:DTOPology <device>,<topology>?
Return Type  Int array
Default  Not Applicable
Calculate:Equation Commands

Controls Equation Editor capabilities.

```
CALCulate:EQUation:
  LIBRary
    | FUNCTions
    | IMPort?
    | REMove
  STATe
  TEXT
  VALid?
```

Click on a keyword to view the command details.

**see Also**

- Example Programs
- Learn about Equation Editor
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

```
CALCulate:EQUation:LIBRary:FUNCTIONs <string>
```
Applicable Models: All

(Read-only) Returns the functions in the specified DLL.

Parameters

<string> Full path and filename of the *.dll to be read.

Examples

```
functions = CALC:EQU:LIBR:FUNC "C:\Program Files(x86)\Keysight\Network Analyzer\UserFunctions\Expansion.dll"
```

Query Syntax

CALCulate:EQUation:LIBRary:FUNCtions?

Return Type

Comma delimited string of function names.

Default

Not Applicable

---

CALCulate:EQUation:LIBRary:IMPort <string>

Applicable Models: All

(Read-Write) Imports the functions in the specified DLL and returns whether the functions have been imported into the VNA.

Parameters

<string> Full path and filename of the *.dll.

Examples

```
'Command - Imports functions

CALC:EQU:LIBR:IMPort "C:\Program Files(x86)\Keysight\Network Analyzer\UserFunctions\Expansion.dll"

'Query if Imported successfully

success = CALC:EQU:LIBR:IMPort? "C:\Program Files(x86)\Keysight\Network Analyzer\UserFunctions\Expansion.dll"

1   'Returns 1 if successfully imported
```

Query Syntax

CALCulate:EQUation:LIBRary:IMPort?

Returns the following:

1 - Imported

0 - NOT imported

Return Type

Boolean
**CALCulate:EQUation:LIBRary:REMove <string>**

**Applicable Models:** All

*(Write-only)* Removes an imported an Equation Editor DLL from the VNA.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;string&gt;</td>
<td>Full path and filename of the *.dll.</td>
</tr>
</tbody>
</table>

**Examples**

```
CALC:EQU:LIBR:REM "C:\Program Files(x86)\Keysight\Network Analyzer\UserFunctions\Expansion.dll"
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**CALCulate<cnum>:EQUation[:STATe] <bool>**

**Applicable Models:** All

*(Read-Write)* Turns ON and OFF the equation on selected measurement for the specified channel. If the equation is not valid, then processing is not performed. Use `CALC:EQUation:VAlid?` to ensure that the equation is valid.

**See Critical Note**

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
<tr>
<td>&lt;bool&gt;</td>
<td>ON (or 1) - turns equation ON. OFF (or 0) - turns equation OFF.</td>
</tr>
</tbody>
</table>

**Examples**

```
CALC:EQU 1
```

```
calculate2:equation:state 0
```

**Query Syntax**

`CALCulate<cnum>:EQUation[:STATe]?`

**Return Type**

Boolean

**Default**

OFF (0)
CALCulate<cnum>:EQUation:TEXT <string>

Applicable Models: All

(Read-Write) Specifies an equation or expression to be used on the selected measurement for the specified channel.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<string> Any valid equation or expression. See Equation Editor.

Examples

'Equation (includes '=')

CALC:EQU:TEXT "foo=S11/S21"

'Expression

calculate2:equation:text "S11/S21"

Query Syntax
CALCulate<cnum>:EQUation:TEXT?

Return Type
String

Default
Not Applicable

CALCulate<cnum>:EQUation:VALid?

Applicable Models: All

(Read-Only) Returns a boolean value to indicate if the current equation on the selected measurement for the specified channel is valid. For equation processing to occur, the equation must be valid and ON (CALC:EQU:STAT 1).

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

calculate2:equation:valid?
<table>
<thead>
<tr>
<th><strong>Return Type</strong></th>
<th>Boolean</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - equation is valid</td>
<td></td>
</tr>
<tr>
<td>0 - equation is NOT valid</td>
<td></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
Calculate:Filter Commands

Controls the gating function used in time domain measurements. The gated range is specified with either (start / stop) or (center / span) commands.

These commands are **Superseded** by the `CALCulate:MEASure:FILTTer` commands.

![CALCulate:Filter Command Diagram](image)

Click on a keyword to view the command details.

**see Also**

- Example Programs
- Learn about Gating
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**Critical Note:** `CALCulate` commands act on the selected measurement. You can select one measurement for each channel using `Calc:Par:MNUM` or `Calc:Par:Select`. Learn more.

`CALCulate<cnum>:FILTer[:GATE]:COUPle:PARameters <num>`
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Specifies the time domain gating parameters to be coupled. The settings for those parameters will be copied from the selected measurement to all other measurements on the channel.

- To enable Trace Coupling, use SENS:COUP:PAR
- To specify Transform parameters to couple, use CALC:TRAN:COUP:PAR

Learn more about Time Domain Trace Coupling

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<num> (Numeric) Parameters to couple. To specify more than one parameter, add the numbers.

1 - Gating Stimulus (Start, Stop, Center, and Span TIME settings.)

2 - Gating State (ON / OFF)

4 - Gating Shape (Minimum, Normal, Wide, and Maximum)

8 - Gating Type (Bandpass and Notch)

Examples

'To couple all parameters:
CALC:FILT:COUP:PAR 15

'To couple Stimulus and Type:
calculate2:filter:gate:couple:parameters 9

Query Syntax

CALCulate<cnum>:FILTer[:GATE]:TIME:CENTer <num>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the gate filter center time.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Center time in seconds; Choose any number between:
  \[ \pm \frac{(\text{number of points}-1)}{\text{frequency span}} \]

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

- `CALC:FILT:GATE:TIME:CENT -5 ns`
- `calculate2:filter:time:center maximum`

Query Syntax

- `CALCulate<cnum>:FILTer[:GATE]:TIME:CENTer?`

Return Type

- Numeric
  - Default: 0

---

CALCulate<cnum>:FIltEr[:GATE]:TIME:SHAPE <char>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the gating filter shape when in time domain.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>` Choose from
  - `MAXimum` - the widest gate filter available
  - `WIDE` -
  - `NORMal` -
  - `MINimum` - the narrowest gate filter available

Examples

- `CALC:FILT:GATE:TIME:SHAPE MAX`
- `calculate2:filter:time:shape normal`

Query Syntax

- `CALCulate<cnum>:FIltEr[:GATE]:TIME:SHAPE?`

Return Type

- Character
CALCulate<nun>:FIILTer[:GATE]:TIME:SPAN <num>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the gate filter span time.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Time span in seconds; Choose any number between: 0 and 2* [(number of points-1) / frequency span]

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```
calc:fil:time:span 5 ns
```

```
calculate2:filter:time:span maximum
```

**Query Syntax**

CALCulate<nun>:FIILTer[:GATE]:TIME:SPAN?

**Return Type** Numeric

**Default** 20 ns

CALCulate<nun>:FIILTer[:GATE]:TIME:STATe <boolean>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Turns gating state ON or OFF.

**See Critical Note**

**Note:** Sweep type must be set to Linear Frequency in order to use Transform Gating.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<boolean>` ON (or 1) - turns gating ON.
  OFF (or 0) - turns gating OFF.

**Examples**

```
calc:filt:time:stat on
```

```
calculate2:filter:gate:time:state off
```

**Default** NORMal
CALCulate<cnum>:FILTER[:GATE]:TIME:START <num>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the gate filter start time.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Start time in seconds; any number between:
  
  ± (number of points-1) / frequency span

  **Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FILT:TIME:STAR 1e-8</td>
<td>Numeric</td>
</tr>
<tr>
<td>calculate2:filter:gate:time:start minimum</td>
<td>Numeric</td>
</tr>
</tbody>
</table>

CALCulate<cnum>:FILTER[:GATE]:TIME:STOP <num>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the gate filter stop time.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Stop time in seconds; any number between: ± (number of points-1) / frequency span

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

```
CALC:FILT:TIME:STOP -1 ns
calculate2:filter:gate:time:stop maximum
```

Query Syntax

```
CALCulate<cnum>:FILTer[:GATE]:TIME:STOP?
```

Return Type

Numeric

Default

10 ns

CALCulate<cnum>:FILTer[:GATE]:TIME[:TYPE] <char>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the type of gate filter used.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>` Choose from:
  - **BPAS** - Includes (passes) the range between the start and stop times.
  - **NOC** - Excludes (attenuates) the range between the start and stop times.

Examples

```
CALC:FILT:TIME BPAS
calculate2:filter:gate:time:type notch
```

Query Syntax

```
CALCulate<cnum>:FILTer[:GATE]:TIME[:TYPE]?
```

Return Type

Character

Default

BPAS
Calculate:Format Commands

These commands are **Superseded** by the CALCulate:MEASure:FORMat commands.

CALCulate:
| FORMat
| UNIT

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using `Calc:Par:MNUM` or `Calc:Par:Select`. Learn more.

**See Also**

- Example using this command.
- Learn About Data Format
- Synchronizing the Analyzer and Controller

CALCulate<cnum>:FORMat <char>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the display format for the measurement.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>` Choose from:
  
  - MLINear
  - MLOGarithmic
  - PHASE
  - UPHase 'Unwrapped phase
- IMAGinary
- REAL
- POLar
- SMITH
- SADMittance 'Smith Admittance
- SWR
- GDELay 'Group Delay
- KELVin
- FAHRenheit
- CELSius
- PPHase 'Positive Phase
- IMPedance
- VOLT
- COMPlex

<table>
<thead>
<tr>
<th>Examples</th>
<th>CALC:FORM MLIN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>calculate2:format polar</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>CALCulate&lt;cnum&gt;:FORMat?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Character</td>
</tr>
<tr>
<td>Default</td>
<td>MLOG</td>
</tr>
</tbody>
</table>

CALCulate<cnum>:FORMat:UNIT <dataFormat>, <units>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the units for the specified data format. Measurements with display formats other than those specified are not affected.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<dataFormat> Choose from:

- **MLOG** - Log magnitude
- **MLIN** - Linear magnitude

<units> For unratioed MLOG measurements, choose from:

- **DB** Units are displayed in decibels.
- **DBM** Units are displayed in dBm. 0 dBm = 0.001 watt
- **DBMV** Units are displayed in dBmV. 0 dBmV = 0.001 volt
  
  DBmV value depends on the reference impedance: dBmV = dBm + 30 + 10*\log_{10}(Z_0)

- **DBMA** Units are displayed in dBmA. 0 dBmA = 0.001 Ampere
- **DBUV** Units are displayed in dBuV. 0 dBuV = 1 uV
  
  DBuV value depends on the reference impedance: dBuV = dBm + 90 + 10*\log_{10}(Z_0)

For unratioed MLIN measurements, choose from:

- **U** - No units. Linear magnitude without conversion
- **W** - Units are displayed in Watts
- **V** - Units are displayed in Volts
- **A** - Units are displayed in Amperes

Examples

```
CALC:FORM:UNIT MLOG, DBM
```

```
calculate2:format:unit mlog,dbmv
```

Query Syntax
CALCulate<cnum>:FORMat:UNIT? <dataFormat>

Return Type Character

**Default** MLOG, DBM
Calculate:F Simulator Commands

Specifies settings and fixturing for Balanced Measurements.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:F SIMulator</td>
<td></td>
</tr>
<tr>
<td>Active More commands</td>
<td></td>
</tr>
<tr>
<td>BALun More commands</td>
<td></td>
</tr>
<tr>
<td>DRAFt More commands</td>
<td></td>
</tr>
<tr>
<td>EMBed More commands</td>
<td></td>
</tr>
<tr>
<td>GLOop More commands</td>
<td></td>
</tr>
<tr>
<td>LEGacy?</td>
<td></td>
</tr>
<tr>
<td>SENDed More commands</td>
<td></td>
</tr>
<tr>
<td>SNP:EXTRapolate</td>
<td></td>
</tr>
<tr>
<td>STATE</td>
<td></td>
</tr>
</tbody>
</table>

Click a keyword to view the command details.

See Also

- Example Programs
- SCPI Command Tree

CALCulate<cnum>:FSIMulator:LEGacy?
Applicable Models: All

(Read only) Returns TRUE if user is only using legacy SCPI fixture commands. If the user has used the GUI or new SCPI commands to create a fixture, it will return FALSE.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

Examples

- `CALC:FSIM:LEG?`
- `calculate2:fsimulator:legacy?`

Query Syntax

CALCulate<cnum>:FSIMulator:LEGacy?

Return Type

Boolean

Default

TRUE

CALCulate<cnum>:FSIMulator:SNP:EXTRapolate <bool> Superseded

Applicable Models: All

Note: This command is changed to CALCulate:FSIMulator:CIRCuIt:FILE:EXTrapolate Learn about Using Fixture Simulator.

(Read-Write) Turns ON and OFF SNP file extrapolation for both 2-port and 4-port embedding/de-embedding. Learn more.

Note: This command affects ALL measurements on the specified channel.

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<bool>` Choose from:

  - **ON or 1** - Turns Extrapolation ON
  - **OFF or 0** - Turns Extrapolation OFF

Examples

- `CALC:FSIM:SNP:EXTR 1`
- `calculate2:fsimulator:snp:extrapolate 0`

Query Syntax

CALCulate<cnum>:FSIMulator:SNP:EXTRapolate?

Return Type

Boolean

Default

OFF
CALCulate<cem>:FSIMulator:STATe <bool>

**Applicable Models:** All

(Read-Write) Turns all three fixturing functions (de-embedding, port matching, impedance conversion) ON or OFF for all ports on the specified channel. Does not affect port extensions.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**
- `<cem>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cem>` is set to 1.
- `<bool>` Choose from:
  - **ON or 1** - Turns Fixturing ON
  - **OFF or 0** - Turns Fixturing OFF

**Examples**
- `CALC:FSIM:STAT 1`
- `calculate2:fsimulator:state 0`

**Query Syntax**
- `CALCulate<cem>:FSIMulator:STATe?`

**Return Type**
- Boolean

**Default**
- OFF

---

CALCulate<cem>:DTOPology <device>, <topology>

**Applicable Models:** E5080A, M9485A
(Write-only) Defines the device type and the topology for a balanced measurement.

This command will replace the following commands:

CALC:F SIM: BAL: TOP: SBAL[:PPOR]
CALC:F SIM: BAL: TOP: SSB[:PPOR]
CALC:F SIM: BAL: TOP: BBAL[:PPOR]
CALC:F SIM: BAL: TOP: BALS[:PPOR]

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
<device> (String) Device type for the balanced measurement. ‘B’ means the Balanced port; ‘S’ means the Single-ended port. Choose from:

B – 1 port balanced device (2 ports)
BB – Balanced - Balanced device (4 ports)
BS – Balanced - Single-ended device (3 ports)
SB – Single-ended - Balanced device (3 ports)
SSB – Single-ended - Single-ended - Balanced device (4 ports)
<topology> (Int array) Physical port numbers mapped to the logical ports, separated by ‘,’.

‘B’ (Balanced) requires 2 physical port numbers: <nPos>, <nNeg>.

‘S’ (Single-ended) requires 1 physical port number.

Examples

CALC: DTOP "SB", 2, 1, 4
calculate:dtopology "SB", 2, 1, 4

Query Syntax

CALCulate<cnum>:MEASure<mnum>:PARameter:BALun[:STATe] <bool>

Return Type Not Applicable
Default Not Applicable
(Read-Write) Sets or gets a balanced measurement state.

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mnum>`: Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, `<mnum>` is set to 1.
- `<bool>`: ON or 1 - Balanced Transform ON. OFF or 0 - Balanced Transform OFF.

Examples

- `CALC2:MEAS3:PAR:BAL ON`
- `calculate2:measure:parameter:balun:state off`

Query Syntax

`CALCulate<cnum>:MEASure<mnum>:PARameter:BALun[:STATe]?

Return Type

Boolean

Default

OFF or 0
CALCulate:FSIMulator:DRAFt:
  | APPLY
  | CIRCuIt:
    | ADD
    | CATalog?
    | DELete
    | DEVeCe
      | PORTs
    | EMBED
    | TYPE
    | FILE
      | EXTRapolate
      | MODify
    | NEXT?
  | PARameter
    | C
    | C2
    | DELay
    | G
    | G2
    | L
    | L2
    | LOSS
      | VALue
    | R
    | R2
    | RIN
    | ROUT
    | Z0
    | RESet
    | STATE
    | VNA
      | PORTs
  | DISCard

| EXTension:PORT:
  | DELay
  | DISTance
    | VALue
  | UNIT
  | END
  | LOSS
| FREQuency | VALue | STATe | MEDium | WAVEguide | FCUToff | COUPle | VELOcity | FACTor | COUPle | DCLoss:VALue | STATe | SAVE | SECTion | CIRCuit | ENABLE | EXTension | ENABLE | ZCONversion | ENABLE | ZCONversion | DIFFerential | STATe | BPORt | COMPLex | SCALar | :LPORt | COMPLex | SCALar | COMMONmode
See Also

- Learn about Using Fixture Simulator

- Example Programs

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

- CALC:PAR:CAT? alone can NOT be used to return a balanced measurement parameter. If a balanced measurement transform is being performed, then additional querying of the CALC:FSIM system is required to determine the balanced parameter type. See an example.

- **BPORt versus LPORt commands** - For each command in this subsystem that includes a BPORt keyword, there is an LPORt equivalent. The commands are identical except for the way in which the balanced / logical port numbers are specified:
  
  - The BPORt commands refer to the Balanced port number. There can only be up to two balanced ports. This method is compatible with the ENA network analyzer.
  
  - The LPORt commands refer to the Logical port number. A balanced port can appear as either logical port 1, 2, or 3. These are the references as they appear in the front-panel user interface.
CALCulate<cnum>:FSIMulator:DRAFt:APPLy

**Applicable Models:** All

*(Write only)* Copies the draft fixture to active fixture. Exact same functionality as CALC:FSIM:APPL

**Parameters**

Not applicable

**Example**

```
calculate:fsimulator:draft:apply
```

**Return Type**

Not applicable

**Default**

Not applicable

---

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:ADD <block type>, <fixt port count>

**Applicable Models:** All

*(Write only)* Create a block of the specified block type with the specified fixture port count. By default, this block will be created on the VNA Port 1 with "de-embed". It can be moved and changed to "embed" with commands below.

**Parameters**

<p>| &lt;cnum&gt; | Any existing channel number. If unspecified, value is set to 1 |
| &lt;circN&gt; | Circuit Number |
| &lt;block type&gt; | File  | SNP File Block (valid parameters: FILE, FILE:EXTRapolate) |
| | RLGLoop | Ground Loop (Shunt L circuit model) (valid parameters: R, L, C, G) |
| | RCGLoop | Ground Loop (Shunt C circuit model) (valid parameters: R, L, C, G) |
| | CGGLoop | Ground Loop (Shunt C circuit model, but for convenience, user can specifier G instead of R) (valid parameters: R, L, C, G) |
| | FGLoop | Ground Loop (valid parameters: FILE, FILE:EXTRapolate, file must be S1P file) |
| | SLPC | Port Matching Circuit: Series L - Shunt C (valid parameters: R, R2, L, C,G, G2) |</p>
<table>
<thead>
<tr>
<th>Block Type</th>
<th>Description</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCSL</td>
<td>Port Matching Circuit: Shunt C - Series L</td>
<td>$R, R_2, L, C, G, G_2$</td>
</tr>
<tr>
<td>PLSC</td>
<td>Port Matching Circuit: Shunt L - Series C</td>
<td>$R, R_2, L, C, G, G_2$</td>
</tr>
<tr>
<td>SCPL</td>
<td>Port Matching Circuit: Series C - Shunt L</td>
<td>$R, R_2, L, C, G, G_2$</td>
</tr>
<tr>
<td>PLPC</td>
<td>Port Matching Circuit: Shunt L - Shunt C</td>
<td>$R, R_2, L, C, G, G_2$</td>
</tr>
<tr>
<td>SCPC</td>
<td>Port Matching Circuit: Series C - Shunt/ C</td>
<td>$R, R_2, L, L_2, C, C_2, G, G_2$</td>
</tr>
<tr>
<td>PCSC</td>
<td>Port Matching Circuit: Shunt C - Series C</td>
<td>$R, R_2, L, L_2, C, C_2, G, G_2$</td>
</tr>
<tr>
<td>SLPL</td>
<td>Port Matching Circuit: Series L - Shunt L</td>
<td>$R, R_2, L, L_2, C, C_2, G, G_2$</td>
</tr>
<tr>
<td>PLSL</td>
<td>Port Matching Circuit: Shunt L - Series L</td>
<td>$R, R_2, L, L_2, C, C_2, G, G_2$</td>
</tr>
<tr>
<td>ILINe</td>
<td>Ideal Line. For this block type, user can specify delay and loss value.</td>
<td></td>
</tr>
<tr>
<td>TRANSformer</td>
<td>Transformer Circuit. For this block type, User can specify Rin and Rout values to represent a transformer.</td>
<td></td>
</tr>
<tr>
<td>D4PMatching</td>
<td>Differential Matching Circuit: PLPC</td>
<td></td>
</tr>
<tr>
<td>D4PFile</td>
<td>Differential Matching Circuit: File Type (S2P)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Port Count</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;fixt port count&gt;</td>
<td>Fixture port number</td>
</tr>
</tbody>
</table>

**Examples**

```
CALC:FSIM:DRAF:CIRC:ADD SLPC,1
```

```
calculate2:fsimulator:draft:circuit:add SLPC,1
```

**Query Syntax**

Not Applicable

**Return Type**

Not Applicable

**Default**

Not Applicable
Applicable Models: All

(Read only) Returns a comma-separated list of circuit numbers created in the draft circuit.

Parameters
- `<cnum>` Any existing channel number. If unspecified, value is set to 1

Examples
- `CALC:FSIM:DRAF:CIRC:CAT?`
- `calculate2:fsimulator:draft:circuit:catalog?`

Query Syntax
- `CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit:CATalog?`

Return Type
- comma-separated string

Default
- Not Applicable

Applicable Models: All

(Write only) Deletes the specified circuit.

Parameters
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

Examples
- `CALC:FSIM:DRAF:CIRC:DEL`
- `calculate2:fsimulator:draft:circuit:delete`

Query Syntax
- Not Applicable

Return Type
- Not Applicable

Default
- Not Applicable
Applicable Models: All

(Read-Write) Sets the Device ports for the specified circuit. The number of ports must be even and correspond to the number of ports of the fixture. This command can be used to reverse or change the order of ports in an SNP file.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

**Examples**

```
CALC:FSIM:DRAF:CIRC:DEV:PORT 2,1
```

```
calculate2:fsimulator:draft:circuit:device:ports 2,1
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:DEVice:PORTs?

**Return Type** Numeric

**Default** Not Applicable

---

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:EMBED:TYPE <embed | deembed>

Applicable Models: All

(Read-Write) Sets whether the circuit should be embedded or de-embedded. Valid for all block Types

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Choose from:
  - **EMBED** - Add Network circuit.
  - **DEEMBED** - Remove Network circuit
- `<circN>` Circuit Number

**Examples**

```
CALC:FSIM:DRAF:CIRC:EMBED:TYPE EMBED
```

```
calculate2:fsimulator:draft:circuit:embed:type deembed
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:EMBED:TYPE?

**Return Type** character
CALCulate\(<cnum>\):FSIMulator:DRAFt:CIRCuit\(<\text{circN}>\):FILE <filename>

**Applicable Models:** All

(Read-Write) Sets the filename. Valid blocks: FILE, D4PFile, FGLoop

The block is created using CALCulate\(<cnum>\):FSIMulator:DRAFt:CIRCuit\([n]\):ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

\(<\text{cnum}>\) Any existing channel number. If unspecified, value is set to 1

\(<\text{filename}>\) File name and extension (.s2P) of the de-embedding circuit.

Files are stored in the default folder "D:\".

To recall from a different folder, specify the full path name.

\(<\text{circN}>\) Circuit Number

**Examples**

```
CALC:FSIM:DRAF:CIRC:FILE 'myFile.s2P'
calculate2:fsimulator:draft:circuit:file "D:\myFile.s2P"
```

**Query Syntax** CALCulate\(<cnum>\):FSIMulator:DRAFt:CIRCuit\(<\text{circN}>\):FILE?

**Return Type** string

**Default** Not Applicable

CALCulate\(<cnum>\):FSIMulator:DRAFt:CIRCuit\(<\text{circN}>\):FILE:EXTRapolate <bool>
Applicable Models: All

(Read-Write) Sets whether or not extrapolation is allowed (TRUE/FALSE). Valid blocks: FILe, D4PFile, FGLoop.

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<n>:ADD <block type>, <fuxt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` Choose from:
  - **TRUE or 1** - Turns Extrapolation ON
  - **FALSE or 0** - Turns Extrapolation OFF
- `<circN>` Circuit Number

**Examples**

```
CALC:FSIM:DRAF:CIRC:FILE:EXTR TRUE

calculate2:fsimulator:draft:circuit:file:extrapolate false
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:FILE:EXTRapolate?

**Return Type**

Boolean

**Default**

Not Applicable

---

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:FILE:MODify <enum>

Applicable Models: All

(Read-Write) Sets all reflection and crosstalk parameters on the DUT side to zero. Learn more.

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<n>:ADD <block type>, <fuxt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<enum>` Choose from:
  - **NONE** - Does not modify the block.
  - **NREFlect** - Sets all reflection parameters on the DUT-side to zero.
NXTalk - Sets all reflection and crosstalk parameters on the DUT-side to zero.

```plaintext
Examples
CALC:FSIM:DRAF:CIRC:FILE:MOD NONE
calculate2:fsimulator:draft:circuit:file:modify nreflect
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:FILE:MODify?

**Return Type**

Enumeration

**Default**

NONE

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:NEXT?

**Applicable Models:** All

*(Read only)* Returns the next free circuit number than be used to create a new circuit block

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

```plaintext
Examples
CALC:FSIM:DRAF:CIRC:NEXT?
calculate2:fsimulator:draft:circuit:next?
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:NEXT?

**Return Type**

Numeric

**Default**

Not Applicable

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:C <value>
Applicable Models: All

(Read-Write) Sets the capacitance (C) circuit element for the specified circuit. Valid blocks: RLGLoop, RCGLoop, CGGLoop, SLPC, PCSL, PLSC, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL.

The block is create using CALCulate<cnm>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**
- `<cnm>` Any existing channel number. If unspecified, value is set to 1
- `<value>` Capacitance value in farads. Choose a value between -1E18 to 1E18
- `<circN>` Circuit Number

**Examples**
- `CALC:FSIM:DRAF:CIRC:PAR:C 0.00002`
- `calculate2:fsimulator:draft:circuit:parameter:C 0.00002`

**Query Syntax**
- `CALCulate<cnm>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:C?`

**Return Type**
- Numeric

**Default**
- 0

---

CALCulate<cnm>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:C2 <value>

Applicable Models: All

(Read-Write) Sets the capacitance (C2) circuit element for the specified circuit. Valid blocks: SLPC, PCSL, PLSC, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL.

The block is create using CALCulate<cnm>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**
- `<cnm>` Any existing channel number. If unspecified, value is set to 1
- `<value>` Capacitance value in farads. Choose a value between -1E18 to 1E18
- `<circN>` Circuit Number

**Examples**
- `CALC:FSIM:DRAF:CIRC:PAR:C2 0.00002`
- `calculate2:fsimulator:draft:circuit:parameter:C2 0.00002`

**Query Syntax**
- `CALCulate<cnm>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:C2?`

**Return Type**
- Numeric
**CALCulate<nump>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:DELay <num>**

**Applicable Models:** All

*(Read-Write)* Sets port extension delay in time. Valid blocks: ILIN.

The block is created using CALCulate<nump>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<nump>` Any existing channel number. If unspecified, value is set to 1
- `<num>` The port extension in seconds; may include suffix. Choose a number between: -1E18 and 1E18
- `<circN>` Circuit Number

**Examples**

```
CALC:FSIM:DRAF:CIRC:PAR:DEL 2MS
```
```
calculate2:fsimulator:draft:circuit:parameter:delay .00025
```

**Query Syntax**

CALCulate<nump>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:DEL?

**Return Type**

Numeric

**Default**

0

---

**CALCulate<nump>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:G <value>**

**Applicable Models:** All

*(Read-Write)* Sets the conductance (G) circuit element for the specified circuit. Valid blocks: RLGLoop, RCGLoop, CGGLoop, SLPC, PCSL, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL.

The block is created using CALCulate<nump>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<nump>` Any existing channel number. If unspecified, value is set to 1
- `<value>` Conductance value in siemens. Choose a value between -1E18 to 1E18
- `<circN>` Circuit Number

---

3408
CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:G <value>

**Applicable Models:** All

(Read-Write) Sets the conductance (G2) circuit element for the specified circuit. Valid blocks: SLPC, PCSL, PLSC, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<value>` Conductance value in siemens. Choose a value between -1E18 to 1E18
- `<circN>` Circuit Number

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:DRAF:CIRC:PAR:G 0.00002</td>
<td></td>
</tr>
<tr>
<td>calculate2:fsimulator:draft:circuit:parameter:g 0.00002</td>
<td></td>
</tr>
</tbody>
</table>

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:G2?

**Return Type**

Numeric

**Default**

0

---

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:L <value>

---

3409
Applicable Models: All

(Read-Write) Sets the inductance (L) circuit element for the specified circuit. Valid blocks: RLGLoop, RCGLoop, CGGLoop, SLPC, PCSL, PLSC, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL.

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<value>` Inductance value in henries. Choose a value between -1E18 to 1E18
- `<circN>` Circuit Number

**Examples**

```
CALC:FSIM:DRAF:CIRC:PAR:L 0.00002
```
```
calculate2:fsimulator:draft:circuit:parameter:l 0.00002
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:L?

**Return Type** Numeric

**Default** 0

Applicable Models: All

(Read-Write) Sets the inductance (L2) circuit element for the specified circuit. Valid blocks: SLPC, PCSL, PLSC, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL.

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<value>` Inductance value in henries. Choose a value between -1E18 to 1E18
- `<circN>` Circuit Number

**Examples**

```
CALC:FSIM:DRAF:CIRC:PAR:L2 0.00002
```
```
calculate2:fsimulator:draft:circuit:parameter:l2 0.00002
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:L2?

**Return Type** Numeric

**Default** 0
CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:LOSS:VALue <num>

**Applicable Models:** All

*(Read-Write)* Sets the loss at DC. Valid blocks: ILIN.

The block is created using `CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>`. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Loss in dB. Choose a value between -90 and 90.
- `<circN>` Circuit Number

**Examples**

```
```

```
calculate2:fsimulator:draft:circuit:parameter:loss:value .1
```

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:LOSS:VALue?
```

**Return Type** Numeric

**Default** 0

---

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:R <value>

**Applicable Models:** All

*(Read-Write)* Sets the resistance (R) circuit element for the specified circuit. Valid blocks: RLGLoop, RCGLoop, CGGLoop, SLPC, PCSL, PLSC, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL.

The block is created using `CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>`. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<value>` Resistance value in ohms. Choose a value between -1E18 to 1E18
- `<circN>` Circuit Number
### CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:R2 <value>

**Applicable Models:** All

*(Read-Write)* Sets the resistance (R2) circuit element for the specified circuit. Valid blocks: SLPC, PCSL, PLSC, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[<n>]:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<value>` Resistance value in ohms. Choose a value between -1E18 to 1E18
- `<circN>` Circuit Number

### Examples
- CALC:FSIM:DRAF:CIRC:PAR:R 0.00002
- calculate2:fsimulator:draft:circuit:parameter:r 0.00002

**Query Syntax**
- CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:R2?

**Return Type**
- Numeric

**Default**
- 0

---

### CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:RIN <value>

**Query Syntax**
- CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:R2?

**Return Type**
- Numeric

**Default**
- 0
**Applicable Models:** All

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<value>` Resistance value in ohms. Choose a value between -1E18 to 1E18
- `<circN>` Circuit Number

**Examples**

```
CALC:FSIM:DRAF:CIRC:PAR:RIN 0.00002
```

```
calculate2:fsimulator:draft:circuit2:parameter:rin 0.00002
```

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:RIN?
```

**Return Type**

Numeric

**Default**

0

---

**Applicable Models:** All

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<value>` Resistance value in ohms. Choose a value between -1E18 to 1E18
- `<circN>` Circuit Number

**Examples**

```
CALC:FSIM:DRAF:CIRC:PAR:ROUT 0.00002
```

```
calculate2:fsimulator:draft:circuit2:parameter:rout 0.00002
```

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:ROUT?
```

**Return Type**

Numeric

**Default**

0
CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:Z0 <value>

**Applicable Models:** All

(Read-Write) Sets the impedance (Z0) for the ideal block (scalar value). Valid blocks: ILIN.

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<value>`: Impedance value in ohms. Choose a value between -1E18 to 1E18
- `<circN>`: Circuit Number

**Examples**
- CALC:FSIM:DRAF:CIRC:PAR:Z0 0.00002
- calculate2:fsimulator:draft:circuit:parameter:z0 0.00002

**Query Syntax**
CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:PARameter:Z0?

**Return Type**
Numeric

**Default**
0

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit:RESet

**Applicable Models:** All

(Write only) Resets the circuit section to preset values

**Note:** This command impacts the circuits and the port extensions. It does not change any impedance conversion values.

**Parameters**
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1

**Examples**
- CALC:FSIM:DRAF:CIRC:RES
- calculate2:fsimulator:draft:circuit:reset

**Query Syntax**
Not Applicable

**Return Type**
Not Applicable

**Default**
Not Applicable
CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:STATe <bool>

**Applicable Models:** All

*(Read-Write)* Turns the circuit ON and OFF. Valid for all block types

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` Choose from:
  - **ON or 1** - Turns the circuit ON
  - **OFF or 0** - Turns the circuit OFF
- `<circN>` Circuit Number

**Examples**

CALC:FSIM:DRAF:CIRC:STAT ON

calculate2:fsimulator:draft:circuit1:state off

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:STATe?

**Return Type**

Boolean

**Default**

OFF

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:VNA:PORTs <port list>

**Applicable Models:** All

*(Read-Write)* Sets the VNA ports for the specified circuit.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

**Examples**


**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:VNA:PORTs?

**Return Type**

Numeric
CALCulate<cnum>:FSIMulator:DRAFt:DISCard

**Applicable Models:** All

*(Write only)* Copies the active fixture into scratch/draft fixture (discards changes on scratch/draft fixture)

**Parameters**
- Not applicable

**Example**
- **CALC:FSIM:DRAF:DIS**
  - calculate:fsimulator:draft:discard

**Return Type**
- Not applicable

**Default**
- Not applicable

CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:DELay <value>

**Applicable Models:** All

*(Read-Write)* Sets and returns the delay value in time at the specified port.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<pnum>` Port Number that will receive the delay setting. If unspecified, value is set to 1.
- `<value>` The port extension in seconds; may include suffix. Choose a number between: -1E18 and 1E18 e.

**Examples**
- **CALC:FSIM:DRAF:EXT:PORT1:DEL 0.02**
  - calculate:fsimulator:draft:extension:port2:delay .003

**Query Syntax**
- CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:DECLay?

**Return Type**
- Numeric

**Default**
- 0
Applicable Models: All

(Read-Write) Sets and returns the units for specifying port extension delay in physical length (distance).

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<pnum>` Port Number. This number is ignored and all ports have the same setting.
- `<char>` Units for delay in distance. Choose from:
  - METer
  - FEET
  - INCH

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

Query Syntax

`CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:DISTance:UNIT?`

Return Type

Character

Default

METer

---

Applicable Models: All

(Read-Write) Sets and returns the port extension delay in physical length (distance).

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<pnum>` Port Number that will receive the delay setting. If unspecified, value is set to 1.

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

Query Syntax

`CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:DISTance:VALue?`

Return Type

Numeric

Default

0
CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:END

Applicable Models: All

(Write Only) Moves the port extension block to the right-most side of the circuit sections. This command allows the port extensions to be moved around in the circuit.

Parameters

<cnun> Any existing channel number. If unspecified, value is set to 1.
<pnum> Port Number that will receive the delay setting. If unspecified, value is set to 1.

Examples

CALC:FSIM:DRAF:EXT:PORT1:END
calculate:fsimulator:draft:extension:port2:end

Query Syntax Not applicable

Return Type Not applicable

Default Not applicable

CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:LOSS<n>:FREQuency <value>

Applicable Models: All

(Read-Write) Sets and returns the frequency for the Freq and Loss pair number and for the specified port number.

Learn about Loss Compensation values.

Parameters

<cnun> Any existing channel number. If unspecified, value is set to 1
<pnum> Port Number that will receive the freq/loss settings. If unspecified, value is set to 1.

<n> Freq and Loss pair number. Choose from 1 or 2. If unspecified, value is set to 1.

<value> Frequency value. Choose a frequency within the frequency span of the VNA.

Examples

calculate:fsimulator:draft:extension:port2:loss1:frequency 2E10

Query Syntax CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:LOSS<n>:FREQuency?

Return Type Numeric
**CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:LOSS<n>:VALue <value>**

**Applicable Models:** All

*(Read-Write)* Sets and returns the Loss value for the specified port number.

Learn about Loss Compensation values.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number that will receive the Freq/Loss settings. If unspecified, value is set to 1.
- `<n>` Loss "Use" number. Choose from 1 or 2. If unspecified, value is set to 1.
- `<value>` Loss in dB. Choose a value between -90 and 90

**Examples**

- `CALC:FSIM:DRAF:EXT:PORT:LOSS1:VAL 1`
- `calculate:fsimulator:draft:extension:port2:loss2:value .1`

**Query Syntax**

`CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:LOSS<n>:VALue?`

**Return Type** Numeric

**Default** 0

**CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:LOSS<n>[:STATe] <bool>**
Applicable Models: All

(Read-Write) Sets and returns the ON/OFF state for the Freq and Loss pair number and for the specified port number.

Learn about Loss Compensation values.

Note: This command affects ALL measurements on the specified channel.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<pnum>`: Port Number that will receive the Freq/Loss settings. If unspecified, value is set to 1.
- `<n>`: Freq and Loss pair. Choose from 1 or 2. If unspecified, value is set to 1.
- `<value>`: State of Freq and Loss values for port extension.

0 or OFF Specified Freq and Loss values are OFF

1 or ON Specified Freq and Loss values are ON

Examples

```
CALC:FSIM:DRAF:EXT:PORT:LOSS:STAT 0
```

calculate:fsimulator:draft:extension:port2:loss2:state on

Query Syntax

CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:LOSS<n>[:STATe]?

Return Type

Boolean

Default

OFF

CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:DCLoss:VALue <value>
### Applicable Models: All

**(Read-Write)** Sets and returns the Port Loss at DC value for the specified port number.

Learn about Loss Compensation values.

**Note:** This command affects ALL measurements on the specified channel.

#### Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<pnum>`: Port number to receive Loss value. If unspecified, value is set to 1.
- `<value>`: Loss in dB. Choose a value between -90 and 90.

#### Examples

```
```
```
calculate:fsimulator:draft:extension:port2:dcloss:value .1
```

#### Query Syntax

```
CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:DCLoss:VALue?
```

#### Return Type

Numeric

**Default** 0

---

**CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:MEDium <char>**

**Applicable Models: All**

**(Read-Write)** Sets and returns the media type of the added fixture or transmission line.

See also SENS:CORR:EXT:PORT:SYSMedia

**Note:** This command affects ALL measurements on the specified channel.

#### Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<pnum>`: Port Number. This number is ignored and all ports have the same setting.
- `<char>`: Medium type. Choose from:
  - COAX
  - WAVEguide
**Examples**

```
CALC:FSIM:DRAF:EXT:PORT:MED COAX
```

```
calculate:fsimulator:draft:extension:port2:medium waveguide
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt: EXTension:PORT<pnum>:MEDium?

**Return Type**

Character

**Default**

COAX

---

**CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:WAVEguide:FCUToff <value>**

**Applicable Models:** All

*(Read-Write)* Sets and returns the cutoff (minimum) frequency of the added waveguide fixture or transmission line.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number for which media type is being set. If unspecified, value is set to 1.
- `<value>` Cutoff frequency in Hz.

This value is ignored when CALC:FSIM:DRAF:EXT:PORT:MED is set to COAX for the same port.

**Examples**

```
CALC:FSIM:DRAF:EXT:PORT:WAVE:FCUT 1e8
```

```
calculate:fsimulator:draft:extension:port2:waveguide:cutoff 100Mhz
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt: EXTension:PORT<pnum>:WAVEguide:FCUToff?

**Return Type**

Numeric

**Default**

System Media Cutoff Frequency

---

**CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:WAVEguide:COUPle <bool>**
Applicable Models: All

(Read-Write) Sets and returns the state of coupling with the system Velocity Factor value. Learn more.

**Note:** This command potentially affects ALL measurements on the VNA.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number. This number is ignored and all ports have the same setting.
- `<bool>` Coupling state. Choose from:
  - **ON** (or 1) - Velocity Factor is coupled with the system setting.
  - **OFF** (or 0) - Velocity Factor is NOT coupled with the system setting.

**Examples**
- `calculate:fsimulator:draft:extension:port2:waveguide:couple off`

**Query Syntax**
- `CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:WAVEguide:COUPle?`

**Return Type** Boolean

**Default** 1 or ON (Coupled)

---

Applicable Models: All

(Read-Write) Sets and returns the velocity factor of the fixture or added transmission line.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number for which velocity factor is being set. If unspecified, value is set to 1.
- `<value>` Velocity Factor.

Set SENS:CORR:EXT:PORT:SYSV to use the system velocity factor.

**Examples**
- `CALC:FSIM:DRAF:EXT:PORT:VEL:FAC .6`
- `calculate:fsimulator:draft:extension:port2:velocity:factor 1`
CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:VELOCITY:FACTOR?

Return Type: Numeric
Default: System Velocity Factor

CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>:VELOCITY:COUPLE <bool>

Applicable Models: All
(Read-Write) Sets and returns the state of coupling with the system Velocity Factor value. Learn more.

Note: This command potentially affects ALL measurements on the VNA.

Parameters
- <cnum> Any existing channel number. If unspecified, value is set to 1
- <pnum> Port Number. This number is ignored and all ports have the same setting.
- <bool> Coupling state. Choose from:
  - ON (or 1) - Velocity Factor is coupled with the system setting.
  - OFF (or 0) - Velocity Factor is NOT coupled with the system setting.

Examples
- \texttt{CALC:FSIM:DRAF:EXT:PORT:VEL:COUP 1}
- \texttt{calculate:fsimulator:draft:extension:port2:velocity:couple off}

CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>[:STATE] <bool>

Return Type: Boolean
Default: 1 or ON (Coupled)
Applicable Models: All

(Read-Write) Sets and returns the state of port extension for the specified port.

Note: This command potentially affects ALL measurements on the VNA.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number for which port extension is being set. If unspecified, value is set to 1.
- `<bool>` Port extension state. Choose from:
  - **ON** (or 1) - Port extension is on
  - **OFF** (or 0) - Port extension is off.

Examples

```
CALC:FSIM:DRAF:EXT:PORT:STAT 1
```

```
calculate:fsimulator:draft:extension:port2:state off
```

Query Syntax

```
CALCulate<cnum>:FSIMulator:DRAFt:EXTension:PORT<pnum>[:STATe]?
```

Return Type

Boolean

Default

0

---

CALCulate<cnum>:FSIMulator:DRAFt:SAVE <snpName>

Applicable Models: All

(Write only) Saves SNP file corresponding to the entire scratch/draft fixture.

Parameters

- `<snpName>` String - snp file name. Appends ".sNp" depending on num ports

Example

```
CALC:FSIM:DRAF:SAVE "myFixture"
```

```
calculate:fsimulator:draft:save "D:\myfixture"
```

Return Type

Not applicable

Default

Not applicable

---

CALCulate<cnum>:FSIMulator:DRAFt:SECTION:CIRcuit:ENABle <ON | OFF>
Applicable Models: All

(Read-Write) Turns all circuits ON or OFF for all ports.

Parameters

<cnun> Any existing channel number. If unspecified, value is set to 1

<ON | OFF> ON (or 1) - turns all circuits ON.

OFF (or 0) - turns all circuits OFF.

Examples

CALC:FSIM:DRAF:SECT:CIRC:ENAB ON

calculate2:fsimulator:draft:section:circuit:enable off

Query Syntax

CALCulate<cnun>:FSIMulator:DRAFt:SECTionCIRCuit:ENABle?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF

CALCulate<cnun>:FSIMulator:DRAFt:SECTion:EXTension:ENABle <ON | OFF>

Applicable Models: All

(Read-Write) Turns port extensions ON or OFF for all ports.

Parameters

<cnun> Any existing channel number. If unspecified, value is set to 1

<ON | OFF> ON (or 1) - turns port extensions ON.

OFF (or 0) - turns port extensions OFF.

Examples

CALC:FSIM:DRAF:SECT:EXT:ENAB ON

calculate2:fsimulator:draft:section:extension:enable off

Query Syntax

CALCulate<cnun>:FSIMulator:DRAFt:SECTion:EXTension:ENABle?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF

CALCulate<cnun>:FSIMulator:DRAFt:SECTion:ZCONversion:ENABle <ON | OFF>
**Applicable Models:** All

*(Read-Write)* Turns all ZConversions ON or OFF (both SE and BAL).

**Note:** This command affects ALL measurements on the specified channel.

### Parameters
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<ON | OFF>`: ON (or 1) - turns all ZConversions ON. OFF (or 0) - turns all ZConversions OFF.

### Examples
- `CALC:FSIM:DRAF:SECT:ZCON:ENAB ON`
- `calculate2:fsimulator:draft:section:zconversion:enable off`

### Query Syntax
- `CALCulate<cnum>:FSIMulator:DRAFt:SECTion:ZCONversion:ENABle?`

### Return Type
- Boolean (1 = ON, 0 = OFF)

### Default
- OFF

---

**Applicable Models:** All

*(Read-Write)* Sets the differential port impedance conversion function ON/OFF. Must also set the fixture simulator function to ON using CALC:FSIM:STAT.

**See Critical Note**

### Parameters
- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<bool>`: State of the differential port impedance conversion function. Choose from
  - **OFF (or 0)**: Differential port impedance conversion OFF
  - **ON (or 1)**: Differential port impedance conversion ON

### Examples
- `CALC:FSIM:DRAF:ZCON:DIFF:STAT 1`
- `calculate2:fsimulator:draft:zconversion:DIFferential:state off`

### Query Syntax
- `CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:DIFFerential:STATe?`

### Return Type
- Boolean
**Applicable Models:** All

(Read-Write) Sets the complex impedance value for the differential port impedance conversion function.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel, unspecified, `<cnum>` is set to 1.
- `<pnum>` Balanced port number. Choose from 1 to 999.

**Note:** See Balanced port versus Logical port.

- `<real>` Real part of the Impedance value in Units. Choose a number between 0 and 1E18
- `<img>` Imaginary part of the Impedance value in Units. Choose a number between 0 and 1E18.

**Examples**

```plaintext
CALC:FSIM:DRAF:ZCON:DIFF:BPOR2:COMPL 200,10
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:DIFFerential:BPORt<pnum>:COMPLex

**Return Type** Numeric

**Default** 0,0

**CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:DIFFerential:BPORt<pnum>:SCALar<value>**
Applicable Models: All

(Read-Write) Sets the real part of impedance value for the differential port impedance conversion function.

See Critical Note

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>`: Balanced port number. Choose from 1 to 999.

Note: See Balanced port versus Logical port.

- `<value>`: Real part of the Impedance value in Units. Choose a number between 0 and 1E18.

Examples

```
CALC:FSIM:DRAF:ZCON:DIFF:BPOR2:SCAL 200,10
```
```
```

Query Syntax

```
CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:DIFFerential:BPORt<pnum>:SCALer?
```

Return Type

Numeric

Default

0

---

Applicable Models: All

(Read-Write) Sets the complex impedance value for the differential mode port impedance conversion function.

See Critical Note

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>`: Logical port number. Choose from 1 to 999.

Note: See Balanced port versus Logical port.

- `<real>`: Real part of the Impedance value in Units. Choose a number between 0 and 1E18.
<table>
<thead>
<tr>
<th><strong>Examples</strong></th>
<th>CALC:FSIM:DRAF:ZCON:DIFF:LPOR2:COMPL 200,10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>calculate2:fsimulator:draft:zconversion:differential:1port2:complex 20,3</td>
</tr>
<tr>
<td><strong>Query Syntax</strong></td>
<td>CALCulate&lt;cnum&gt;:FSIMulator:DRAFt:ZCONversion:DIFFerential:LPORt&lt;pnum&gt;:COM?</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Numeric</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0,0</td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:DIFFerential::LPORt<pnum>:SCALar <value>**

**Applicable Models:** All

*(Read-Write)* Sets the real part of impedance value for the differential:mode port impedance conversion function.

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel; if unspecified, `<cnum>` is set to 1.

- `<pnum>`: Logical port number. Choose from 1 to 999.

**Note:** See Balanced port versus Logical port.

- `<value>`: Real part of the Impedance value in Units. Choose a number between 0 and 1E18.

**Examples**

- CALC:FSIM:DRAF:ZCON:COMM:LPOR2:Diff 200,10
- calculate2:fsimulator:draft:zconversion:commonmode:1port2:scalar 20,3

**Query Syntax**

- CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:COMMONmode:STATe <bool>

**Return Type**

- Numeric

**Default**

- 0
Applicable Models: All

(Read-Write) Sets the common mode port impedance conversion function ON/OFF. Must also set the fixture simulator function to ON using CALC:FSIM:STAT.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<bool>` State of the differential port impedance conversion function. Choose from
  - **OFF (or 0)** Differential port impedance conversion OFF
  - **ON (or 1)** Differential port impedance conversion ON

**Examples**

CALC:FSIM:DRAF:ZCON:COMM:STAT 1

calculate2:fsimulator:draft:zconversion:commonmode:state off

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:COMMonmode:STATe?

**Return Type**

Boolean

**Default**

Off

---

CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:COMMonmode:BPORt<pnum>:COMPlex

REAL, IMAG

Applicable Models: All

(Read-Write) Sets the complex impedance value for the common mode port impedance conversion function.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>` Balanced port number. Choose from 1 to 999.
- `<real>` Real part of the Impedance value in Units. Choose a number between 0 and 1E18
- `<img>` Imaginary part of the Impedance value in Units. Choose a number between 0 and 1E18.

**Note:** See Balanced port versus Logical port.
CALCulate\(<cnum>\):FSIMulator:DRAFT:ZCONversion:COMMONmode:BPOR\(<pnum>\):COMPLex \(<real>,<img>\)

**Examples**

CALC:FSIM:DRAF:ZCON:COMM:BPOR2:COMPL 200,10
calculate2:fsimulator:draft:zconversion:commonmode:bport2:complex 20,3

**Query Syntax**

CALCulate\(<cnum>\):FSIMulator:DRAFT:ZCONversion:COMMONmode:BPOR\(<pnum>\):COMPlex?

**Return Type**

Numeric

**Default**

0,0

Applicable Models: All

(Read-Write) Sets the real part of impedance value for the common mode port impedance conversion function.

See Critical Note

**Parameters**

\(<cnum>\) Channel number of the measurement. There must be a selected measurement on that channel, unspecified, \(<cnum>\) is set to 1.

\(<pnum>\) Balanced port number. Choose from 1 to 999.

\(<value>\) Real part of the Impedance value in Units. Choose a number between 0 and 1E18

**Examples**


**Query Syntax**

CALCulate\(<cnum>\):FSIMulator:DRAFT:ZCONversion:COMMONmode:BPOR\(<pnum>\):SCALar \(<value>\)

**Return Type**

Numeric

**Default**

0,0
Applicable Models: All

(Read-Write) Sets the complex impedance value for the common mode port impedance conversion function.

See Critical Note

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>`: Logical port number. Choose from 1 to 999.

Note: See Balanced port versus Logical port.

- `<real>`: Real part of the Impedance value in Units. Choose a number between 0 and 1E18.
- `<img>`: Imaginary part of the Impedance value in Units. Choose a number between 0 and 1E18.

Examples

```
CALC:FSIM:DRAF:ZCON:COMM:LPOR2:COMPL 200,10
```

```
calculate2:fsimulator:draft:zconversion:commonmode:lport2:complex 20,3
```

Query Syntax

```
CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:COMMONmode:LPORt<pnum>:COMPL?
```

Return Type: Numeric

```
CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:COMMONmode:LPORt<pnum>:SCALar <value>
```

Applicable Models: All

(Read-Write) Sets the real part of impedance value for the common mode port impedance conversion function.

See Critical Note

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>`: Logical port number. Choose from 1 to 999.

Note: See Balanced port versus Logical port.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;value&gt;</td>
<td>Real part of the Impedance value in Units. Choose a number between 0 and 1E18</td>
</tr>
<tr>
<td></td>
<td>calculate2:fsimulator:draft:zconversion:commonmode:1port2:scalar 20</td>
</tr>
<tr>
<td>Query Syntax</td>
<td>CALCulate&lt;cnum&gt;:FSIMulator:DRAFt:ZCONversion:COMMONmode:LPORt&lt;pnum&gt;:SC</td>
</tr>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:SENDed:PORT<pnum>:STATe <bool>**

**Applicable Models:** All

*(Read-Write)* Sets the single-ended impedance conversion function ON/OFF. Must also set the fixture simulator function to ON using CALC:FSIM:STAT.

**See Critical Note**

**Parameters**

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<pnum>** Port number. Choose from 1 to 999.
- **<bool>** State of the single-ended port impedance conversion function. Choose from **OFF (or 0)** single-ended port impedance conversion OFF
  - **ON (or 1)** single-ended port impedance conversion ON

**Examples**

<table>
<thead>
<tr>
<th></th>
<th>CALC:FSIM:DRAF:ZCON:SEND:PORT3:STAT 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>calculate2:fsimulator:draft:zconversion:sended:port2:state off</td>
</tr>
</tbody>
</table>

**Query Syntax**

CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:SENDed:PORT<pnum>:STATe?

**Return Type**

Boolean

**Default** Off

**CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:SENDed:PORTpnum>:COMPLEX**

Applicable Models: All

(Read-Write) Sets the complex impedance value for the single-ended port impedance conversion function.

See Critical Note

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>`: Pport number. Choose from 1 to 999.
- `<real>`: Real part of the Impedance value in Units. Choose a number between 0 and 1E18.
- `<img>`: Imaginary part of the Impedance value in Units. Choose a number between 0 and 1E18.

Examples

```
CALC:FSIM:DRAF:ZCON:SEND:POR2:COMPL 200,10
```

Query Syntax


Return Type

Numeric

Default

0,0

---

**CALCulate<cnum>:FSIMulator:DRAFt:ZCONversion:SENDed:PORT<pnum>:SCALer <val>**

Applicable Models: All

(Read-Write) Sets the real part of impedance value for the single-ended port impedance conversion function.

See Critical Note

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>`: Pport number. Choose from 1 to 999.
- `<val>`: Real part of the Impedance value in Units. Choose a number between 0 and 1E18.

Examples

```
CALC:FSIM:DRAF:ZCON:SEND:POR2:SCAL 0
```

```
calculate2:fsimulator:draft:zconversion:sended:port2:scalar 20,3
```
<table>
<thead>
<tr>
<th>Query</th>
<th>CALCulate&lt;cnum&gt;:FSIMulator:DRAFT:ZCONversion:SENDed:PORT&lt;pnum&gt;:SCALer?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax</td>
<td></td>
</tr>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
</tbody>
</table>
Calculate:FSimulator Active Commands

CALCulate:FSIMulator:
  APPLy
CIRCuit:
  | CATalog
  | DEVice
    | PORTs
  | EMBED
    | TYPE
  | FILE
    | EXTRapolate
    | MODify?
  | PArameter
    | C
    | C2
    | DELay
    | G
    | G2
    | L
    | L2
    | LOSS
      | VALue
    | R
    | R2
    | RIN
    | ROUT
    | Z0
| STATe  |
| VNA    |
| PORTs  |
| EXTension:PORT: |
| DELay  |
| DISTance |
| VALue  |
| UNIT   |
| LOSS   |
| FREQuency |
| VALue  |
| STATe  |
| MEDium |
| WAVEguide |
| FCUToff |
| COUPLE |
| VELocity |
| FACTor |
| COUPLE |
| DCLoss:VALue |
| STATe  |
| POWER  |
| COMPensate |
| MODE   |
| PORT   |
| COMPensate |
| STATe  |
| SAVE   |
| SECTion
| CIRCuit
  | ENABle
| EXTension
  | ENABle
| ZCONversion
  | ENABle
| TOPology
  | LOAD
  | SAVE
| ZCONversion
  | DIFFerential
    | :STATe
    | :BPORt
      | :COMPLex
      | :SCALar
    | :LPORt
      | :COMPLex
      | :SCALar
| COMMonmode
  | :STATe
  | :BPORt
    | :COMPLex
    | :SCALar
  | :LPORt<pnum>
    | :COMPLex
    | :SCALar
| SENDed:PORT
  | :STATe
  | :COMPLex
See Also

- Learn about Using Fixture Simulator
- Example Programs

**CALCulate<cnum>:FSIMulator:APPLy**

**Applicable Models:** All

*(Write only) Copies the draft fixture to active fixture. Exact same functionality as CALC:FSIM:DRAFT:APPL*

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Not applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example</td>
<td>CALC:FSIM:APPL</td>
</tr>
<tr>
<td></td>
<td>calculate:fsimulatorapply</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:FSIMulator:CIRCuit:CATalog?**

**Applicable Models:** All

*(Read only) Returns a comma-separated list of circuit numbers created in the draft circuit.*

<table>
<thead>
<tr>
<th>Parameters</th>
<th>&lt;cnum&gt; Any existing channel number. If unspecified, value is set to 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>CALC:FSIM:CIRC:CAT?</td>
</tr>
<tr>
<td></td>
<td>calculate2:fsimulator:circuit:catalog?</td>
</tr>
<tr>
<td>Query Syntax</td>
<td>CALCulate&lt;cnum&gt;:FSIMulator:CIRCuit:CATalog?</td>
</tr>
<tr>
<td>Return Type</td>
<td>comma-separated string</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

3440
CALCulate<cnum>:FSIMulator:CIRCuit<circN>:DEVice:PORTs? <port list - in, out>

Applicable Models: All

(Read only) Returns the Device ports for the specified circuit.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <circN> Circuit Number

Examples

- CALC:FSIM:CIRC:DEV:PORT?
- calculate2:fsimulator:circuit:device:ports?

Query Syntax

CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<circN>:DEVice:PORTs?

Return Type Numeric

Default Not Applicable

CALCulate<cnum>:FSIMulator:CIRCuit<circN>:EMBED:TYPE?

Applicable Models: All

(Read only) Returns whether the circuit is embedded or de-embedded. Valid for all block Types

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <circN> Circuit Number

Examples

- CALC:FSIM:CIRC:EMBED:TYPE?
- calculate2:fsimulator:circuit:embed:type?

Query Syntax

CALCulate<cnum>:FSIMulator:CIRCuit<circN>:EMBED:TYPE?

Return Type character

Default Not Applicable

CALCulate<cnum>:FSIMulator:CIRCuit<circN>:FILE?
Applicable Models: All

(Read only) Returns the filename. Valid blocks: FILE, D4PFile, FGLoop

The block is create using CALCulate<cnm>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

Parameters

- `<cnm>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

Examples

- CALC:FSIM:CIRC:FILE?
- calculate2:fsimulator:circuit:file?

Query Syntax

CALCulate<cnm>:FSIMulator:CIRCuit<circN>:FILE?

Return Type

string

Default

Not Applicable

---

CALCulate<cnm>:FSIMulator:CIRCuit<circN>:FILE:EXTRapolate?

Applicable Models: All

(Read only) Returns whether or not extrapolation is allowed (TRUE/FALSE). Valid blocks: FILE, D4PFile, FGLoop.

The block is create using CALCulate<cnm>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

Parameters

- `<cnm>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

Examples

- CALC:FSIM:CIRC:FILE:EXTR?
- calculate2:fsimulator:circuit:file:extrapolate?

Query Syntax

CALCulate<cnm>:FSIMulator:CIRCuit<circN>:FILE:EXTRapolate?

Return Type

Boolean

Default

Not Applicable
CALCulate<cnum>:FSIMulator:CIRCuit<circN>:FILE:MODify?

**Applicable Models:** All

*(Read-only)* Returns the current state of the Modify setting (NONE, NREFlect, or NXTalk). Set the Modify state using the CALCulate:FSIMulator:DRAFt:CIRCuit:FILE:MODify command. Learn more.

The block is created using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<n>:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

**Examples**
- `CALC:FSIM:CIRC:PAR:C?`
- `calculate2:fsimulator:circuit:file:modify?`

**Query Syntax**
- `CALCulate<cnum>:FSIMulator:CIRCuit<circN>:FILE:MODify?`

**Return Type**
- Enumeration
  - Default: NONE

CALCulate<cnum>:FSIMulator:CIRCuit<circN>:PARameter:C?

**Applicable Models:** All

*(Read only)* Returns the capacitance (C) circuit element for the specified circuit. Valid blocks: RLGLoop, RCGLoop, CGGLoop, SLPC, PCBS, SCPL, PCP, PCPC, PCSC, SLPL, PLPL.

The block is created using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit<n>:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

**Examples**
- `CALC:FSIM:CIRC:PAR:C?`
- `calculate2:fsimulator:circuit:parameter:C?`

**Query Syntax**
- `CALCulate<cnum>:FSIMulator:CIRCuit<circN>:PARameter:C?`

**Return Type**
- Numeric
## Applicable Models: All

(Read only) Returns the capacitance (C2) circuit element for the specified circuit. Valid blocks: SLPC, PCSL, PLSC, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

### Parameters
- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<circN>** Circuit Number

### Examples
- `CALC:FSIM:CIRC:PAR:C2?`
- `calculate2:fsimulator:circuit:parameter:C2?`

### Query Syntax
- `CALCulate<cnum>:FSIMulator:CIRCuit<circN>:PARameter:C2?`

### Return Type
- Numeric

### Default
- 0

---

## Applicable Models: All

(Read only) Returns port extension delay in time. Valid blocks: ILIN.

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

### Parameters
- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<circN>** Circuit Number

### Examples
- `CALC:FSIM:CIRC:PAR:DEL?`
- `calculate2:fsimulator:circuit:parameter:delay?`

### Query Syntax
- `CALCulate<cnum>:FSIMulator:CIRCuit<circN>:PARameter:DEL?`
**CALCulate<cnum>:FSIMulator:CIRCuit<circN>:PARameter:G?**

**Applicable Models:** All

*(Read only)* Returns the conductance (G) circuit element for the specified circuit. Valid blocks: RLGLoop, RCGLoop, CGGLoop, SLPC, PCSL, PLSC, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL.

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

**Examples**

- `CALC:FSIM:CIRC:PAR:G?`
- `calculate2:fsimulator:circuit:parameter:g?`

**Query Syntax**

CALCulate<cnum>:FSIMulator:CIRCuit<circN>:PARameter:G?

**Return Type** Numeric

**Default** 0

---

**CALCulate<cnum>:FSIMulator:CIRCuit<circN>:PARameter:G2?**

**Applicable Models:** All

*(Read only)* Returns the conductance (G2) circuit element for the specified circuit. Valid blocks: SLPC, PCSL, PLSC, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL.

The block is create using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number
### CALCulate<cnum>:FSIMulator:CIRCuit<circN>:PARameter:G2?

**Examples**

CALC:FSIM:CIRC:PAR:G2?

calculate2:fsimulator:circuit:parameter:g2?

**Query Syntax**

CALCulate<cnum>:FSIMulator:CIRCuit<circN>:PARameter:G2?

**Return Type**

Numeric

**Default**

0

---

### CALCulate<cnum>:FSIMulator:CIRCuit<circN>:PARameter:L?

#### Applicable Models:

All

*(Read only)* Returns the inductance (L) circuit element for the specified circuit. Valid blocks: RLGLoop, RCGLoop, CGGLoop, SLPC, PCSL, SCPL, PLSC, SCPC, PCSC, SLPL, PLSL.

The block is created using CALCulate<cnum>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count>. Setting a value not used by the selected circuit will have no effect. Learn more.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

**Examples**

CALC:FSIM:CIRC:PAR:L?

calculate2:fsimulator:circuit:parameter:l?

**Query Syntax**

CALCulate<cnum>:FSIMulator:CIRCuit<circN>:PARameter:L?

**Return Type**

Numeric

**Default**

0

---

### CALCulate<cnum>:FSIMulator:CIRCuit<circN>:PARameter:L2?

**Examples**

CALC:FSIM:CIRC:PAR:L2?

**Default**

0

---

3446
Applicable Models: All

(Read only) Returns the inductance (L2) circuit element for the specified circuit. Valid blocks: SLPC, PCSL, PLSC, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL.

The block is created using CALCulate<cn>:FSIMulator:DRAFt:CIRCuit<n>:ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cn>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

**Examples**

```
CALC:FSIM:CIRC:PAR:L2?
CALCulate2:fsimulator:circuit:parameter:l2?
```

**Query Syntax**

```
CALCulate<cn>:FSIMulator:CIRCuit<circN>:PARameter:L2?
```

**Return Type**

Numeric

**Default**

0

CALCulate<cn>:FSIMulator:CIRCuit<circN>:PARameter:LOSS:VALue?

Applicable Models: All

(Read only) Returns the loss at DC. Valid blocks: ILIN.

The block is created using CALCulate<cn>:FSIMulator:DRAFt:CIRCuit<n>:ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- `<cn>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

**Examples**

```
CALC:FSIM:CIRC:PAR:LOSS:VAL?
CALCulate2:fsimulator:circuit:parameter:loss:value?
```

**Query Syntax**

```
CALCulate<cn>:FSIMulator:CIRCuit<circN>:PARameter:LOSS:VALue?
```

**Return Type**

Numeric

**Default**

0
CALCulate\(<\text{cnum}>\):FSIMulator:CIRCuit\(<\text{circN}>\):PARameter:R?

**Applicable Models:** All

*(Read only)* Returns the resistance (R) circuit element for the specified circuit. Valid blocks: RLGLoop, RCGLoop, CGGLoop, SLPC, PCSL, PLSC, PLPC, SCPC, PCSC, SLPL, PLSL.

The block is create using CALCulate\(<\text{cnum}>\):FSIMulator:DRAFt:CIRCuit\([n]\):ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- \(<\text{cnum}>\): Any existing channel number. If unspecified, value is set to 1
- \(<\text{circN}>\): Circuit Number

**Examples**

CALC:FSIM:CIRC:PAR:R?
calculate2:fsimulator:circuit:parameter:r?

**Query Syntax**

CALCulate\(<\text{cnum}>\):FSIMulator:CIRCuit\(<\text{circN}>\):PARameter:R?

**Return Type**

Numeric

**Default**

0

---

CALCulate\(<\text{cnum}>\):FSIMulator:CIRCuit\(<\text{circN}>\):PARameter:R2?

**Applicable Models:** All

*(Read only)* Returns the resistance (R2) circuit element for the specified circuit. Valid blocks: SLPC, PCSL, PLSC, SCPL, PLPC, SCPC, PCSC, SLPL, PLSL

The block is create using CALCulate\(<\text{cnum}>\):FSIMulator:DRAFt:CIRCuit\([n]\):ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

- \(<\text{cnum}>\): Any existing channel number. If unspecified, value is set to 1
- \(<\text{circN}>\): Circuit Number

**Examples**

CALC:FSIM:CIRC:PAR:R?
calculate2:fsimulator:circuit:parameter:r2?

**Query Syntax**

CALCulate\(<\text{cnum}>\):FSIMulator:CIRCuit\(<\text{circN}>\):PARameter:R2?

**Return Type**

Numeric

**Default**

0
CALCulate<cn>:FSIMulator:CIRCuit<cn>:PARameter:RIN?

**Applicable Models:** All

*(Read only)* Returns the resistance (R) value at input of transformer (real only). Valid blocks: TRANs

The block is created using CALCulate<cn>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>,<fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cn&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1</td>
</tr>
<tr>
<td>&lt;cn&gt;</td>
<td>Circuit Number</td>
</tr>
</tbody>
</table>

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:DRAF:CIRC:PAR:RIN?</td>
<td></td>
</tr>
<tr>
<td>calculate2:fsimulator:draft:circuit:parameter:rin?</td>
<td></td>
</tr>
</tbody>
</table>

**Query Syntax**

CALCulate<cn>:FSIMulator:CIRCuit<cn>:PARameter:RIN?

**Return Type**

Numeric

**Default**

0

---

CALCulate<cn>:FSIMulator:CIRCuit<cn>:PARameter:ROUT?

**Applicable Models:** All

*(Read only)* Returns the resistance (R) value at output of transformer (real only). Valid blocks: TRANs

The block is created using CALCulate<cn>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>,<fixt port count>. Setting a value not used by the selected circuit will have no affect. Learn more.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cn&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1</td>
</tr>
<tr>
<td>&lt;cn&gt;</td>
<td>Circuit Number</td>
</tr>
</tbody>
</table>

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:CIRC:PAR:ROUT?</td>
<td></td>
</tr>
<tr>
<td>calculate2:fsimulator:circuit:parameter:rout?</td>
<td></td>
</tr>
</tbody>
</table>

**Query Syntax**

CALCulate<cn>:FSIMulator:CIRCuit<cn>:PARameter:ROUT?

**Return Type**

Numeric

**Default**

0
### CALCulate<nun>:FSIMulator:CIRCuit<circN>:PARameter:Z0?

**Applicable Models:** All

*(Read only)* Returns the Impedance (Z0) for the ideal block (scalar value). Valid blocks: ILIN.

The block is create using CALCulate<nun>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;nun&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1</td>
</tr>
<tr>
<td>&lt;circN&gt;</td>
<td>Circuit Number</td>
</tr>
</tbody>
</table>

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:FSIM:CIRC:PAR:Z0?</code></td>
<td></td>
</tr>
<tr>
<td><code>calculate2:fsimulator:circuit:parameter:z0?</code></td>
<td></td>
</tr>
</tbody>
</table>

**Default**

0

### CALCulate<nun>:FSIMulator:CIRCuit<circN>:STATe?

**Applicable Models:** All

*(Read only)* Returns the circuit ON/OFF. Valid for all block types

The block is create using CALCulate<nun>:FSIMulator:DRAFt:CIRCuit[n]:ADD <block type>, <fixt port count> . Setting a value not used by the selected circuit will have no affect. Learn more.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;nun&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1</td>
</tr>
<tr>
<td>&lt;circN&gt;</td>
<td>Circuit Number</td>
</tr>
</tbody>
</table>

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:FSIM:CIRC:STAT?</code></td>
<td></td>
</tr>
<tr>
<td><code>calculate2:fsimulator:circuit:state?</code></td>
<td></td>
</tr>
</tbody>
</table>

**Default**

OFF
CALCulate<cnum>:FSIMulator:CIRCuit<circN>:VNA:PORTs? <port list>

**Applicable Models:** All

*(Read only)* Returns the VNA ports for the specified circuit.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<circN>` Circuit Number

**Examples**
- CALC:FSIM:CIRC:VNA:PORT?
- calculate2:fsimulator:circuit:vna:ports?

**Query Syntax**
- CALCulate<cnum>:FSIMulator:CIRCuit<circN>:VNA:PORTs?

**Return Type** Numeric

**Default** Not Applicable

CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:DELay

**Applicable Models:** All

*(Read-Write)* Returns the delay value in time at the specified port.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<pnum>` Port Number that will receive the delay setting. If unspecified, value is set to 1.
- `<value>` The port extension in seconds; may include suffix. Choose a number between: -1E18 and 1E18 e.

**Examples**
- CALC:FSIM:EXT:PORT1:DEL?
- calculate2:fsimulator:extension:port2:delay?

**Query Syntax**
- CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:DELay?

**Return Type** Numeric

**Default** 0

CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:DISTance:UNIT
Applicable Models: All

(Read-only) Returns the units for specifying port extension delay in physical length (distance).

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<pnum>` Port Number. This number is ignored and all ports have the same setting.
- `<char>` Units for delay in distance. Choose from:
  - METer
  - FEET
  - INCH

**Examples**

```
calculate:fsimulator:extension:port2:distance:unit?
calculate:fsimulator:extension:port3:distance:unit?
```

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:EXTension:PORT:DISTance:UNIT?
```

**Return Type** Character

**Default** METer

---

**CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:DISTance:VALue**

Applicable Models: All

(Read-only) Returns the port extension delay in physical length (distance).

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<pnum>` Port Number that will receive the delay setting. If unspecified, value is set to 1.
- `<value>` Physical length of fixture of added transmission line.

**Examples**

```
CALC:FSIM:EXT:PORT1:DIST:VAL?
calculate:fsimulator:extension:port2:distance:value?
```

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:DISTance:VALue?
```

**Return Type** Numeric

**Default** 0
CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:LOSS<n>:FREQuency

Applicable Models: All

(Read-only) Returns the frequency for the Freq and Loss pair number and for the specified port number.

Learn about Loss Compensation values.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<pnum>`: Port Number that will receive the freq/loss settings. If unspecified, value is set to 1.
- `<n>`: Freq and Loss pair number. Choose from 1 or 2. If unspecified, value is set to 1.
- `<value>`: Frequency value. Choose a frequency within the frequency span of the VNA.

Examples

```
CALC:FSIM:EXT:PORT1:LOSS2:FREQ?
calculate:fsimulator:extension:port2:loss1:frequency?
```

Query Syntax

CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:LOSS<n>:FREQuency?

Return Type: Numeric

Default: 1 GHz

---

CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:LOSS<n>:VALue

Applicable Models: All

(Read-only) Returns the Loss value for the specified port number.

Learn about Loss Compensation values.

Note: This command affects ALL measurements on the specified channel.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<pnum>`: Port Number that will receive the Freq/Loss settings. If unspecified, value is set to 1.
- `<n>`: Loss "Use" number. Choose from 1 or 2. If unspecified, value is set to 1.
- `<value>`: Loss in dB. Choose a value between -90 and 90.
CALCulate\(<cnum>\):FSIMulator:EXTension:PORT\(<pnum>\):LOSS\(\langle n\rangle\):VALue?

**Query Syntax**
CALCulate\(<cnum>\):FSIMulator:EXTension:PORT\(<pnum>\):LOSS\(\langle n\rangle\):VALue?

**Return Type**
Numeric

**Default**
0

CALCulate\(<cnum>\):FSIMulator:EXTension:PORT\(<pnum>\):LOSS\(\langle n\rangle\)[:STATe]

**Applicable Models:** All

(Read-only) Returns the ON/OFF state for the Freq and Loss pair number and for the specified port number.

Learn about Loss Compensation values.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**
- \(<cnum>\) Any existing channel number. If unspecified, value is set to 1
- \(<pnum>\) Port Number that will receive the Freq/Loss settings. If unspecified, value is set to 1.
- \(<n>\) Freq and Loss pair. Choose from 1 or 2. If unspecified, value is set to 1.
- \(<value>\) State of Freq and Loss values for port extension.

**Examples**
- Specified Freq and Loss values are OFF
- Specified Freq and Loss values are ON

CALC:FSIM:EXT:PORT:LOSS1:STAT?
calculate:fsimulator:extension:port2:loss2:state?

**Query Syntax**
CALCulate\(<cnum>\):FSIMulator:EXTension:PORT\(<pnum>\):LOSS\(\langle n\rangle\)[:STATe]?

**Return Type**
Boolean

**Default**
OFF

CALCulate\(<cnum>\):FSIMulator:EXTension:PORT\(<pnum>\):DCLoss:VALue

3454
**Applicable Models:** All

*(Read-only)* Returns the Port Loss at DC value for the specified port number.

Learn about Loss Compensation values.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port number to receive Loss value. If unspecified, value is set to 1.
- `<value>` Loss in dB. Choose a value between -90 and 90

**Examples**

```
CALC:FSIM:EXT:PORT:DCL:VAL?
calculate:fsimulator:extension:port2:dcloss:value?
```

**Query Syntax**

`CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:DCLoss:VALue?`

**Return Type**

Numeric

**Default**

0

---

**CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:MEDium**

**Applicable Models:** All

*(Read-only)* Returns the media type of the added fixture or transmission line.

See also SENS:CORR:EXT:PORT:SYSMedia

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number. This number is ignored and all ports have the same setting.
- `<char>` Medium type. Choose from:

  - COAX
  - WAVEguide
**Examples**

`CALC:FSIM:EXT:PORT:MED?`

`calculate:fsimulator:extension:port2:medium?`

**Query Syntax**

`CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:MEDium?`

**Return Type**

Character

**Default**

`COAX`

---

`CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:WAVEguide:FCUToff`

**Applicable Models:** All

*(Read-only)* Returns the cutoff (minimum) frequency of the added waveguide fixture or transmission line.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number for which media type is being set. If unspecified, value is set to 1.
- `<value>` Cutoff frequency in Hz.

This value is ignored when `CALC:FSIM:EXT:PORT:MED` is set to **COAX** for the same port.

**Examples**

`CALC:FSIM:EXT:PORT:WAVE:FCUT?`

`calculate:fsimulator:extension:port2:waveguide:cutoff?`

**Query Syntax**

`CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:WAVEguide:FCUToff?`

**Return Type**

Numeric

**Default**

System Media Cutoff Frequency

---

`CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:WAVEguide:COUPlE`
**Applicable Models:** All

*(Read-only)* Returns the state of coupling with the system Velocity Factor value. Learn more.

**Note:** This command potentially affects ALL measurements on the VNA.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number. This number is ignored and all ports have the same setting.
- `<bool>` Coupling state. Choose from:
  - **ON** (or 1) - Velocity Factor is coupled with the system setting.
  - **OFF** (or 0) - Velocity Factor is NOT coupled with the system setting.

**Examples**

```
CALC:FSIM:EXT:PORT:WAVE:COUP?
calculate:fsimulator:extension:port2:waveguide:couple?
```

**Query Syntax**

`CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:WAVEguide:COUPle?`

**Return Type** Boolean

**Default** 1 or ON (Coupled)

---

**CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:VELocity:FACtor**

**Applicable Models:** All

*(Read-only)* Returns the velocity factor of the fixture or added transmission line.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number for which velocity factor is being set. If unspecified, value is set to 1.
- `<value>` Velocity Factor.

Set `SENS:CORR:EXT:PORT:SYSV` to use the system velocity factor.

**Examples**

```
CALC:FSIM:EXT:PORT:VEL:FAC?
calculate:fsimulator:extension:port2:velocity:factor?
```

**Query Syntax**

`CALCulate<cnum>:FSIMulator:EXTension:PORT<pnum>:VELocity:FACtor?`
**Return Type**  Numeric  
**Default**  System Velocity Factor

CALCulate\<cnum\>:FSIMulator:EXTension:PORT\<pnum\>:VELocity:COUPle

**Applicable Models:** All

*(Read-only)* Returns the state of coupling with the system Velocity Factor value. Learn more.

**Note:** This command potentially affects ALL measurements on the VNA.

**Parameters**

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<pnum>`  Port Number. This number is ignored and all ports have the same setting.
- `<bool>`  Coupling state. Choose from:

  - **ON** (or 1) - Velocity Factor is coupled with the system setting.
  - **OFF** (or 0) - Velocity Factor is NOT coupled with the system setting.

**Examples**

```
CALC:FSIM:EXT:PORT:VEL:COUP?
calculate:fsimulator:extension:port2:velocity:couple?
```

**Query Syntax**  
CALCulate\<cnum\>:FSIMulator:EXTension:PORT\<pnum\>:VE locality:COUPle?

**Return Type**  Boolean

**Default**  1 or ON (Coupled)
Applicable Models: All

(Read-only) Returns the state of port extension for the specified port.

Note: This command potentially affects ALL measurements on the VNA.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number for which port extension is being set. If unspecified, value is set to 1.
- `<bool>` Port extension state. Choose from:
  - **ON** (or 1) - Port extension is on
  - **OFF** (or 0) - Port extension is off.

Examples

```
CALC:FISM:EXT:PORT:STAT?
calculate:fimulator:extension:port2:state?
```

Query Syntax

CALCulate `<cnum>`:FSimulator:EXTension:PORT `<pnum>`[:STATe]?

Return Type

Boolean

Default

0

---

CALCulate `<cnum>`:FSIMulator:POWer:COMPensate:MODE `<char>`

Applicable Models: All

(Read-Write) Adjust the source power compensation for gain/loss through “ALL” fixture components or through “DEEMbed” components only.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Mode selection. Choose from:
  - **ALL**
  - **DEEMbed**

Examples

```
CALC:FISM:POW:COMP:MODE ALL
calculate:fimulator:power:compensate:mode deembed
```
**Query Syntax** | CALCulate<cnum>:FSIMulator:POWer:COMPensate:MODE?
---|---
**Return Type** | Character
**Default** | ALL

**CALCulate<cnum>:FSIMulator:POWer:PORT<pnum>:COMPensate[:STATe] <bool>**

**Applicable Models:** All

*(Read-Wrrite)* Sets and returns on power compensation for the active fixture

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number for which port extension is being set. If unspecified, value is set to 1.
- `<bool>` power compensation state. Choose from:
  - **ON** (or 1) - Power compensation is on
  - **OFF** (or 0) - Power compensation is off.

**Examples**

```
CALC:FSIM:EXT:POW:PORT ON
calculate:fsimulator:power:port:compensate 0
```

**Query Syntax** | CALCulate<cnum>:FSIMulator:POWer:PORT[p]:COMPensate[:STATe]?
---|---
**Return Type** | Boolean
**Default** | 0

**CALCulate<cnum>:FSIMulator:SAVE <snpName>**
Applicable Models: All

(Write only) Saves SNP file corresponding to the entire active fixture.

**Parameters**

<snpName> String - snp file name. Appends ".sNp" depending on num ports

**Example**

```
CALC:FSIM:SAVE "myFixture"
```

```
calculate:fsimulator:save "D:\myfixture"
```

**Return Type** Not applicable

**Default** Not applicable

---

CALCulate<cnum>:FSIMulator:SECTion:CIRCuit:ENABle?

Applicable Models: All

(Read only) Returns all circuits ON/OFF for all ports.

**Parameters**

<cnum> Any existing channel number. If unspecified, value is set to 1

**Examples**

```
CALC:FSIM:SECT:CIRC:ENAB?
```

```
calculate2:fsimulator:section:circuit:enable?
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:SECTion:CIRCuit:ENABle?

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** OFF

---

CALCulate<cnum>:FSIMulator:SECTion:EXTension:ENABle?
Applicable Models: All

(Read only) Returns the port extensions ON/OFF for all ports.

Parameters
<cnm> Any existing channel number. If unspecified, value is set to 1

Examples
CALC:FSIM:DRAF:SECT:EXT:ENAB?
calculate2:fsimulator:draft:section:extension:enable?

Query Syntax
CALCulate<cnm>:FSIMulator:DRAFt:SECTion:EXTension:ENABle?

Return Type
Boolean (1 = ON, 0 = OFF)

Default
OFF

Applicable Models: All

(Read only) Returns all ZConversions ON/OFF (both SE and BAL).

Parameters
<cnm> Any existing channel number. If unspecified, value is set to 1

Examples
CALC:FSIM:DRAF:SECT:ZCON:ENAB?
calculate2:fsimulator:draft:section:zconversion:enable?

Query Syntax
CALCulate<cnm>:FSIMulator:DRAFt:SECTion:ZCONversion:ENABle?

Return Type
Boolean (1 = ON, 0 = OFF)

Default
OFF

CALCulate<cnm>:FSIMulator:TOPology:LOAD <fileName>
(Write only) Loads an active fixture from a file.

**Parameters**
- `<fileName>`: Fixture topology file name.

**Example**
```
CALC:FSIM:TOP:LOAD "example.topo"
calculate:fsimulator:topology:load "example.topo"
```

**Return Type**: Not applicable

**Default**: Not applicable

---

CALCulate<cnum>:FSIMulator:TOPology:SAVE <fileName>

(Write only) Saves the current active fixture topology to a file.

**Parameters**
- Not applicable

**Example**
```
CALC:FSIM:TOP:SAVE "example.topo"
calculate:fsimulator:topology:save "example.topo"
```

**Return Type**: Not applicable

**Default**: Not applicable

---

CALCulate<cnum>:FSIMulator:ZCONversion:DIFFerential:STATe
Applicable Models: All

(Read Only) Returns the differential port impedance conversion function ON/OFF.

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

CALC:FSIM:ZCON:DIFF:STAT?
calculate2:fsimulator:zconversion:DIFFerential:state?

Query Syntax

CALCulate<cnum>:FSIMulator:ZCONversion:DIFFerential:STATe?

Return Type

Boolean

0 Differential port impedance conversion OFF

1 Differential port impedance conversion ON

Default 0

CALCulate<cnum>:FSIMulator:ZCONversion:DIFFerential:BPORt<pnum>:COMPLex

Applicable Models: All

(Read-Only) Returns the complex impedance value for the differential port impedance conversion function.

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum> Balanced port number. Choose from 1 to 999.

Note: See Balanced port versus Logical port.

Examples

CALC:FSIM:ZCON:DIFF:BPOR2:COMPL?

Query Syntax

CALCulate<cnum>:FSIMulator:ZCONversion:DIFFerential:BPORt<pnum>:COMPLex?

Return Type

Numeric <real>,<img>

Real part of the Impedance value in Units, Imaginary part of the Impedance value in Units.
CALCulate<cnum>:FSIMulator:ZCONversion:DIFFerential:BPORt<pnum>:SCALar

**Applicable Models:** All

(Read Only) Returns the real part of impedance value for the differential port impedance conversion function.

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>`: Balanced port number. Choose from 1 to 999.

**Note:** See Balanced port versus Logical port.

**Examples**

```
CALC:FSIM:ZCON:DIFF:BPOR2:SCAL 200,10
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:ZCONversion:DIFFerential:BPORt<pnum>:SCALer?

**Return Type**

- Numeric
- Real part of the Impedance value in Units

**Default**

0

---

CALCulate<cnum>:FSIMulator:ZCONversion:DIFFerential:LPORt<pnum>:COMPLex
Applicable Models: All

(Read Only) Returns the complex impedance value for the common mode port impedance conversion function.

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
<pnum> Logical port number. Choose from 1 to 999.

Note: See Balanced port versus Logical port.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>calculate2:fsimulator:zconversion:differential:lport2:complex?</td>
<td>Real part of the Impedance value in Units, Imaginary part of the Impedance value in Units</td>
</tr>
<tr>
<td>Default</td>
<td>0,0</td>
</tr>
</tbody>
</table>

CALCulate<cnum>:FSIMulator:ZCONversion:DIFFerential:LPORt<pnum>:SCALar

Applicable Models: All

(Read Write) Returns the real part of impedance value for the differential mode port impedance conversion function.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
<pnum> logical port number. Choose from 1 to 999.

Note: See Balanced port versus Logical port.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>calculate2:fsimulator:zconversion:differential:lport2:scala?</td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>3466</td>
</tr>
</tbody>
</table>
**Query Syntax**

CAlCulate<cnum>:FSIMulator:ZCONversion:DIfferential:LPORt<pnum>:SCALer?

**Return Type**

Numeric

Real part of the Impedance value in Units

**Default**

0

**CALCulate<cnum>:FSIMulator:ZCONversion:COMMonmode:STATe**

**Applicable Models:** All

(Read Only) Returns the common mode port impedance conversion function ON/OFF.

**Parameters**

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

**Examples**

CALC:FSIM:ZCON:COMM:STAT?
calculate2:fsimulator:zconversion:commonmode:state?

**Query Syntax**

CAlCulate<cnum>:FSIMulator:COMMonmode:DIfferential:STATe?

**Return Type**

Boolean

0 Common mode port impedance conversion OFF

1 Common mode port impedance conversion ON

**Default**

0

**CALCulate<cnum>:FSIMulator:ZCONversion:COMMonmode:BPORt<pnum>:COMPLex**


Applicable Models: All

(Read-Write) Returns the complex impedance value for the common mode port impedance conversion function.

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>` Balanced port number. Choose from 1 to 999.

Note: See Balanced port versus Logical port.

Examples

- `CALC:FSIM:ZCON:COMM:BPOR2:COMPL?`

Query Syntax

`CALCulate<cnum>:FSIMulator:ZCONversion:COMMonmode:BPORt<pnum>:COMPlex?`

Return Type

Numeric `<real>,<img>`

Real part of the Impedance value in Units, Imaginary part of the Impedance value in Units

Default

0,0

CALCulate<cnum>:FSIMulator:ZCONversion:COMMonmode:BPORt<pnum>:SCALar

Applicable Models: All

(Read Only) Returns the real part of impedance value for the common mode port impedance conversion function.

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>` Balanced port number. Choose from 1 to 999.

Note: See Balanced port versus Logical port.

Examples


Query Syntax

`CALCulate<cnum>:FSIMulator:ZCONversion:COMMonmode:BPORt<pnum>:SCALar?`
**CALCulate<cnum>:FSIMulator:ZCONversion:COMMONmode:LPORt<pnum>:COMPLex**

**Applicable Models:** All

*(Read Only)* Returns the complex impedance value for the common mode port impedance conversion function.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>` Logical port number. Choose from 1 to 999.

**Note:** See Balanced port versus Logical port.

**Examples**

```
CALC:FSIM:ZCON:COMM:LPOR2:COMPL?
calculate2:fsimulator:zconversion:commonmode:lport2:complex?
```

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:ZCONversion:COMMONmode:LPORt<pnum>:COMPLex
```

**Return Type**

Numeric `<real>,<img>`

Real part of the Impedance value in Units, Imaginary part of the Impedance value in Units.

**Default**

0,0
**Applicable Models:** All

*(Read Only)* Returns the real part of impedance value for the common mode port impedance conversion function.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>` logical port number. Choose from 1 to 999.

**Note:** See Balanced port versus Logical port.

**Examples**

```
CALC:FSIM:ZCON:COMM:LPOR2:SCAL?
calculate2:fsimulator:zconversion:commonmode:lport2:scalar?
```

**Query Syntax**

CALCulate`<cnum>`:FSIMulator:ZCONversion:COMMonmode:LPORt`<pnum>`:SCALer?

**Return Type** Numeric, Real part of the Impedance value in Units.

**Default** 0

---

**CALCulate`<cnum>`:FSIMulator:ZCONversion:SENDed:PORt`<pnum>`:STATe**

**Applicable Models:** All

*(Read Only)* Returns the single-ended impedance conversion function ON/OFF. Must also set the fixture simulator function to ON using CALC:FSIM:STAT.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>` Port number. Choose from 1 to 999.

**Examples**

```
CALC:FSIM:ZCON:SEND:PORT3:STAT?
calculate2:fsimulator:zconversion:sended:port2:state?
```

**Query Syntax**

CALCulate`<cnum>`:FSIMulator:ZCONversion:SENDed:PORt`<pnum>`:STATe?

**Return Type** Boolean

0 single-ended port impedance conversion OFF

1 single-ended port impedance conversion ON
CALCulate<cnum>:FSIMulator:ZCONversion:SENDed:PORt<pnum>:COMPLex

Applicable Models: All

(Read Only) Returns the complex impedance value for the single-ended port impedance conversion function.

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>`: Pport number. Choose from 1 to 999.

Examples

```
CALC:FSIM:ZCON:SEND:POR2:COMPL?
calculate2:fsimulator:zconversion:sended:port2:complex?
```

Query Syntax

```
CALCulate<cnum>:FSIMulator:ZCONversion:SENDed:PORt<pnum>:COMPLex?
```

Return Type

Numeric `<real>`, `<img>`

Real part of the Impedance value in Units, Imaginary part of the Impedance value in Units.

Default 0,0

CALCulate<cnum>:FSIMulator:ZCONversion:SENDed:PORT<pnum>:SCALer

Applicable Models: All

(Read Only) Returns the real part of impedance value for the single-ended port impedance conversion function.

See Critical Note

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>`: Pport number. Choose from 1 to 999.

Examples

```
CALC:FSIM:ZCON:SEND:PORT:SCAL?
calculate2:fsimulator:zconversion:sended:port2:scalar?
```

Query Syntax

```
CALCulate<cnum>:FSIMulator:ZCONversion:SENDed:PORT<pnum>:SCALer?
```
<table>
<thead>
<tr>
<th><strong>Return Type</strong></th>
<th>Numeric</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
</tbody>
</table>
**Calculate:Function Commands**

These commands are **Superseded** by the `CALCulate:MEASure:FUNCtion` commands.

---

Click on a keyword to view the command details.

**see Also**

- [Example Programs](#)
- [Learn about Trace Statistics](#)
- [Synchronizing the Analyzer and Controller](#)
- [SCPI Command Tree](#)

**Critical Note:** `CALCulate` commands act on the selected measurement. You can select one measurement for each channel using `Calc:Par:MNUM` or `Calc:Par:Select`. Learn more.

```
CALCulate<cnum>:FUNCtion:DATA?
```

3473
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-only)* Returns the trace statistic data for the selected statistic type for the specified channel. Select the type of statistic with `CALC:FUNC:TYPE`.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

**Return Type** Depends on `FORM:DATA`

**Example**

```
CALCulate2:FUNCtion:DATA?
```

**Default** Not applicable

---

**CALCulate<cnum>:FUNCtion:DOMain:USER[:RANGE] <range>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the range used to calculate trace statistics. Each channel has 16 user ranges. The x-axis range is specified with the `CALC:FUNC:DOM:USER:START` and `STOP` commands.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<range>` Range number. Choose from: 0 to 16
  - 0 is Full Span of the current x-axis range
  - 1 to 16 are user-specified ranges

**Examples**

- `CALC:FUNC:DOM:USER 4`
- `calculate2:function:domain:user:range 0`

**Query Syntax** `CALCulate<cnum>:FUNCtion:DOMain:USER[:RANGe]?`

**Return Type** Numeric

**Default** 0 - Full Span

---

**CALCulate<cnum>:FUNCtion:DOMain:USER:STARt <range>, <start>**
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the start of the specified user-domain range.

To apply this range, use **CALC:FUNC:DOM:USER**

To set the stop of the range, use **CALC:FUNC:DOM:USER:STOP**.

See Critical Note

**Note:** This command does the same as **CALC:MARK:FUNC:DOM:USER:STAR**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <code>&lt;cnum&gt;</code> is set to 1.</td>
</tr>
<tr>
<td><code>&lt;range&gt;</code></td>
<td>Range number that will receive the start value. Choose an integer between 1 and 16</td>
</tr>
<tr>
<td><code>&lt;start&gt;</code></td>
<td>Start value of the specified range. Choose a real number between: the analyzer's <strong>Minimum</strong> and <strong>Maximum</strong> x-axis value.</td>
</tr>
</tbody>
</table>

**Examples**

```
CALC:FUNC:DOM:USER:STAR 1,1e9
calculate2:function:domain:user:start 2,2e9
```

**Query Syntax**

```
CALCulate<cnum>:FUNCtion:DOMain:USER:STARt? <range>
```

**Return Type**

Numeric

**Default**

The analyzer's **Minimum** x-axis value

---

**CALCulate<cnum>:FUNCtion:DOMain:USER:STOP** `<range>`, `<stop>`

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the stop value of the specified user-domain range.

To apply this range, use **CALC:FUNC:DOM:USER**.

To set the start of the range, use **CALC:FUNC:DOM:USER:START**

See Critical Note

**Note:** This command does the same as **CALC:MARK:FUNC:DOM:USER:STOP**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <code>&lt;cnum&gt;</code> is set to 1.</td>
</tr>
</tbody>
</table>
Range number that will receive the stop value. Choose an integer between 1 and 16.

Stop value of the specified range. Choose a real number between: the analyzer's Minimum and Maximum x-axis value.

**Examples**

```
CALC:FUNC:DOM:USER:STOP 4,5e9
```

```
calculate2: function: domain: user: stop 3,8e9
```

**Query Syntax**

```
CALCulate<cnum>:FUNCtion:DOMain:USER:STOP? <range>
```

**Return Type**

Numeric

**Default**

The analyzer's Maximum x-axis value

---

**CALCulate<cnum>:FUNCtion:EXECute**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Write-only)* For the active trace of specified channel, executes the statistical analysis specified by the **CALC:FUNC:TYPE** command.

**See Critical Note**

**Parameters**

**<cnum>**

Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

**Examples**

```
CALC:FUNC:EXEC
```

```
calculate2: function: execute
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**CALCulate<cnum>:FUNCtion:STATistics[:STATe] <ON|OFF>**
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Displays and hides the trace statistics (peak-to-peak, mean, standard deviation) on the screen.

The analyzer will display either measurement statistics or Filter Bandwidth statistics; not both.

**See Critical Note**

**Parameters**

- `<cnum>`  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<ON|OFF>` ON - Displays trace statistics
  OFF - Hides trace statistics

**Examples**

```
CALC:FUNC:STAT ON
calculate2:function:statistics:state off
```

**Query Syntax**

CALCulate<cnum>:FUNCtion:STATistics[:STATe]?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF (0)

---

**CALCulate<cnum>:FUNCtion:TYPE <char>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets statistic TYPE that you can then query using CALC:FUNCtion:DATA?.

**Note:** This command affects only the selected measurement on the specified channel.

**See Critical Note**

**Parameters**

- `<cnum>`  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>`  Choose from:
  - **PTPeak** - the difference between the max and min data points on the trace.
  - **STDEV** - standard deviation of all data points on the trace
  - **MEAN** - mean (average) of all data points on the trace
  - **MIN** - lowest data point on the trace
MAX - highest data point on the trace

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:FUNC:TYPE PTP</code></td>
</tr>
<tr>
<td><code>calculate2:function:type stdev</code></td>
</tr>
</tbody>
</table>

**Query Syntax**

CALCulate<enum>:FUNCtion:TYPE?

**Return Type**

Character

**Default**

PTPeak
Reads Gain Compression data from the current Gain Compression acquisition.

These commands are **Superseded** by the `CALCulate:MEASure:GCData` commands.

### CALCulate:GCData:

- `DATA?`
- `IMAG?`
- `ITERations?`
- `REAL?`

Click on a keyword to view the command details.

**Other Gain Compression commands**

The calibration commands listed in this topic are supplemental to the Guided Cal commands.

- `CALC:CUSTom:DEFine` - creates a gain compression measurement.
- `SENS:GCSetup` - Most Gain Compression settings.
- `CALC:GCMeas:ANAL` - Gain Compression Analysis settings
- Gain compression data can also be saved to a *.csv file. [Learn how.](#)

**See Also**

- Example Programs
- Learn about Gain Compression Application
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

---

**CALCulate<ch>:GCData:DATA? <param>**

**Applicable Models:** N522xB, N524xB, M9485A, E5080A

**(Read-Only)** Returns measurement data at all frequency and power data points for GCA SMART sweeps and 2D sweeps.
- When using SMART sweep, ALL data is returned including ALL background iteration sweeps. Use CALC:GCD:ITER to determine the number of iteration sweeps. The number of data points that are returned is always going to be number of frequency points times the number of iteration sweeps.

- When using 2D sweeps, ALL data is returned. The number of data points returned / freq may vary. Learn more.

Use Calc:Data? to return just the displayed data results (not the background sweeps).

A compression parameter must be present. Learn more.

The format of the data is the same as the format of the measurement that you select using Calc:Par:Select. If the measurement is scalar, than one number is returned per sweep per data point. If complex (such as Smith Chart format) than both real and imaginary numbers are returned.

If correction is on, corrected data are returned. Otherwise, raw data are returned.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<param>` (String) Parameter to read. NOT Case-sensitive. The specified parameter need NOT be displayed or selected. However, a compression parameter must be present. Learn more.

Choose from:

- "pin" - (CompIn21) Input power at the compression point.
- "pout" - (CompOut21) Output power at the compression point.
- "gain" - (CompGain21) Device gain (S21) at the compression point.
- "inputmatch" - (CompS11) Input match at the compression point.
- "DeltaGain" - (DeltaGain21) Measured Gain (watts) / Ref Gain (watts). Learn more.
- "AI1" and "AI2" - ADC measurements at the specified compression level. Learn more.

Learn more about GCA parameters.

**Examples**

```plaintext
data = CALC:GCD:DATA? "pin"
data = calculate:gcdata:data? "pout"
```

**Return Type**

- Array of data

**Default**

- Not Applicable
CALCulate<ch>:GCData:IMAG? <char>, <dpoint>, <param>

Applicable Models: N522xB, N524xB, M9485A, E5080A

(Read-Only) For a specified data point, returns the imaginary part of the specified Gain Compression data. If correction is on, corrected data are returned. Otherwise, raw data are returned. Can be used with Smart and 2D sweeps.

- For SMART sweep, the number of data points that are returned is always going to be the number of iteration sweeps. Use **CALC:GCD:ITER** to determine the number of iteration sweeps.
- For 2D sweeps, the number of data points returned / freq may vary. Learn more.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Choose from:
  - **FREQuency** - for the specified frequency data point, returns all of the measured data for each power stimulus.
  - **POWer** - for the specified power data point, returns all of the measured data for each frequency stimulus.
- `<dPoint>` Data point (FREQ or POWer) for which data is returned.
- `<param>` (String) Parameter to read. NOT Case-sensitive. The specified parameter need NOT be displayed. However, a compression parameter must be present. Learn more.
  - "pin" - (ComplIn21) Input power at the compression point.
  - "pout" - (CompOut21) Output power at the compression point.
  - "gain" - (CompGain21) Device gain (S21) at the compression point.
  - "inputmatch" - (CompS11) Input match at the compression point.
  - "DeltaGain" - (DeltaGain21) Measured Gain (watts) / Ref Gain (watts). Learn more.
  - "AI1" and "AI2" - ADC measurements at the specified compression level. Learn more.

**Examples**

For the fifth frequency data point, returns 'Power Output' imaginary (phase) data from all power stimulus values.

For SmartSweep, if there are 30 power sweep points, 30 values are returned.

For 2D sweeps, 30 or 31 power sweep points may be returned. Learn more.
CALCulate<cnum>:GCData:ITERations?

**Applicable Models:** N522xB, N524xB, M9485A, E5080A

*(Read-only)* In a SMART sweep, returns the max number of iterations that it took for ALL frequencies to converge. Use this number to determine the size of the block data that is returned from Gain Compression SMART sweep data queries.

For a 2D sweep, returns the number of power points.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

**Examples**

```
data = CALC:GCD:ITER?
```

**Return Type** Numeric

**Default** Not Applicable

---

CALCulate<ch>:GCData:REAL? <char>, <dpoint>, <param>

**Applicable Models:** N522xB, N524xB, M9485A, E5080A

*(Read-Only)* For a specified data point, returns the real part of the Gain Compression data. If correction is on, corrected data are returned. Otherwise, raw data are returned. Can be used with Smart and 2D sweeps.

- For SMART sweep, the number of data points that are returned is always going to be the number of iteration sweeps. Use **CALC:GCD:ITER** to determine the number of iteration sweeps.

- For 2D sweeps, the number of data points returned / freq may vary. Learn more.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Choose from:
  - **FREQuency** - for the specified frequency data point, returns all of the measured
data for each power stimulus.

- **POWER** - for the specified power data point, returns all of the measured data for each frequency stimulus.

<dPoint> Data point (FREQ or POWer) for which data is returned.

<param> (String) Parameter to read. NOT Case-sensitive. The specified parameter need NOT be displayed. However, a compression parameter must be present. Learn more.

- "pin" - (Compln21) Input power at the compression point.
- "pout" - (CompOut21) Output power at the compression point.
- "gain" - (CompGain21) Device gain (S21) at the compression point.
- "inputmatch" - (CompS11) Input match at the compression point.
- "DeltaGain" - (DeltaGain21) Measured Gain (watts) / Ref Gain (watts). Learn more.
- "AI1" and "AI2" - ADC measurements at the specified compression level. Learn more.

Examples

For the fifth frequency data point, returns 'Power Output' real data from all power stimulus values.

For SmartSweep, if there are 30 power sweep points, 30 values are returned.

For 2D sweeps, 30 or 31 power sweep points may be returned. Learn more.

data = CALC:GCD:REAL? FREQ,5,"pout"

Return Type Array of data

Default Not Applicable
Gain Compression Analysis Commands

Sets and reads Gain Compression Analysis controls.

These commands are **Superseded** by the \texttt{CALCulate:MEASure:GCMeas} commands.

\begin{verbatim}
CALCulate:GCMeas:ANALysis
  CWFRequency
  ENABle
  ISDisfreq
  XAXis
\end{verbatim}

Click on a keyword to view the command details.

**Other Gain Compression commands**

The calibration commands listed in this topic are supplemental to the Guided Cal commands.

- \texttt{CALC:CUSTom:DEFine} - creates a gain compression measurement.
- \texttt{SENS:GCSetup} - Most Gain Compression settings.
- \texttt{GC:DATA} - Gain Compression data commands
- Gain compression data can also be saved to a *.csv file. Learn how.

**See Also**

- Example Programs
- Learn about Compression Analysis
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

\begin{verbatim}
CALCulate<cn>:GCMeas:ANALysis:ENABle <bool>
\end{verbatim}
**Applicable Models:** N522xB, N524xB, M9485A, E5080A

*(Read-Write)* Enables and disables a compression analysis trace.

**Parameters**

- `<cnum>`: Channel number of the GCA measurement. There must be a selected measurement on that channel using `Calc:Par:Sel`. If unspecified, `<cnum>` is set to 1.
- `<bool>`: `ON` (or 1) - Enable compression analysis. `OFF` (or 0) - Disable compression analysis.

**Examples**

- `CALC:GCM:ANAL:ENAB ON`
- `calculate2:gcmeas:analysis:enable off`

**Query Syntax**

`CALCulate<cnum>:GCMeas:ANALysis:ENABle?`

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

---

**CALCulate<cnum>:GCMeas:ANALysis:CWFRequency <num>**

*(Read-Write)* Set and return the CW frequency for a compression analysis trace.

**Parameters**

- `<cnum>`: Channel number of the GCA measurement. There must be a selected measurement on that channel using `Calc:Par:Sel`. If unspecified, `<cnum>` is set to 1.
- `<num>`: CW frequency in Hz. Choose a frequency within the range of the gain compression channel.

**Examples**

- `CALC:GCM:ANAL:CWFR 1e9`
- `calculate2:gcmeas:analysis:cwfrequency 1e10`

**Query Syntax**

`CALCulate<cnum>:GCMeas:ANALysis:CWFRequency?`

**Return Type**

Numeric

**Default**

Not Applicable

---

**CALCulate<cnum>:GCMeas:ANALysis:DISCrete | ISD[:STATe] <bool>**
Applicable Models: N522xB, N524xB, M9485A, E5080A

(Read-Write) Sets and returns whether the CW frequency for the compression analysis trace can be set to only the discrete frequencies or provides interpolation.

Parameters

- `<cnum>` Channel number of the GCA measurement. There must be a selected measurement on that channel using `Calc:Par:Sel`. If unspecified, `<cnum>` is set to 1.
- `<bool>` ON (or 1) - Discrete data points only.
  OFF (or 0) - Interpolated data points.

Examples

```
CALC:GCM:ANAL:ISD ON
calculate2:gcmeas:analysis:isdisfrequency off
```

Query Syntax

`CALCulate<cnum>:GCMeas:ANALysis:ISDisfrequency?`

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF

---

CALCulate<cnum>:GCMes:ANALysis:XAXis <char>

Applicable Models: N522xB, N524xB, M9485A, E5080A

(Read-Write) Sets and returns the type of data to display on the x-axis of a compression analysis trace.

Parameters

- `<cnum>` Channel number of the GCA measurement. There must be a selected measurement on that channel using `Calc:Par:Sel`. If unspecified, `<cnum>` is set to 1.
- `<bool>` Data to display on X-axis. Choose from:
  
  - PIN - Input power to the DUT.
  - PSOure - power from the source.

Examples

```
CALC:GCM:ANAL:XAX PIN
calculate2:gcmeas:analysis:xaxis psource
```

Query Syntax

`CALCulate<cnum>:GCMeas:ANALysis:XAXis?`

Return Type

Character

Default

PIN
Group Delay Aperture Commands

Controls the Aperture setting used to make Group Delay measurements.

These commands are **Superseded** by the `CALCulate:MEASure:GDElay` commands.

<table>
<thead>
<tr>
<th>CALCulate:GDElay</th>
</tr>
</thead>
<tbody>
<tr>
<td>FREQuency</td>
</tr>
<tr>
<td>PERCent</td>
</tr>
<tr>
<td>POINts</td>
</tr>
</tbody>
</table>

Click on a keyword to view the command details.

**See Also**

- Learn about Group Delay Aperture
- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**Critical Note:** `CALCulate` commands act on the selected measurement. You can select one measurement for each channel using `Calc:Par:MNUM` or `Calc:Par:Select`. Learn more.

`CALCulate<cnum>:GDElay:FREQuency <value>`

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

**(Read-Write)** Sets group delay aperture using a fixed frequency range.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<value>` Frequency range (in Hz) to use for the aperture setting. Choose between the equivalent of two data points and the channel frequency span.

**Examples**

`CALC : GDEL : FREQ 1E6`

**Query Syntax**

`CALCulate<cnum>:GDElay:FREQuency?`
CALCulate<cnum>:GDELay:PERCent <value>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets group delay aperture using a percent of the channel frequency span.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<value>` Percent of frequency span to use for the aperture setting. Choose between the equivalent of two data points and 100 percent of the channel frequency span.

**Examples**

- *set to 25 percent of the channel frequency span*

  ```
  CALC:GDEL:PERC 25
  ```

**Query Syntax**
CALCulate<cnum>:GDELay:PERCent?

**Return Type** Numeric

**Default** Percent of frequency span that equates to 11 points. This can be changed to two points with a preference setting.

CALCulate<cnum>:GDELay:POINts <value>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

**(Read-Write)** Sets group delay aperture using a fixed number of data points.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

- `<value>` Number of data points to use for the aperture setting. Choose between two points and the number of points in the channel.

**Examples**

'\textit{set to 25 data points}'

\begin{verbatim}
CALC:GDEL:POIN 25
\end{verbatim}

**Query Syntax**

\texttt{CALCulate\langle\textit{cnum}\rangle:GDELay:POINts?}

**Return Type**

Numeric

**Default**

11 points. This can be changed to two points with a preference setting.
Calc:Limit Commands

Controls the limit segments used for pass / fail testing.

These commands are **Superseded** by the CALCulate:MEASure:LIMit commands.

```
CALCulate:LIMit:
  DATA
    | DELete
  DISPlay
    | [STATe]
  FAIL?
  REPort
    | ALL?
    | DATA?
    | POINts?
  SEGMENT
    | AMPLitude
    | START
    | STOP
    | COUNT?
    | STIMulus
    | START
    | STOP
    | TYPE
  SOUNd
    | [STATe]
  [STATe]
```

Click on a keyword to view the command details.
CALCulate<cnum>:LIMit:DATA <block>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets data for limit segments.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement for which limit lines are to be set. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

- `<block>` Data for all limit segments. The following is the data format for 1 segment:
  
  **Type, BegStim, EndStim, BegResp, EndResp**

  - **Type** Type of limit segment. Choose from
    - 0 - Off
    - 1 - Max
    - 2 - Min

  - **BegStim** Start of X-axis value (freq, power, time)

  - **EndStim** End of X-axis value

  - **BegResp** Y-axis value that corresponds with Start of X-axis value

  - **EndResp** Y-axis value that corresponds with End of X-axis value

**Examples**

The following writes three max limit segments for a bandpass filter.

```
CALC:LIM:DATA 1,3e5,4e9,-60,0,1,4e9,7.5e9,0,0,1,7.5e9,9e9,0,-30
```

**Query Syntax**

CALCulate<cnum>:LIMit:DATA?

**Return Type**

Depends on FORM:DATA
CALCulate<cnum>:LIMit:DATA:DELete

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Write-only)* Deletes all limit line data for the selected measurement on the specified channel.

*See Critical Note*

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

**Examples**

```
CALC2:LIM:DATA:DEL
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

CALCulate<cnum>:LIMit:DISPlay[:STATe] <ON | OFF>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Turns the display of limit segments ON or OFF (if the data trace is turned ON).

*See Critical Note*

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

- `<ON | OFF>`: 
  - ON (or 1) - turns the display of limit segments ON.
  - OFF (or 0) - turns the display of limit segments OFF.

**Examples**

```
CALC:LIM:DISP:STAT ON
calculate2:limit:display:state off
```

**Query Syntax**

CALCulate<cnum>:LIMit:DISPlay[:STATe]?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON

---

CALCulate<cnum>:LIMit:FAIL?
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Returns the Pass / Fail status of the limit line test. Returns 1 (Fail) if any data point fails for any limit segment.

Limit display (CALC:LIM:DISP) does NOT have to be ON.

See Critical Note

Parameters

\(<\text{cnum}>\) Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \(<\text{cnum}>\) is set to 1.

Examples

CALC:LIM:FAIL?

Return Type

Boolean

- 0 is returned when Pass
- 1 is returned when Fail

Default

Not Applicable

CALCulate<\text{cnum}>:LIMit:REPort:ALL? <\text{block}>

Applicable Models: N522xB, N523xB, N524xB

(Read-only) Returns the test results (stimulus value, limit test result, upper limit value and lower limit value of all measurement points), for the selected trace. This command returns a point by point description of the limit table and pass/fail test result.

See Critical Note

Parameters

\(<\text{cnum}>\) Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \(<\text{cnum}>\) is set to 1.

\(<\text{block}>\) Depends on FORM:DATA

If the number of the measurement points is N,

\(<\text{Block}> = \langle\text{first stimulus}\rangle, \langle\text{test result}\rangle, \langle\text{upper limit}\rangle, \langle\text{lower limit}\rangle, \ldots, \langle\text{Nth stimulus}\rangle, \langle\text{test result}\rangle, \langle\text{upper limit}\rangle, \langle\text{lower limit}\rangle\)

Where \(<\text{test result}>\) = -1: No limit, 0: Fail, 1: Pass

The following example returns three points:

CALC:LIM:REP:ALL?
CALCulate<cnum>:LIMit:REPort[:DATA]?<block>

**Applicable Models:** N522xB, N523xB, N524xB

(Read-only) Returns the stimulus values (frequency, power level or time) at all the measurement points that failed the limit test, for the selected trace. If there are no failures, a large number is returned (+9.91000000000E+037).

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<block>` Depends on FORM:DATA

If the number of the measurement points that failed the limit test is N, `<block>`=<First failed stimulus>, ..., <Nth failed stimulus>.
The following example assumes that there are no failures:

```
CALC:LIM:REP:DATA?
+9.91000000000E+037
```

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Numeric</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>9.91E37 (no failures)</td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:LImit:REPort:POINts?**

**Applicable Models:** N522xB, N523xB, N524xB

(Read-only) Reads the number of the measurement points that failed the limit test, for the active trace of selected channel.

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

**Examples**

```
CALC:LIM:REP:POIN?
```

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Numeric</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>0</td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:LImit:SEGment<snum>:AMPLitude:STARt <num>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the start (beginning) of the Y-axis amplitude (response) value.

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<snum>`: Segment number; if unspecified, value is set to 1.
- `<num>`: Choose any number between: -500 and 500

Display value is limited to the Maximum and Minimum displayed Y-axis values.

**Examples**

```
CALC:LIM:SEG1:AMPL:STAR 10
calculate2:limit:segment2:amplitude:start 10
```
### Query Syntax
```
CALCulate<cnum>:LIMit:SEGMent<snum>AMPLitude:STARt?
```

### Return Type
- **Numeric**
- **Default** 0

### Applicable Models:
- N522xB, N523xB, N524xB, M937xA, P937xA

### Parameters
- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<snum>`: Segment number; if unspecified, value is set to 1.
- `<num>`: Choose any number between: `-500` and `500`

Display value is limited to the Maximum and Minimum displayed Y-axis values.

### Examples
```
CALC:LIM:SEGM1:AMPL:STOP 10
calculate2:limit:segment2:amplitude:stop 10
```

### Query Syntax
```
CALCulate<cnum>:LIMit:SEGMent<snum>AMPLitude:STOP <num>
```

### Return Type
- **Numeric**
- **Default** 0

### Applicable Models:
- N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the stop (end) of the Y-axis amplitude (response) value.

### See Critical Note

```
CALCulate<cnum>:LIMit:SEGMent:COUNt?
```

### Query Syntax
```
CALCulate<cnum>:LIMit:SEGMent:COUNt
```

### Return Type
- **Numeric**
- **Default** 3497

### Display value is limited to the Maximum and Minimum displayed Y-axis values.
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Returns the number of segments used in a limit test. All segments are counted, whether they are on or not.

**Parameters**
- Not Applicable
- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

**Examples**

CALC:LIM:SEGM:COUN?

**Return Type**
- Numeric
- **Default** Not Applicable

**CALCulate<cnum>:LIMit:SEGMent<snum>:STIMulus:STARt <num>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the start (beginning) of the X-axis stimulus value.

**See Critical Note**

**Parameters**
- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<snum>** Segment number; if unspecified, value is set to 1.
- **<num>** Choose any number within the X-axis span of the analyzer.

**Examples**

CALC:LIM:SEGM1:STIM:STAR 10

**Query Syntax**

CALCulate<cnum>:LIMit:SEGMent<snum>:STIMulus:STARt?

**Return Type**
- Numeric
- **Default** 0

**CALCulate<cnum>:LIMit:SEGMent<snum>:STIMulus:STOP <num>**
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the stop (end) of the X-axis stimulus value.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<snum>` Segment number; if unspecified, value is set to 1.
- `<num>` Choose any number within the X-axis span of the analyzer.

**Examples**

```
calc:lim:seg1:ampl:stop 10
```
```
calculate2:limit:segment2:stimulus:stop 10
```

**Query Syntax**

CALCulate<cnum>:LIMit:SEGMent<snum>:STIMulus:STOP?

**Return Type**

Numeric

Default

0

---

**CALCulate<cnum>:LIMit:SEGMent<snum>:TYPE <char>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the type of limit segment.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<snum>` Segment number. Choose any number between 1 and 100. If unspecified, value is set to 1.
- `<char>` Choose from:
  - `LMAX` - a MAX limit segment. Any response data exceeding the MAX value will fail.
  - `LMIN` - a MIN limit segment. Any response data below the MIN value will fail.
  - `OFF` - the limit segment (display and testing) is turned OFF.
Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:LIM:SEG:M:T LMIN</code></td>
<td>Character</td>
<td>OFF</td>
</tr>
<tr>
<td><code>calculate2:limit:segment3:TYPE lmax</code></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:LIMit:SEGMent<snum>:TYPE?**

**Query Syntax**

CALCulate<cnum>:LIMit:SEGMent<snum>:TYPE?

**Return Type**

Character

**Default**

OFF

**CALCulate<cnum>:LIMit:SOUNd[:STATe] <ON | OFF>**

**Applicable Models:**

N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Turns limit testing fail sound ON or OFF.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<ON | OFF>`
  - ON (or 1) - turns sound ON.
  - OFF (or 0) - turns sound OFF.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:LIM:SOUN ON</code></td>
<td>Boolean</td>
<td>1 (ON)</td>
</tr>
<tr>
<td><code>calculate2:limit:sound:state off</code></td>
<td></td>
<td>0 (OFF)</td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:LIMit[:STATe] <ON | OFF>**
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Turns limit segment testing ON or OFF.

- Use `CALC:LIM:DISP` to turn ON and OFF the display of limit segments.
- If using Global Pass/Fail status, trigger the VNA AFTER turning Limit testing ON.

See Critical Note

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
</tbody>
</table>
| <ON | OFF>         | **ON** (or 1) - turns limit testing ON.  
|              | **OFF** (or 0) - turns limit testing OFF.                                     |

**Examples**

- `CALC:LIM:STAT ON`
- `calculate2:limit:state off`

**Query Syntax**

CALCulate<cnum>:LIMit:STATe?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF
**Calculate:Marker Commands**

Controls the marker settings used to remotely output specific data to the computer.

These commands are **Superseded** by the `CALCulate:MEASure:MARKer` commands.

<table>
<thead>
<tr>
<th><strong>CALCulate:MARKer:</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>AOFF</td>
</tr>
<tr>
<td>BUCKET</td>
</tr>
<tr>
<td>BWIDTH</td>
</tr>
<tr>
<td><strong>COMPression</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>COUpling</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>DELTA</td>
</tr>
<tr>
<td>DIScrete</td>
</tr>
<tr>
<td>DISTance</td>
</tr>
<tr>
<td>FORMat</td>
</tr>
<tr>
<td><strong>FUNCtion</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Click on a keyword to view the command details.

See Also

- Marker example program
- Marker Readout number and size commands.
- Learn about Markers
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.
Important: Learn about programming the reference marker.

CALCulate<cnump>:MARKer:AOFF

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

(Write-only) Turns all markers off for selected measurement.

See Critical Note

Parameters

<cnump>  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnump> is set to 1.

Examples  

CALC:MARK:AOFF

calculate2:marker:aoff

Query Syntax  Not applicable

Default  Not applicable

CALCulate<cnump>:MARKern:<num>:BUCKet <nump>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

(Read-Write) Sets and reads the data point (bucket) number of the trace on which the marker resides. When the markers are interpolated (non-discrete), the returned value is the nearest marker bucket position.

See Critical Note

Parameters

<cnump>  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnump> is set to 1.

<n>  Marker number to move or query. The marker must already exist. If unspecified, <n> is set to 1.

<num>  Data point (bucket) number. Choose any data point between: 0 and the number of data points minus 1.

Examples  

CALC:MARK:BUCK 5

calculate2:marker2:bucket 200
CALCulate<cnum>:MARKer<marker>:BUCKet?

Return Type: Integer

Default: The first marker is set to the middle of the span. Subsequent markers are set to the bucket number of the previously active marker.

CALCulate<cnum>:MARKer<marker>:BWIDth <num>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

(Read-Write) Turns on and sets markers 1 through 4 to calculate filter bandwidth. The <num> parameter sets the value below the maximum bandwidth peak that establishes the bandwidth of a filter. For example, if you want to determine the filter bandwidth 3 db below the bandpass peak value, set <num> to -3.

To turn off the Bandwidth markers, either turn them off individually or turn them All Off.

The analyzer screen will show either Bandwidth statistics OR Trace statistics; not both.

To search a User Range with the bandwidth search, first activate marker 1 and set the desired User Range. Then send the CALC:MARK:BWID command. The user range used with bandwidth search only applies to marker 1 searching for the max value. The other markers may fall outside the user range.

See Critical Note

Parameters

- <cnum>: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num>: Target value below filter peak. Choose any number between -500 and 500

Examples

- CALC:MARK:BWID -3
- calculate2:marker:bwidth -2.513

Query Syntax

CALCulate<cnum>:MARKer<marker>:BWIDth?

Returns the results of bandwidth search:

Return Type: Numeric - Four Character values separated by commas: bandwidth, center Frequency, Q, loss.

Default: -3

CALCulate<cnum>:MARKer<mkr>:COMPression:LEVel <num>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

(Read-Write) Set and read the marker compression level. A compression marker must already exist. Use CALC:MARK ON and CALC:MARK:FUNC COMP to create compression markers.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any existing marker number from 1 to 15; if unspecified, value is set to 1.
- `<num>` Compression level. Choose any number between: -500 dB to 500 dB

Standard gain compression values are positive.

Examples

- **CALC:MARK:COMP:LEV 1**
- **calculate2:marker:compression:level 1.5**

Query Syntax

- `CALCulate<cnum>:MARKer:mkr:COMPression:LEVel?`

Return Type

- Numeric
- **Default**: +1

---

CALCulate<cnum>:MARKer<mkr>:COMPression:PIN?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

(Read-only) Reads the input power at the marker compression level. First send CALC:MARK:FUNC:EXEC COMP or CALC:MARK:FUNC:TRAC ON

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any existing marker number from 1 to 15; if unspecified, value is set to 1.

Examples

- **CALC:MARK:COMP:PIN?**
- **calculate2:marker:compression:pin?**

Return Type

- Numeric
- **Return Type**: Numeric
- **Default**: 3506
CALCulate<cnum>:MARKer<mkr>:COMPression:POUT?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

(Read-only) Reads the output power at the marker compression level. First send CALC:MARK:FUNC:EXEC COMP or CALC:MARK:FUNC:TRAC ON

See Critical Note

**Parameters**

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<mkr>** Any existing marker number from 1 to 15; if unspecified, value is set to 1.

**Examples**

```
CALC:MARK:COMP:POUT?
calculate2:marker2:compression:pout?
```

**Return Type** Numeric

**Default** Not applicable

---

CALCulate<cnum>:MARKer<mkr>:COUPling:METHod <char>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

(Read-Write) Sets and reads the scope of Coupled Markers. This is a global setting that affects all markers. Learn more.

**Note:** This command will not take effect until Coupled Markers is turned on using CALC:MARK:COUP:STATe ON.

**Note:** The preset behavior of Coupled Markers depends on the setting of SYStem:PREFerences:ITEM:MCControl, SYStem:PREFerences:ITEM:MCMethod, and SYStem:PREFerences:ITEM:MCPRest.

**Parameters**

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
<mkr> Any existing marker number from 1 to 15; if unspecified, value is set to 1.

<char> CHANnel - Coupling is limited to traces in the same channel.

ALL - Coupling occurs across all channels.

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MARK:COUP:METH CHAN</td>
</tr>
<tr>
<td>calculatel:marker1:coupling all</td>
</tr>
</tbody>
</table>

Query Syntax
CALCulate:MARKer:COUPling:METHod?

Return Type
Character

Default
ALL

CALCulate<cnum>:MARKer<mkr>:COUPling:[STATe]<ON|OFF>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

(Read-Write) Sets and reads the state of Coupled Markers (ON and OFF). The scope of coupled markers can be changed with CALC:MARK:COUP:METH.

See Critical Note

Parameters

<cnun> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<mkr> Any existing marker number from 1 to 15; if unspecified, value is set to 1.

<ON|OFF> OFF (0) - Turns Coupled Markers OFF

ON (1) - Turns Coupled Markers ON

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MARK:COUP ON</td>
</tr>
<tr>
<td>calculatel:marker1:coupling off</td>
</tr>
</tbody>
</table>

Query Syntax
CALCulate:MARKer:COUPling:[STATe]?

Return Type
Boolean (1 = ON, 0 = OFF)

Default
OFF

CALCulate<cnum>:MARKer<mkr>:DELTa <ON|OFF>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

(Read-Write) Specifies whether marker is relative to the Reference marker or absolute.

Note: The reference marker must already be turned ON with CALC:MARK:REF:STATE.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any existing marker number from 1 to 15; if unspecified, value is set to 1.
- `<ON|OFF>` ON (or 1) - Specified marker is a Delta marker
  OFF (or 0) - Specified marker is an ABSOLUTE marker

**Examples**

```
CALC:MARK:DELT ON
calculate2:marker8:delta off
```

**Query Syntax**

CALCulate<cnum>:MARKer<mkr>:DELTa?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

CALCulate<cnum>:MARKer<mkr>:DISCrete <ON|OFF>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

(Read-Write) Makes the specified marker display either a calculated value between data points (interpolated data) or the actual data points (discrete data).

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any existing marker number from 1 to 15; if unspecified, value is set to 1.

3509
<ON|OFF>  
ON (or 1) - Specified marker displays the actual data points  
OFF (or 0) - Specified marker displays calculated data between the actual data points.

Examples

CALC:MARK:DISC ON
calculate2:marker8:discrete off

Query Syntax  
CALCulate<cnum>:MARKer<mkr>:DISCrete?

Return Type
Boolean (1 = ON, 0 = OFF)

Default
OFF

CALCulate<cnum>:MARKer<mkr>:DISTance <num>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Set or query marker distance on a time domain trace.

CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

The Write command moves the marker to the specified distance value. Once moved, you can read the Y axis value or read the X-axis time value. (Distance is calculated from the X-axis time value.)

The Read command reads the distance of the marker.

If the marker is set as delta, the WRITE and READ data is relative to the reference marker.

See Critical Note

Parameters
<cnun>  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnun> is set to 1.
<mkr>  Any existing marker number from 1 to 15; if unspecified, value is set to 1.
<num>  Marker distance in the unit of measure specified with CALC:TRAN:TIME:MARK:UNIT

Examples

CALC:MARK:DIST .1
calculate2:marker8:distance 5

Query Syntax
CALCulate<cnum>:MARKer<mkr>:DISTance?

Return Type
Numeric

Default
Not Applicable

CALCulate<cnum>:MARKer<mkr>:FORMat <char>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

(Read-Write) Sets the format of the data that will be returned in a marker data query CALC:MARK:Y? and the displayed value of the marker readout. The selection does not have to be the same as the measurement's display format.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1
- `<char>` Choose from:
  - **DEFault** - The format of the selected measurement
  - **MLINear** - Linear magnitude
  - **MLOGarithmic** - Logarithmic magnitude
  - **IMPedance** - (R+jX)
  - **ADMittance** - (G+jB)
  - **PHASE** - Phase
  - **IMAGinary** - Imaginary part (Im)
  - **REAL** - Real part (Re)
  - **POLar** - (Re, Im)
  - **GDELay** - Group Delay
  - **LINPhase** - Linear Magnitude and Phase
  - **LOGPhase** - Log Magnitude and Phase
  - **KELVin** - temperature
  - **FAHRENheit** - temperature
  - **CELSius** - temperature
  - **NOISe** - Noise (available ONLY in IM Spectrum and SA measurement classes).
### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MARK:FORMat MLIN</td>
<td>Character</td>
</tr>
<tr>
<td>calculate2:marker8:format Character</td>
<td>Character</td>
</tr>
</tbody>
</table>

### Query Syntax

CALCulate<cnum>:MARKer<mkr>:FORMat?

### Return Type

Character

### Default

DEFault

---

CALCulate<cnum>:MARKer<mkr>:FUNCTION:APEak:EXCursion <num><unit>

#### Applicable Models:
N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

(Read-Write) Sets amplitude peak excursion for the specified marker. The Excursion value determines what is considered a "peak". This command applies to marker peak searches (Next peak, Peak Right, Peak Left).

See Critical Note

#### Parameters

- **<cnum>**
  
  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

- **<mkr>**
  
  Any existing marker number from 1 to 15; if unspecified, value is set to 1.

- **<num>**
  
  Excursion value. Choose any number between **-500** and **500**.

  **Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

- **<unit>**
  
  Varies depending on the data format.

  - Log magnitude (MLOG): dB (decibel)
  - Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): deg (degree)
  - Group delay (GDEL): s (second)
  - Others: No unit

#### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MARK:FUNC:APE:EXC 10</td>
<td>Numeric</td>
</tr>
<tr>
<td>calculate2:marker8:function:apeak:excursion maximum</td>
<td>Numeric</td>
</tr>
</tbody>
</table>

### Default

3

---

3512
CALCulate<cnum>:MARKer<mkr>:FUNCtion:APEak:THReshold <num><unit>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

(Read-Write) Sets peak threshold for the specified marker. If a peak (using the criteria set with :EXCursion) is below this reference value, it will not be considered when searching for peaks. This command applies to marker peak searches (Next peak, Peak Right, Peak Left).

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1
- `<num>` Threshold value. Choose any number between -500 and 500.
- `<unit>` Varies depending on the data format.
  - Log magnitude (MLOG): dB (decibel)
  - Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): deg (degree)
  - Group delay (GDEL): s (second)
  - Others: No unit

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```
CALC:MARK:FUNC:APE:THR -40
calculate2:marker8:function:apeak:threshold -55
```

**Query Syntax**

CALCulate<cnum>:MARKer<mkr>:FUNCtion:APEak:THReshold?

**Return Type** Numeric

**Default** -100

CALCulate<cnum>:MARKer<mkr>:FUNCtion:DOMain:USER[:RANGE] <range>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

*(Read-Write)* Assigns the specified marker to a range number. The x-axis travel of the marker is constrained to the range's span. The span is specified with the CALC:MARK:FUNC:DOM:USER:START and STOP commands, unless range 0 is specified which is the full span of the analyzer.

Each channel has **16** user ranges. (Trace statistics use the same ranges.) More than one marker can use a domain range.

*See Critical Note*

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1
- `<span>` User span. Choose any Integer from **0 to 16**
  
  - **0** is Full Span of the analyzer
  
  - **1 to 16** are available for user-defined x-axis span

**Examples**

```
CALC:MARK:FUNC:DOM:USER 1
calculate2:marker8:function:domain:user:range 1
```

**Query Syntax**

CALCulate:<cnum>:MARKer:<mkr>:FUNCTION:DOMain:USER[:RANGe]?

Returns the user span number that the specified marker is assigned to.

**Return Type**

Numeric

**Default**

0 - Full Span

CALCulate:<cnum>:MARKer:<mkr>:FUNCTION:DOMain:USER:STARt <start>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

(Read-Write) Sets the start of the span that the specified marker's x-axis span will be constrained to.

Use CALC:MARK:FUNC:DOM:USER<range> to set range number


Note: If the marker is assigned to range 0 (full span), the USER:STARt and STOP commands generate an error. You cannot set the STARt and STOP values for "Full Span".

Note: This command does the same as CALC:FUNC:DOM:USER:STAR

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1
- `<start>` The analyzer's Minimum x-axis value

**Examples**

```
calculate2:marker8:function:domain:user:start 1e12
```

**Query Syntax**


**Return Type**

Numeric

**Default**

The analyzer's Minimum x-axis value

CALCulate<cnum>:MARKer<mkr>:FUNCtion:DOMain:USER:STOP <stop>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

*(Read-Write)* Sets the stop of the span that the marker's x-axis travel will be constrained to.

Use `CALC:MARK:FUNC:DOM:USER<range>` to set range number

Use `CALC:MARK:FUNC:DOM:USER:STARt` to set the stop value.

**Note:** If the marker is assigned to range 0 (full span), the USER:STARt and STOP commands generate an error. You cannot set the STARt and STOP values for "Full Span".

**Note:** This command does the same as `CALC:FUNC:DOM:USER:STOP`

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<stop>` Stop value of x-axis span; Choose any number between the analyzer's MINimum and MAXimum x-axis value.

**Examples**

```
calculate2:marker8:function:domain1:user:stop 1e12
```

**Query Syntax**

`CALCulate<cnum>:MARKer<mkr>:FUNCtion:DOMain:USER:STOP?`

**Return Type** Numeric

**Default** The analyzer's MAXimum x-axis value.

```
CALCulate<cnum>:MARKer<mkr>:FUNCtion:EXECute <func>
```
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

(Write-only) Immediately executes (performs) the specified search function.

Learn more about Marker Search

See Critical Note

Parameters

- `<cnum>`  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>`  Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<func>`  The function to be performed. Choose from:
  - MAXimum - finds the highest value
  - MINimum - finds the lowest value
  - RPEak - finds the next valid peak to the right
  - LPEak - finds the next valid peak to the left
  - NPEak - finds the next highest value among the valid peaks
  - TARGET - finds the target value to the right, wraps around to the left
  - LTARGET - finds the next target value to the left of the marker
  - RTARGET - finds the next target value to the right of the marker
  - COMPression - finds the compression level on a Power Swept S21 trace.

Examples

```
CALC:MARK:FUNC:EXEC MAX
```

Query Syntax

- Not applicable

Default

Not applicable

CALCulate<cnum>:MARKer<mkr>:FUNCTION[:SELect] <char>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

(Read-Write) Sets the search function that the specified marker will perform when executed. Use CALC:MARK:FUNC:TRAC ON to automatically execute the search every sweep.

Learn more about Marker Search

See Critical Note

**Parameters**

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<mkr>** Any marker number from 1 to 15; if unspecified, value is set to 1.
- **<char>** Marker function. Choose from:
  - **MAXimum** - finds the highest value
  - **MINimum** - finds the lowest value
  - **RPEak** - finds the next valid peak to the right
  - **LPEak** - finds the next valid peak to the left
  - **NPEak** - finds the next highest value among the valid peaks
  - **TARGET** - finds the target value to the right, wraps around to the left
  - **LTARGET** - finds the next target value to the left of the marker
  - **RTARGET** - finds the next target value to the right of the marker
  - **COMPression** - finds the compression level on a power-swept S21 trace.

**Examples**

```
CALC:MARK:FUNC MAX
CALCulate2:marker8:function:select ltarget
```

**Query Syntax**

CALCulate<cnum>:MARKer<mkr>:FUNCTION[:SE lect]?

**Return Type**

Character

**Default**

MAX

CALCulate<cnum>:MARKer<mkr>:TARGET[:VALUE] <num><unit>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

*(Read-Write)* Sets the target value for the specified marker when doing Target Searches with CALC:MARK:FUNC:SEL <TARGet | RTARGet | LTARGet>

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<num>` Target value to search for; Units are NOT allowed.
- `<unit>` Varies depending on the data format.
  
  - Log magnitude (MLOG): dB (decibel)
  - Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): deg (degree)
  - Group delay (GDEL): s (second)
  - Others: No unit

**Examples**

- `CALC:MARK:TARG 2.5`
- `calculate2:marker8:target:value -10.3`

**Query Syntax**

`CALCulate<cnum>:MARKer<mkr>:TARGet[:VALue]`?

**Return Type**

Numeric

**Default**

0

`CALCulate<cnum>:MARKer<mkr>:FUNCtion:TRACking <ON | OFF>`
CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

(Read-Write) Sets the tracking capability for the specified marker. The tracking function finds the selected search function every sweep. In effect, turning Tracking ON is the same as doing a CALC:MARK:FUNC:EXECute command every sweep.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<ON | OFF>`

  - **ON** (or 1) - The specified marker will "Track" (find) the selected function every sweep.
  - **OFF** (or 0) - The specified marker will find the selected function only when the CALC:MARK:FUNC:EXECute command is sent.

Examples

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MARK:FUNC:TRAC ON</td>
</tr>
<tr>
<td>calculate2:marker8:function:tracking off</td>
</tr>
</tbody>
</table>

Query Syntax

CALCulate<cnum>:MARKer<mkr>:FUNCtion:TRACking?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF

CALCulate<cnum>:MARKer:REFerence[:STATe] <ON | OFF>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

**(Read-Write)** Turns the reference marker ON or OFF. When turned OFF, existing Delta markers revert to general-purpose markers.

**Important:** Learn about programming the reference marker.

See Critical Note

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<ON | OFF>`:
  - **ON** (or 1) - turns reference marker ON
  - **OFF** (or 0) - turns reference marker ON

**Examples**

```
CALC:MARK:REF ON
```

```
calculate2:marker:reference:state OFF
```

**Query Syntax**

CALCulate<cnum>:MARKer:REFerence[:STATe]?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

---

CALCulate<cnum>:MARKer:REference:X <num>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

**(Read-Write)** Sets and returns the absolute x-axis value of the reference marker.

**Important:** Learn about programming the reference marker.

See Critical Note

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>`: X-axis value. Choose any number within the operating domain of the reference marker.

**Examples**

```
CALC:MARK:REF:X 1e9
```

```
calculate2:marker:reference:x 1e6
```
CALCulate<cnum>:MARKer:REFerence:X?

Return Type: Numeric

Default: If the first Marker, turns ON in the middle of the X-axis span. If not, turns ON at the position of the active marker.

CALCulate<cnum>:MARKer:REFerence:Y?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are Superseded by the CALCulate:MEASure:MARKer commands.

(Read-only) Returns the absolute Y-axis value of the reference marker.

Important: Learn about programming the reference marker.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

CALC:MARK:REF:Y?
calculate2:marker:reference:y?

Return Type: Character

Default: Not applicable

CALCulate<cnum>:MARKer<mkr>:SET <char>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

*(Write-only)* Sets the selected instrument setting to assume the value of the specified marker.

Marker Functions CENT, SPAN, STARt, and STOP do not work with channels that are in **CW** or **Segment Sweep** mode.

See Critical Note

### Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1
- `<char>` Choose from:
  - **CENTer** - changes center frequency to the value of the marker
  - **SA** - creates an SA channel with a marker at the same CW frequency. [Learn more.](#)
  - **SPAN** - changes the sweep span to the span that is defined by the delta marker and the marker that it references. Unavailable if there is no delta marker.
  - **STARt** - changes the start frequency to the value of the marker
  - **STOP** - changes the stop frequency to the value of the marker
  - **RLEVel** - changes the reference level to the value of the marker
  - **DElay** - changes the line length at the receiver input to the phase slope at the active marker stimulus position.
  - **CWFReq** - Sets the CW frequency to the frequency of the active marker. Does NOT change sweep type. NOT available in CW or Power Sweep. Use this argument to first set the CW Frequency to a value that is known to be within the current calibrated range, THEN set Sweep:Type to POWer or CW.

### Examples

```
CALC:MARK:SET CENT
calculate2:marker8:set span
```

### Query Syntax

- Not Applicable
- **Default** Not Applicable

CALCulate<cnum>:MARKer<mkr>[:STATe] <ON|OFF>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

**(Read-Write)** Turns the specified marker ON or OFF. To turn all markers off, use CALC:MARK:AOFF.

See Critical Note

**Parameters**

<**cnum**> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<**mkr**> Any marker number from 1 to 15; if unspecified, value is set to 1.

<**ON|OFF**> ON (or 1) - turns marker ON.

OFF (or 0) - turns marker OFF.

**Examples**

```
CALC:MARK ON
```

```
calculate2:marker8 on
```

**Query Syntax**

CALCulate<cnum>:MARKer<mkr>:STATE?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default** Off

---

CALCulate<cnum>:MARKer<mkr>:TYPE <**char**>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

**(Read-Write)** Sets the type of the specified marker.

See Critical Note

**Parameters**

<**cnum**> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<**mkr**> Any marker number from 1 to 15; if unspecified, value is set to 1

<**char**> Choose from:

- **NORMAL** - a marker that stays on the assigned X-axis position unless moved or searching.
- **FIXed** - a marker that will not leave the assigned X or current Y-axis position.

### Examples

```
CALC:MARK:TYPE NORM
calculate2:marker2:type fixed
```

### Query Syntax

```
CALCulate<cnum>:MARKer<mkr>:TYPE?
```

### Return Type

Character

**Default**

NORMal

---

**CALCulate<cnum>:MARKer<mkr>:X <num>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the `CALCulate:MEASure:MARKer` commands.

(Read-Write) Sets the marker's X-axis value (frequency, power, or time). If the marker is set as delta, the SET and QUERY data is relative to the reference marker.

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>`: Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<num>`: Any X-axis position within the measurement span of the marker.

**Note:** This command will accept `MIN` or `MAX` instead of a numeric parameter. See [SCPI Syntax](#) for more information.

```
CALC:MARK:X 100Mhz
calculate2:marker8:x maximum
```

**Query Syntax**

```
CALCulate<cnum>:MARKer<mkr>:X?
```

**Return Type**

Numeric

**Default**

First Marker turns ON in the middle of the X-axis span. Subsequent markers turn ON at the position of the active marker.

---

**CALCulate<cnum>:MARKer<mkr>:Y?**
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

CALCulate:MARKer commands are **Superseded** by the CALCulate:MEASure:MARKer commands.

*(Read-only)* Reads the marker's Y-axis value. The format of the value depends on the current CALC:MARKER:FORMAT setting. If the marker is set as delta, the data is relative to the reference marker. The query always returns two numbers:

- Smith and Polar formats - (Real, Imaginary)
- LINPhase and LOGPhase - (Real, Imaginary)
- All other formats - (Value,0)

**Note:** To accurately read the marker Y-axis value with trace smoothing applied, the requested format must match the displayed format. Otherwise, the returned value is un-smoothed data. For example, to read the smoothed marker value when measuring group delay, both the display format and the marker format must be set to (Group) Delay.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.

**Examples**

- `CALC:MARK:Y?`
- `calculate2:marker3:y?`

**Query Syntax**

CALCulate<cnum>:MARKer<mkr>:Y?

**Return Type**

- Numeric

**Default**

- Not applicable
Calculate: Math Commands

Controls math operations on the currently selected measurement and memory.

These commands are **Superseded** by the CALCulate:MEASure:MATH commands.

<table>
<thead>
<tr>
<th>CALCulate: MATH:</th>
</tr>
</thead>
<tbody>
<tr>
<td>FUNCTION</td>
</tr>
<tr>
<td>INTERpolate</td>
</tr>
<tr>
<td>MEMorize</td>
</tr>
</tbody>
</table>

Click on a keyword to view the command details.

See Also

- Example Programs
- Learn about Math Operations
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

CALCulate<cnum>:MATH:FUNCTION <char>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets math operations on the currently selected measurement and the trace stored in memory. (There MUST be a trace stored in Memory. See CALC:MATH MEM)

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>` The math operation to be applied. Choose from the following:
CALCulate<cnum>:MATH:FUNCtion?

Query Syntax: CALCulate<cnum>:MATH:FUNCtion?

Return Type: Character

Default: NORMal

CALCulate<cnum>:MATH:INTerpolate <bool>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets and reads the state of the memory data interpolation. Learn more.

See Critical Note

Parameters

<ch> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <ch> is set to 1.

<bool> Choose from:

0 - OFF - Turn memory data interpolation OFF.

1 - ON - Turn memory data interpolation ON.

Examples: CALC2:MATH:INT 1

Query Syntax: CALCulate<ch>:MATH:INTerpolate?

Return Type: Boolean

Default: 0

CALCulate<cnum>:MATH:MEMorize
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Write-only)* Puts the currently selected measurement trace into memory. *(Data-> Memory)*.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

**Examples**

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:MATH:MEM</code></td>
</tr>
<tr>
<td><code>calculate2:math:memorize</code></td>
</tr>
</tbody>
</table>

**Query Syntax**

Not applicable

**Default**

Not applicable
These commands are for setting up measurements.

<table>
<thead>
<tr>
<th>CALCulate:MEASure</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLIMit More commands</td>
</tr>
<tr>
<td>COMPutation</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>CONVersion</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>CORRection More commands</td>
</tr>
<tr>
<td>DATA More commands</td>
</tr>
<tr>
<td>DEFine</td>
</tr>
<tr>
<td>DELete</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>DISTortion More commands</td>
</tr>
<tr>
<td>EQUation</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>FILTertion More commands</td>
</tr>
<tr>
<td>FORMat</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>FUNCtion More commands</td>
</tr>
<tr>
<td>GCDData More</td>
</tr>
</tbody>
</table>
commands
GCMeas More commands
GDELay More commands

HOLD
| CLEar
| [TYPE]

LiMit More commands
MARKer More commands

MATH
| FUNCTION
| INTerpolate[:STATe]
| MEMorize

MIXer
| XAXis

OFFSet More commands
PARameter More commands
PN More commands

RDATa?
RLIMit More commands
SA More commands
SMOothing More commands
TRANsform More commands
CALCulate<cnum>:MEASure<mnum>:CONVersion:FUNCtion <char>

**Applicable Models:** All

(Read-Write) Sets or gets the parameter after conversion using the parameter conversion function, for the active trace of selected channel.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<char>` Select from the following parameters after conversion:
  - "OFF"
  - "ZREFlection" - Specifies the equivalent impedance in reflection measurement.
  - "ZTRANSMIT" - Specifies the equivalent impedance (series) in transmission measurement.
  - "ZTSHUNT" - Specifies the equivalent impedance (shunt) in transmission measurement.
  - "YREFlection" - Specifies the equivalent admittance in reflection measurement.
  - "YTRANSMIT" - Specifies the equivalent admittance (series) in transmission measurement.
  - "YTSHUNT" - Specifies the equivalent admittance (shunt) in transmission measurement.
  - "INVERSION" - Specifies the inverse S-parameter (1/S).
"CONJugation" - Specifies the conjugate.

Examples

SCPI sequence to retrieve resistance from a polar chart:

```
CALC:MEAS:CONV:FUNC ZREFlection
CALC:MEAS:FORM:REAL
CALC:MEAS:DATA:FDATA?
```

SCPI sequence to retrieve reactance from a polar chart:

```
CALC:MEAS:CONV:FUNC ZREFlection
CALC:MEAS:FORM:IMAG
CALC:MEAS:DATA:FDATA?
```

Query Syntax

```
CALCulate<cnum>:MEASure<mnum>:CONVersion:FUNCtion?
```

Return Type

Character

Default

"OFF"

CALCulate<cnum>:MEASure<mnum>:COMPutation:DEViation <char>

Applicable Models: All

(Read-Write) Calculates the deviation from a least-squares best fit line.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<char>` Select from the following parameters:

  - "OFF"
  - "LINear" - Fit to 1st order curve minimizing RSS deviation.
  - "PARabolic" - Fit to 2nd order curve minimizing RSS deviation.
  - "CUBic" - Fit to 3rd order curve minimizing RSS deviation.

Examples

```
CALC:MEAS1:COMP:DEV LIN
```

Query Syntax

```
CALCulate<cnum>:MEASure<mnum>:COMPutation:DEViation?
```

Return Type

Character

Default

"OFF"
CALCulate<cnum>:MEASure<mnum>:DEFine <string>

Applicable Models: All

(Write-only) Creates a measurement but does NOT display it, on an existing or new channel. When a new channel is created, any licensed measurement class can be used. Up to 580 (2000 for M980xA/P50xxA and E5080B) measurements can be created.

Note: For setting up a TDR measurement class, use SYSTem:TDR:INITialize.

Note that each display window can only display a limited number of traces. See Traces, Channels, and Windows on the VNA.

- Use DISP:WIND:STATe to create a window if it doesn't already exist.
- Use DISP:MEAS<mnum>:FEED<wnum> to display the measurement in window <wnum>.

This command replaces the following commands:

CALCulate:PARameter[DEFine]
CALCulate:PARameter[DEFine]:EXTended
CALCulate:CUSTom:DEFine

Parameters

<cnum> Channel number of the new measurement. If unspecified, value is set to 1.

If the specified channel does not exist, then a channel of the specified type will be created. If no type of channel is specified, then a standard channel will be created.

If the specified channel exists, then the parameter will be added to the channel provided the existing channel supports the parameter (otherwise, an error will be generated).

<mnum> Measurement number for the new measurement.

If the specified measurement number is already in use, an error will be generated.

<string> (String) Measurement Parameter and optional measurement class name separated by a ":" (colon). For example, "S21:Gain Compression" creates an S21 measurement and selects the Gain Compression measurement class for the channel.

Note: If a measurement class of a channel does not support the defined
measurement parameter, an error is generated.

Case sensitive.

**For S-parameters:**

Any S-parameter available in the VNA

Single-digit port numbers CAN be separated by "_" (underscore). For example: "S21" or "S2_1"

Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

**For ratioed measurements:**

Any two VNA physical receivers separated by forward slash '/' followed by comma and source port.

For example: "A/R1, 3"

[Learn more about ratioed measurements](#)

See a [block diagram](#) showing the receivers in YOUR VNA.

**For non-ratioed measurements:**

Any VNA physical receiver followed by comma and source port.

For example: "A, 4"

[Learn more about unratioed measurements](#)

See the [block diagram](#) showing the receivers in YOUR VNA.

**Ratioed and Unratioed** measurements can also use [logical receiver notation](#) to refer to receivers. This notation makes it easy to refer to receivers with an [external test set](#) connected to the VNA. You do not need to know which physical receiver is used for each test port. [Learn more](#).

**For ADC measurements:**

Any ADC receiver in the VNA followed by a comma, then the source port.

For example: "AI1,2" indicates the Analog Input1 with source port of 2.

[Learn more about ADC receiver measurements](#)
The following are the existing valid measurement parameters for each measurement class (click a link or scroll down to view). The Measurement Class must be installed and registered on the VNA.

- "Standard"
- "Active Hot Parameters"
- "Vector Mixer/Converter"
- "Scalar Mixer/Converter"
- "Gain Compression"
- "Gain Compression Converters"
- "Modulation Distortion"
- "Noise Figure Cold Source"
- "Noise Figure Converters"
- "Swept IMD"
- "IM Spectrum"
- "Swept IMD Converters"
- Modulation Distortion Converters
- Phase Noise
- "IM Spectrum Converters"
- "Differential I/Q"
- "Spectrum Analyzer"

**variant** Measurement names to create:

<table>
<thead>
<tr>
<th>Meas Class</th>
<th>Measurement Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Standard&quot;</td>
<td>&quot;S11&quot;, &quot;S21&quot;, and so forth</td>
<td>S-parameter name</td>
</tr>
<tr>
<td></td>
<td>&quot;A_1&quot;, &quot;A_2&quot;, and so forth</td>
<td>Unratioed parameter names with notation: &quot;receiver_source port&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See balanced parameter names</td>
</tr>
<tr>
<td>Active Hot Parameters</td>
<td>Port 1 is the Source Port (DUT input). Port 3 or Port 2 can be chosen as the output of the DUT.</td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;HotS11&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;HotS31&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;HotS13&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;HotS33&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;IPwr&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;OPwr&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Gamma&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Pmax&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Xs(3,3)&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Xt(3,3)&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Xf(3,1)&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;DeltaOPwr&quot;</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>For output port Y (input port must be 1):</th>
<th>Learn about VMC parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;S11&quot;</td>
<td>Note: Input and output ports are set up using the Mixer Setup dialog. If the ports are not set up using the Mixer Setup dialog, then ports 1 and 2 are the default input and output ports and the only ports that can be used.</td>
</tr>
<tr>
<td>&quot;VCY1&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;SYY&quot;</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>For input port X and output port Y:</th>
<th>Learn about SMC parameters</th>
</tr>
</thead>
</table>
### Scalar Mixer/Converter

- "SCXY"
- "SCYX"
- "SXX"
- "SYY"
- "Ipwr"
- "RevIPwr"
- "Opwr"
- "RevOPwr"

**Note:** Input and output ports are set up using the Mixer Setup dialog. If the ports are not set up using the Mixer Setup dialog, then ports 1 and 2 are the default input and output ports and the only ports that can be used.

### Gain Compression

**Learn more**

**GCA and GCX:**

- "CompIn21" Input power at the compression point.
- "CompOut21" Output power at the compression point.
- "CompGain21" Gain at the compression point.
- "CompS11" Input Match at the compression point
- "RefS21" Linear Gain
- "DeltaGain21" CompGain21 -Linear Gain
- "S11", "S21", "S12", "S22" Standard S-parameters; measured at port 1 and port 2

**GCX - All Gain Compression parameters (except S21 and S12) plus the following:**

- "S11"
- "SC21"
- "SC12"
- "S22"
- "Ipwr"
- "RevIPwr"
- "Opwr"
- "RevOPwr"

**Learn more**

---

"Gain Compression Converters"

---

3538
Modulation Distortion and Modulation Distortion Converters:

- "Pin1" Power In
- "POut1" Reflected Power In
- "POut2" Power Out
- "PModFile" Power of modulation file
- "MSig2" Modulation Signal Out
- "MDist2" Modulation Distortion Out
- "MDistIR2" Modulation Distortion Input-referred
- "MGain21" Modulation Gain
- "MComp21" Modulation Compression
- "PGain21" Power Gain
- "LMatch2" Load match of VNA port
- "CarrIn1" Input Band Power
- "CarrOut2" Output Band Power
- "CarrGain21" Band Power Gain
- "NPRIn1" NPR at Input
- "NPROut2" NPR at Output
- "NPRDist21" NPR Distortion, Added by DUT
- "NPRPwrIn1" NPR Input Power
- "NPRPwrOut2" NPR Output Power
- "ACPIn1" ACP at input
- "ACPOut2" ACP at output
- "ACPDist21" ACP distortion, Added by DUT
- "ACPPwrIn1" ACP Input Power
- "ACPPwrOut2" ACP Output Power
- "EVMDistEq21" EVM Equalized Distortion, Added by DUT
- "EVMDistUn21" EVM Unequalized Distortion, Added by DUT
- "EVMPwrIn1" EVM Input Power
- "EVMPwrOut2" EVM Output Power
- "ModFilter" Measurement Modulation Filter
- "A", "b1" Port 1 test port receiver
<table>
<thead>
<tr>
<th>Port 2 test port receiver</th>
<th>Port 3 test port receiver</th>
<th>Port 4 test port receiver</th>
<th>Port 1 reference receiver</th>
<th>Port 2 reference receiver</th>
<th>Port 3 reference receiver</th>
<th>Port 4 reference receiver</th>
</tr>
</thead>
</table>

**Modulation Distortion ONLY - NOT Modulation Distortion Converters**

| "S11" | Linear Input Match |
| "S21" | Linear Gain |
| "LPIn1" | Linear Input Match |
| "LPOut1" | Linear Reflected Power In |
| "LPOut2" | Linear Power Out |

**Noise Figure AND NFX:**

| "NF" | Noise figure |
| "ENR" | Validate noise source measurements. |
| "T-Eff" | Effective noise temperature. |
| "DUTRNP" | DUT noise power ratio. (Noise power expressed in Kelvin divided by 290). |
| "DUTRNPI" | |
| "SYSRNP" | System noise power ratio |
| "SYSRNPI" | |
| "DUTNPD" | DUT noise power density. (Noise power expressed in dBm/Hz). |
| "DUTNPDI" | |
| "SYSNPD" | System noise power density. |
| "SYSNPDI" | |
| "OvrRng" (Opt 029 Only) | Indication that the noise receiver is being over powered. |
| "T-Rcvr" (Opt 029 Only) | Temperature reading (in Kelvin) of the noise receiver board. |
### Noise Figure ONLY - NOT NFX:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;S11&quot;, &quot;S21&quot;,</td>
<td>Standard S-parameters; measured with the port1 and port2 noise switches set for noise mode.</td>
</tr>
<tr>
<td>&quot;S12&quot;, &quot;S22&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;A_1&quot;, &quot;A_2&quot;...</td>
<td>Unratioed parameters; with notation: &quot;receiver, source port&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;GammaOpt&quot;</td>
</tr>
<tr>
<td></td>
<td>Optimum Complex Reflection Coefficient</td>
</tr>
<tr>
<td></td>
<td>&quot;Rn&quot;</td>
</tr>
<tr>
<td></td>
<td>Noise Resistance</td>
</tr>
<tr>
<td></td>
<td>&quot;NFMin&quot;</td>
</tr>
<tr>
<td></td>
<td>Minimum noise figure that occurs at GammaOpt</td>
</tr>
</tbody>
</table>

### NFX ONLY:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;S11&quot;</td>
<td>Mixer parameters</td>
</tr>
<tr>
<td>&quot;SC21&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;SC12&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;S22&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;Ipwr&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;RevIPwr&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;Opwr&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;RevOPwr&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;ALO1&quot;,&quot;BLO1&quot;...</td>
<td>Test port receiver at LO1 frequency</td>
</tr>
<tr>
<td></td>
<td>Unratioed parameters with notation: &quot;receiver_source port&quot;</td>
</tr>
<tr>
<td>&quot;R1_1&quot;, &quot;B_2&quot;...</td>
<td></td>
</tr>
</tbody>
</table>

### Phase Noise

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PN</td>
<td>Phase Noise</td>
</tr>
<tr>
<td>AM</td>
<td>AM Noise</td>
</tr>
</tbody>
</table>

There are over 150 possible Swept IMD parameters, too many to list here.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Swept IMD&quot;</td>
<td>Build the parameters with the Swept IMD Parameter dialog, then copy the parameter name to the remote command. The following are a few example parameters:</td>
</tr>
<tr>
<td>&quot;PwrMainLo&quot;</td>
<td>Absolute power of the Low tone at the DUT output.</td>
</tr>
<tr>
<td>&quot;IM3&quot;</td>
<td>Power of the third product relative to the average power of the f1 and f2 tones measured at the DUT output.</td>
</tr>
<tr>
<td>&quot;OIP3&quot;</td>
<td>Theoretical power level at which the third product will be the same power level as the average of the main tones at the output of the DUT.</td>
</tr>
<tr>
<td>&quot;IM Spectrum&quot;</td>
<td>View signals OUT of the DUT and into VNA port 2 (B receiver).</td>
</tr>
<tr>
<td>&quot;Input&quot;</td>
<td>View signals IN to the DUT (R1 receiver).</td>
</tr>
<tr>
<td>&quot;Reflection&quot;</td>
<td>View signals reflected off the DUT input and back into VNA port 1 (A receiver).</td>
</tr>
<tr>
<td>&quot;IMx Spectrum Converters&quot;</td>
<td>View signals OUT of the DUT and into VNA port 2 (B receiver).</td>
</tr>
<tr>
<td>&quot;Output&quot;</td>
<td>Create custom parameters using Sens:DIQ:Par:Def, then specify your custom parameter name here. The following are default parameters:</td>
</tr>
<tr>
<td>&quot;IPwrF1&quot;</td>
<td>Input Power over F1 range</td>
</tr>
<tr>
<td><strong>&quot;OPwrF1&quot;</strong></td>
<td><strong>&quot;GainF1&quot;</strong></td>
</tr>
<tr>
<td><strong>&quot;GainF1&quot;</strong></td>
<td>Gain over F1 range</td>
</tr>
</tbody>
</table>

| **"Spectrum Analyzer"** | **"a<n>"** | Reference receiver |
| **"ImageReject<n>"** | **"b<n>"** | Test port receiver |
| where <n> is the port number to measure |

| **"Vector Signal Analyzer"** | **"a<n>"** | Reference receiver |
| **"ImageReject<n>"** | **"b<n>"** | Test port receiver |
| where <n> is the port number to measure |

| Impedance Analysis | **"Z"** | Impedance |
| **"Y"** | Admittance |
| **"Cs"** | Equivalent series capacitance |
| **"Cp"** | Equivalent parallel capacitance |
| **"Ls"** | Equivalent series inductance |
| **"Lp"** | Equivalent parallel inductance |
| **"Q"** | Q value |
| **"D"** | Dissipation factor |

**Examples**

```
CALC1:MEAS2:DEF "S11" 'Defines an S11 measurement for channel 1, measurement number 2.
```

```
CALC4:MEAS3:DEF "S21:Gain Compression" 'Defines an S21 measurement for channel 4, measurement number 3, and creates a
```
CALC2:MEAS:DEF "R1,1:Standard" 'Defines an R1,1 measurement for channel 2, measurement number 1 (default), and creates a Standard channel.

SYST:FPrest
CALC1:MEAS1:DEF "NF:Noise Figure Cold Source" 'Defines a Noise Figure measurement for channel 2, measurement number 2, and creates a Noise Figure channel.
DISP:WIND1 ON
DISP:MEAS1:FEED 1
CALC2:MEAS2:DEF "CompIn21:Gain Compression" 'Defines an input power at Compression point but doesn't display on channel 2.
DISP:WIND2 ON
DISP:MEAS2:FEED 2 'CompIn21 measurement in window number 2.
CALC2:MEAS3:DEF "CompOut21:Gain Compression" 'Defines an output power at Compression point but doesn't display on channel 2
DISP:MEAS3:FEED 2 'CompOut21 measurement in window number 2

### Query Syntax

| Default | Not Applicable |

CALCulate<cnum>:MEASure<mnum>:DELete

**Applicable Models:** All

*(Write-only)* Deletes the specified measurement.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number.

**Examples**

- `CALC:MEAS2:DEL`
- `calculate2:measure2:delete`

**Query Syntax**

| Default | Not Applicable |

CALCulate:MEASure:DELete:ALL
Applicable Models: All

(Write-only) Deletes all measurements on the VNA.

### Parameters

### Examples

```
CALC:MEAS:DEL:ALL
```

### Query Syntax

Not Applicable

### Default

Not Applicable

---

CALCulate<cnum>:MEASure<mnum>:EQUation:FAST <bool>

Applicable Models: All

(Read-Write) Set and return equation editor trace update delay. This command delays updating the equation editor trace until all trace references have finished updating to ensure that all data is present.

**Note:** This command does not work in application channels. In addition, this command does not work with the standard channel when the channel is in HOLD and then SINGLE sweeps are sent.

### Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` Choose from:
  - **OFF (0)** Do not delay equation editor trace update.
  - **ON (1)** Delay equation editor trace update.

### Examples

```
CALC:MEAS:EQU:FAST 1
calculate2:measure1:equation:fast OFF
```

### Query Syntax

CALCulate<cnum>:MEASure<mnum>:EQUation:FAST?

### Return Type

Boolean

### Default

**OFF** or **0**

---

CALCulate<cnum>:MEASure<mnum>:EQUation[:STATe] <bool>
Applicable Models: All

(Read-Write) Turns ON and OFF the equation on selected measurement for the specified channel. If the equation is not valid, then processing is not performed. Use CALC:EQUation:VALid? to ensure that the equation is valid.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` ON (or 1) - turns equation ON.
  OFF (or 0) - turns equation OFF.

Examples

```
CALC:MEAS:EQU 1
```
```
calculate2:measure1:equation:state 0
```

Query Syntax

CALCulate<cnum>:MEASure<mnum>:EQUation[:STATe]?

Return Type

Boolean

Default

OFF (0)

---

CALCulate<cnum>:MEASure<mnum>:EQUation:TEXT <string>

Applicable Models: All

(Read-Write) Specifies an equation or expression to be used on the selected measurement for the specified channel.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<string>` Any valid equation or expression. See Equation Editor.

Examples

```
'Equation (includes '=')

CALC:MEAS:EQU:TEXT "foo=S11/S21"

'Expression

calculate2:measure1:equation:text "S11/S21"
```

Query Syntax

CALCulate<cnum>:MEASure<mnum>:EQUation:TEXT?

Return Type

String
CALCulate<cnum>:MEASure<mnum>:EQUation:VALid?

**Applicable Models:** All

**(Read-Only)** Returns a boolean value to indicate if the current equation on the selected measurement for the specified channel is valid. For equation processing to occur, the equation must be valid and ON (CALC:EQU:STAT 1).

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**
```
CALC:MEAS:EQU:VAL?
calculate2:measure1:equation:valid?
```

**Return Type** Boolean
- 1 - equation is valid
- 0 - equation is NOT valid

**Default** Not Applicable

---

CALCulate<cnum>:MEASure<mnum>:FORMat <char>

**Applicable Models:** All

**(Read-Write)** Sets the display format for the measurement.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<char>` Choose from:
  - MLINear
  - MLOGarithmic
  - PHASE
- UPHase 'Unwrapped phase
- IMAGinary
- REAL
- POLar
- SMITh
- SADMittance 'Smith Admittance
- SWR
- GDELay 'Group Delay
- KELVin
- FAHRenheit
- CELSius
- PPHase 'Positive Phase
- COMPlex

Examples

```
CALC:MEAS:FORM MLIN
calculate2:measure1:format polar
```

Query Syntax

```
CALCulate<cnum>:MEASure<mnum>:FORMat?
```

Return Type

Character

Default

MLOG

```
CALCulate<cnum>:MEASure<mnum>:FORMat:UNIT <dataFormat>, <units>
```
Applicable Models: All

(Read-Write) Sets and returns the units for the specified data format. Measurements with display formats other than those specified are not affected.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<dataFormat>` Choose from:
  - **MLOG** - Log magnitude
  - **MLIN** - Linear magnitude
- `<units>` For unratioed MLOG measurements, choose from:
  - **dB** Units are displayed in decibels.
  - **DBM** Units are displayed in dBm. 0 dBm = 0.001 watt
  - **DBMV** Units are displayed in dBmV. 0 dBmV = 0.001 volt
    DBmV value depends on the reference impedance: dBmV = dBm + 30 + 10*log10(Z0)
  - **DBMA** Units are displayed in dBmA. 0 dBmA = 0.001 Ampere
  - **DBUV** Units are displayed in dBuV. 0 dBuV = 1 uV
    DBuV value depends on the reference impedance: dBuV = dBm + 90 + 10*log10(Z0)

For unratioed MLIN measurements, choose from:

- **UNIT** - No units. Linear magnitude without conversion
- **W** - Units are displayed in Watts
- **V** - Units are displayed in Volts
- **A** - Units are displayed in Amperes

**Examples**

```
CALC:MEAS:FORM:UNIT MLOG, DBM
CALCulate2:measure1:format:unit mlog,dbmv
```

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:FORMat:UNIT? <dataFormat>

**Return Type**

Character

**Default**

MLOG, DBM

CALCulate<ch>:MEASure<mnum>:HOLD:CLEAR
Applicable Models: All

(Write-only) Resets the currently-stored data points to the live data trace and restarts the currently-selected Trace Hold type.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

Examples

```
CALC:MEAS:HOLD:CLE
calculate2:measure1:hold:clear
```

Query Syntax

Not Applicable

Default

Not Applicable

CALCulate<ch>:MEASure<mnum>:HOLD:TYPE <value>

Applicable Models: All

(Read-Write) Sets the type of trace hold to perform.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<value>` Trace Hold type. Choose from:

  - `OFF` - Disables the Trace Hold feature.
  - `MINimum` - Sets Trace Hold to store the lowest measured data points.
  - `MAXimum` - Sets Trace Hold to store the highest measured data points.

Examples

```
CALC:MEAS:HOLD:TYPE MAX
calculate2:measure1:hold:type minimum
```

Query Syntax

CALCulate<ch>:MEASure<mnum>:HOLD:TYPE?

Return Type

Character

Default

OFF

CALCulate<cnum>:MEASure<mnum>:MATH:FUNCtion <char>
Applicable Models: All

(Read-Write) Sets math operations on the currently selected measurement and the trace stored in memory. (There MUST be a trace stored in Memory. See `CALC:MEAS:MATH MEM`)

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<char>` The math operation to be applied. Choose from the following:
  - NORMal: Trace data only
  - ADD: Data + Memory
  - SUBTract: Data - Memory
  - MULTiply: Data * Memory
  - DIVide: Data / Memory

Examples

- `CALC:MEAS:FUNC NORM`
- `calculate2:measure1:math:function subtract`

Query Syntax

`CALC<ch>:MEAS<mnum>:MATH:FUNCtion?`

Return Type

Character

Default

NORMal

CALCulate<ch>:MEASure<mnum>:MATH:INTerpolate[:STATE]?

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets and reads the state of the memory data interpolation. Learn more.

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<ch>` is set to 1.
- `<mnum>` Measurement number for each measurement.
- `<bool>` Choose from:
  - 0 - OFF - Turn memory data interpolation OFF.
  - 1 - ON - Turn memory data interpolation ON.

Examples

- `CALC2:MEAS:INT 1`

Query Syntax

`CALC<ch>:MEAS<mnum>:MATH:INTerpolate?`

Return Type

Boolean

Default

0
CALCulate<cnum>:MEASure<mnum>:MATH:MEMorize

Applicable Models: All

(Write-only) Puts the currently selected measurement trace into memory. (Data-> Memory).

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

Examples

- CALC:MEAS:MATH:MEM
- calculate2:measure1:math:memorize

Query Syntax

- Not applicable
- Default Not applicable

CALCulate<ch>:MEASure<mnum>:MIXer:XAXis <char>

Applicable Models: All

(Read-Write) Sets or returns the swept parameter to display on the X-axis for the selected FCA and GCX measurement.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<mnum>` Measurement number for each measurement.
- `<char>` Parameter to display on the X-axis. Choose from:
  - INPUT - Input frequency span
  - OUTPUT - Output frequency span
  - LO_1 - First LO frequency span
  - LO_2 - Second LO frequency span

Examples

- CALC:MEAS:MIX:XAX INPUT
- calculate2:measure1:mixer:xaxis output

See an example that creates, selects, and calibrates an SMC and VMC measurement using SCPI.

Query Syntax

CALCulate<ch>:MEASure<mnum>:MIXer:XAXis?
CALCulate<cnum>:MEASure<mnum>:RDATA? <char>

Applicable Models: All

(Read-only) Returns receiver data for the selected measurement.

Parameters

- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <char> Choose from any physical receiver in the VNA.

For example: "A"

Also, REF - returns data for either R1 or R2 data depending on the source port of the selected measurement.

See the block diagram showing the receivers in YOUR VNA.

Note: Logical receiver notation is NOT allowed with this command. Learn more.

Example

GPIB.Write "INITiate:CONTinuous OFF"
GPIB.Write "INITiate:IMMediate:*wai"
GPIB.Write "CALCulate:MEASure2:RDATA? A"

GPIB.Write "CALCulate:MEASure2:RDATA? REF"

Return Type Depends on FORM:DATA - Two numbers per data point

Default Not Applicable
CALCulate:MEASure:DIStortion Commands

Calculate and display amplitude modulation or phase modulation per trace. This feature is supported on all standard channels and on all memory traces in a standard channel.

```
| CALCulate:MEASure:DISTortion |
|                             |
| BACKoff:                  |
|   | COMPression            |
|   | [:STATe]               |
| MODE                      |
| SLOPe:                    |
|   | APERTure               |
|   | [:STATe]               |
```

Click a keyword to view the command details.

See Also

- Calibrating with SCPI
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

```
| CALCulate<cnum>:MEASure<mnum>:DISTortion:BACKoff:COMPression <num> |
```

3554
(Read-Write) Sets the compression level used in determining the back off calculation.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Compression level.

**Examples**
- `CALC1:MEAS2:DIST:BACK:COMP 1 dB`
- `calculate2:measure2:distortion:backoff:compression 1 dB`

**Query Syntax**
`CALCulate<cnum>:MEASure<mnum>:DISTortion:BACKoff:COMPression?`

**Return Type** Numeric

**Default** 1 dB

(Read-Write) Enables/disables compression level back off function specified using the `CALCulate:MEASure:DISTortion:BACKoff:COMPression` command. When enabled, the x-axis changes to display the back off from the compression point.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` Compression level back off state. Choose from:
  - 0 - Compression level back off state OFF
  - 1 - Compression level back off state ON

**Examples**
- `CALC:MEAS2:CORR ON`
- `calculate:measure2:correction:state off`

**Query Syntax**
`CALCulate<cnum>:MEASure<mnum>:DISTortion:BACKoff[:STATe] <bool>`

**Return Type** Boolean

**Default** OFF
CALCulate\(<cnum>\):MEASure\(<mnum>\):DISTortion:MODE \(<char>\)

**Applicable Models:** All

*(Read-Write)* Displays phase or amplitude distortion.

**Parameters**
- \(<cnum>\) Channel number of the measurement (optional).
- \(<mnum>\) Measurement number for each measurement.
- \(<char>\) Choose from:
  - OFF
  - AMPM - Displays phase distortion (in degrees).
  - AMAM - Displays amplitude distortion (in dB).

**Examples**
- `CALC:MEAS:DIST:MODE AMAM`
- `calculate2:measure1:distortion:mode amam`

**Query Syntax**
- `CALCulate\(<cnum>\):MEASure\(<mnum>\):DISTortion:MODE?`

**Return Type**
- Character

**Default**
- OFF

---

CALCulate\(<cnum>\):MEASure\(<mnum>\):DISTortion:SLOPe:APERture \(<num>\)

**Applicable Models:** All

*(Read-Write)* Sets the aperture value over which the phase or gain slope will be calculated.

**Parameters**
- \(<cnum>\) Channel number of the measurement (optional).
- \(<mnum>\) Measurement number for each measurement.
- \(<num>\) Aperture value.

**Examples**
- `CALC1:MEAS2:DIST:SLOP:APER 1 dB`
- `calculate2:measure2:distortion:slope:aperture 1 dB`

**Query Syntax**
- `CALCulate\(<cnum>\):MEASure\(<mnum>\):DISTortion:SLOPe:APERture?`

**Return Type**
- Numeric

**Default**
- 1 dB
CALCulate<cnum>:MEASure<mnum>:DISTortion:SLOPe[:STATe] <bool>

Applicable Models: All

(Read-Write) Enables/disables phase slope (AMPM) or gain slope (AMAM) over the slope aperture to be displayed.

Parameters

- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <bool> Compression level back off state. Choose from:
  0 - Phase or gain slope over aperture state OFF
  1 - Phase or gain slope over aperture state ON

Examples

- CALC:MEAS2:DIST:SLOP ON
- calculate:measure2:distortion:slope:state off

Query Syntax

CALCulate<cnum>:MEASure<mnum>:DISTortion:SLOPe[:STATe]?

Return Type

Boolean

Default

OFF
CALCulate:MEASure:BLIMit Commands

These commands are for setting up bandwidth tests.

```
CALCulate:MEASure:BLIMit
  BWIDth
    | THReshold
  DISPlay
    | MARKer
    | STATe
  FAIL?
  MAXimum
  MI Nimum
  REPort
    | [:DATA]?
    [:STATe]
```

Click a keyword to view the command details.

See Also

- Calibrating with SCPI
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

CALCulate<cnum>:MEASure<mnum>:BLIMit:BWIDth:THReshold <value>
**Applicable Models:** All

**(Read-Write)** Sets bandwidth threshold value (attenuation from the peak) of the bandwidth test.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<value>` Bandwidth N dB points

**Examples**
- `CALC:MEAS:BLIM:BWID:THR 5`

**Query Syntax**
- `CALCulate<cnum>:MEASure<mnum>:BLIMit:BWIDth:THReshold?`

**Return Type**
- Numeric
- Default: 3

---

**CALCulate<cnum>:MEASure<mnum>:BLIMit:DISPlay:MARKer:STATe <bool>**

**Applicable Models:** All

**(Read-Write)** Turns ON/OFF the bandwidth value display of the bandwidth test, for the active trace of selected channel.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` ON or 1 - Turns limit testing ON.
  OFF or 0 - Turns limit testing OFF.

**Examples**
- `CALC:MEAS:BLIM:DISP:MARK:STAT ON`
- `calculate2:measure2:blimit:display:marker:state off`

**Query Syntax**
- `CALCulate<cnum>:MEASure<mnum>:BLIMit:DISPLay:BWIDth:STATe?`

**Return Type**
- Boolean
- Default: OFF

---

**CALCulate<cnum>:MEASure<mnum>:BLIMit:FAIL?**
Applicable Models: All

(Read-only) Get the bandwidth limit test results, for the active trace of selected channel.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` Boolean

- 0 is returned when Pass
- 1 is returned when Fail

**Examples**

CALC:MEAS:BLIM:FAIL?
calculate2:measure2:blimit:fail?

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:BLIMit:FAIL?

**Return Type**

Boolean

Default Not Applicable

CALCulate<cnum>:MEASure<mnum>:BLIM:MAXimum `<max>`

Applicable Models: All

(Read-Write) Sets/gets the upper limit value of the bandwidth test, for the selected channel.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<max>` Maximum bandwidth

**Examples**

CALC:MEAS:BLIM:MAX 1E6
calculate2:measure2:blimit:maximum 1E6

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:BLIMit:MAXimum?

**Return Type**

Numeric

Default

CALCulate<cnum>:MEASure<mnum>:BLIM:MINimum `<min>`
**Applicable Models:** All

**(Read-Write)** Sets or returns the lower limit value of the bandwidth test, for the selected channel.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<min>` Minimum bandwidth

**Examples**
- `CALC:MEAS:BLIM:MIN 1E6`
- `calculate2:measure2:blimit:minimum 1E6`

**Query Syntax**
- `CALCulate<cnum>:MEASure<mnum>:BLIMit:MINimum?`

**Return Type** Numeric

---

**Applicable Models:** All

**(Read-only)** Read the bandwidth value of the bandwidth test, for the active trace of selected channel.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**
- `CALC:MEAS:BLIM:REP:DATA?`
- `calculate2:measure2:blimit:report:DATA?`

**Query Syntax**
- `CALCulate<cnum>:MEASure<mnum>:BLIMit:REPort:DATA?`

**Return Type** Variant

Default OFF

---

**CALCulate<cnum>:MEASure<mnum>:BLIMit[:STATe]**
Applicable Models: All

(Read-Write) Turns ON/OFF the bandwidth test function, for the active trace of selected channel.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` ON or 1 - Turns limit testing ON.
  OFF or 0 - Turns limit testing OFF.

Examples

CALC:MEAS:LIM:STAT ON  
calculate2:measure2:limit:state off

Query Syntax

CALCulate<cnum>:MEASure<mnum>:BLIMit:DIPLay:MARKer:STATe?

Return Type

Boolean

Default

OFF
CALCulate:MEASure:Correction Commands

Controls error correction functions.

<table>
<thead>
<tr>
<th>CALCulate:MEASure:CORRection</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDELay</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>[:STATe]</td>
</tr>
<tr>
<td>TYPE</td>
</tr>
</tbody>
</table>

Click a keyword to view the command details.

See Also

- Calibrating with SCPI
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

CALCulate<cnum>:MEASure<mnum>:CORRection:EDELay:DISTance <num>
Applicable Models: All

(Read-Write) Sets the electrical delay in physical length (distance) for the selected measurement.

**Parameters**
- `<cnum>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
- `<num>`: Electrical delay in distance.


Use `SENS:CORR:RVEL:COAX <num>` to set Velocity factor.

This parameter supports MIN and MAX as arguments. Learn more.

**Examples**
```
CALC1:MEAS2:CORR:EDEL:DIST 5
```
```
calculate2:measure2:correction:edelay:distance .003
```

**Query Syntax**
`CALCulate<cnum>:MEASure<mnum>:CORRection:EDELay:DISTance?`

**Return Type**
Numeric
- **Default**: 0

---

CALCulate<cnum>:MEASure<mnum>:CORRection:EDELay:MEDium <char>

**Applicable Models**: All

(Read-Write) Sets the media used when calculating the electrical delay.

**Parameters**
- `<cnum>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
- `<num>`: Choose from: **COAX** for coaxial medium, **WAVE**guide for waveguide medium.

**Examples**
```
CALC:MEAS2:CORR:EDEL:MED COAX
```
```
calc3:measure2:correction:edelay:medium waveguide
```

**Query Syntax**
`CALCulate<cnum>:MEASure<mnum>:CORRection:EDELay:MEDium?`

**Return Type**
Character
- **Default**: COAX
CALCulate\textsubscript{\textless\textit{cnum}\textgreater}:MEASure\textsubscript{\textless\textit{mnum}\textgreater}:CORRection:EDELay[\textsubscript{\textless\textit{TIME}\textgreater}] \textless\textit{num}\textgreater

**Applicable Models:** All

**(Read-Write)** Sets the electrical delay for the selected measurement.

**Parameters**

- \textless\textit{cnum}\textgreater: Channel number of the measurement (optional).
- \textless\textit{mnum}\textgreater: Measurement number for each measurement.
- \textless\textit{num}\textgreater: Electrical delay in seconds. Choose any number between \textbf{-10.00} and \textbf{10.00}
  
  Use \texttt{SENS:CORR:RVEL:COAX <\textit{num}>} to set Velocity factor.
  
  This parameter supports MIN and MAX as arguments. [Learn more.]

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{CALC1:MEAS2:CORR:EDEL:TIME 1NS}</td>
<td>\texttt{calculate2:measure2:correction:time 0.5e-12}</td>
</tr>
</tbody>
</table>

**Query Syntax**

\texttt{CALCulate<\textless\textit{cnum}\textgreater>:MEASure<\textless\textit{mnum}\textgreater>:CORRection:EDELay[:\textless\textit{TIME}\textgreater]?}

**Return Type**

- **Numeric**
- **Default** 0 seconds

---

CALCulate\textsubscript{\textless\textit{cnum}\textgreater}:MEASure\textsubscript{\textless\textit{mnum}\textgreater}:CORRection:EDELay:UNIT \textless\textit{char}\textgreater

**Applicable Models:** All

**(Read-Write)** Sets and returns the units for specifying electrical delay in physical length (distance).

**Parameters**

- \textless\textit{cnum}\textgreater: Channel number of the measurement (optional).
- \textless\textit{mnum}\textgreater: Measurement number for each measurement.
- \textless\textit{char}\textgreater: Units for delay in distance. Choose from:
  
  - METer
  - FEET
  - INCH

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{CALC:MEAS2:CORR:EDEL:UNIT MET}</td>
<td>\texttt{calc3:meas2:corr:edelay:unit inch}</td>
</tr>
</tbody>
</table>

**Query Syntax**

\texttt{CALCulate<\textless\textit{cnum}\textgreater>:MEASure<\textless\textit{mnum}\textgreater>:CORRection:EDELay:UNIT?}
CALCulate<cnum>:MEASure<mnum>:CORRection:EDELay:WGCutoff <num>

**Applicable Models:** All

(Read-Write) Sets the waveguide cutoff frequency used when the electrical delay media is set to WAVEguide. (See CALCulate:MEAS:CORRection:EDELay:MEDium <char>.)

**Parameters**
- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <num> Waveguide cutoff frequency used with the electrical delay calculation.

This parameter supports MIN and MAX as arguments. Learn more.

**Examples**
```
CALC:MEAS2:CORR:EDEL:WGC 18.067 GHz
```
```
```

**Query Syntax**
CALCulate<cnum>:MEASure<mnum>:CORRection:EDELay:WGCutoff?

**Return Type** Numeric

**Default** 45 MHz

CALCulate<cnum>:MEASure<mnum>:CORRection[:STATe] <bool>

**Applicable Models:** All

(Read-Write) Turns error correction ON or OFF for the selected measurement on the specified channel.

To turn error correction ON or OFF for a channel, use SENS:CORR:STATe.

**Parameters**
- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
Correction state. Choose from:

0 - Correction OFF

1 - Correction ON

**Examples**

```
CALC:MEAS2:CORR ON
```

```
calculate:measure2:correction:state off
```

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:CORRection:STATe?

**Return Type**

Boolean

**Default**

Not Applicable

---

CALCulate<cnum>:MEASure<mnum>:CORRection[:STATe]:INDicator?

**Applicable Models:** All

(Read-only) Returns the error correction state for the selected measurement on the specified channel.

To turn error correction ON or OFF for a channel, use SENS:CORR:STATe.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS2:CORR:IND?
```

```
calculate2:measure2:correction:state:indicator?
```

**Return Type**

Character

**NONE** - No error correction

**MAST** (Master) - Original error correction terms

**INT** - Error terms are interpolated. Learn more.

**DELT** - Delta Match calibration terms. Learn more.

**INV** - Error terms are not valid

**Default**

NONE

---

CALCulate<cnum>:MEASure<mnum>:CORRection:TYPE <string>
Applicable Models: All

(Read-Write) Sets the Cal Type for the selected measurement on the specified channel. This is used when a Cal Set is applied. Learn more about applying Cal Types.

- Use SENS:CORR:TYPE:CAT? to list the Cal Types in the VNA.
- Use SENS:CORR:CSET:TYPE:CAT? to list the Cal Types contained in the active Cal Set for the channel.
- Use SENS:CORR:COLL:METH to set the Cal type to perform a new Unguided calibration,

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<string>` (String) Cal type. Case sensitive. Use one of the following:

For Full Calibrations (NO Power Cal included):

This command does not distinguish between TRL and SOLT. The same number of error terms is applied for both Cal Types.

"Full <n> Port(x,y,z...)"

where

<n> = the number of ports to calibrate
x,y,z = the port numbers to calibrate

For example:

"Full 4 Port(1,2,3,4)"

For Full Calibrations (including Power Cal):

After the Full <n> port, include the string, "with power"

For example:

"Full 4 Port with power(1,2,3,4)"

For Response Calibrations:

"Response(param)" OR

"ResponseAndIsolation(param)"
Where param =

- S-parameter. For example
  - "Response(S21)"
  - "ResponseAndIsolation(A/R)"
- Single or ratioed receivers using either logical receiver notation or physical receiver notation. For example:
  - "Response(A)"
  - "ResponseAndIsolation(a3/b4)"

For Enhanced Response Calibrations:

"EnhancedResp(sourcePort, recPort)

Where:

- sourcePort = stimulus port number
- recPort = receiver port number

For FCA Calibrations:

Learn more about this setting.

- "SMC_2P" (Response + Input + Output) All four sweeps required. Most accurate.

  **Note:** The “SMC_2P” is the only correction type that automatically forces the reverse measurements (SC12, RevOPwr, RevIPwr). If you require any of the reverse measurements while using a lesser correction type, either include those parameters in your channel or use the SENSE:MIXer:REVerse command. Keep in mind that adding the reverse measurements or forcing the reverse sweep using the SENSE:MIXer:REVerse command will increase the number of sweeps required in the channel.

- "SMCRsp+IN" No Output match. Saves two sweeps.
- "SMCRsp+OUT" No Output match. Saves one sweep.
- "SMCRsp" No Input or Output match. Saves two sweeps.

For VMC, multiple Cal types are not available.
For Gain Compression Cal

where r = receive port; s = source port

- "GCA 2P (r,s)" - full 2-port cal
- "GCA Enh Resp (r,s)" - Enhanced Response Cal

Examples

CALC:MEAS2:CORR:TYPE "Scalar Mixer Cal"

Query Syntax

CALCulate<cnum>:MEASure<mnum>:CORRection:TYPE?

Return Type

String

Default

Not Applicable
CALCulate:MEASure:DATA Commands

Controls writing and reading VNA measurement data.

<table>
<thead>
<tr>
<th>CALCulate:MEASure:DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Click a keyword to view the command details.

See Also

- Calibrating with SCPI
- Synchronizing the Analyzer and Controller
- SCPI Command Tree
- See CALCulate:MEASure:X:VALues for stimulus point data.

(Write) CALCulate<cnum>:MEASure<mnum>:DATA:<char> <data>

(Read) CALCulate<cnum>:MEASure<mnum>:DATA:<char>?

Applicable Models: All

Reads or writes Measurement data, Memory data, or Normalization Divisor data from the Data Access Map location.

- For Measurement data, use FDATA or SDATA
- For Memory data, use FMEM or SMEM. When querying memory, you must first store a trace into memory using CALC:MEAS:MATH:MEMorize.
- For Normalization Divisor (Receiver Power Cal error term) data, use SDIV.
- Use FORMAT:DATA to change the data type (<REAL,32>, <REAL,64> or <ASCii,0>).
- Use FORMAT:BORDer to change the byte order. Use “NORMal” when transferring a binary block from LabView or VEE. For other programming languages, you may need to "SWAP" the byte order.

Equation Editor Notes:

- When equation editor is active on a trace in a standard S-parameter channel, Calc:Data returns the data from the parameter on the trace that was measured last. For example, for the equation “S22 + S33 + S11”, then S33 is the last measured parameter because it uses source port 3.
- In applications, if equation editor is active and the original parameter for the trace is not requested anywhere in the channel, then zeros are returned. If the original parameter is being measured within the channel, then data for the original parameter is returned.
- In general, if an equation contains no measurement parameters, then data for the original parameter is returned.

Note: The Calc:Data SCORR command to read/write error terms is Superseded with
SCORR commands do NOT accommodate greater than 12 error terms.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement (optional).</td>
<td></td>
</tr>
<tr>
<td>&lt;mnum&gt;</td>
<td>Measurement number for each measurement.</td>
<td></td>
</tr>
<tr>
<td>&lt;char&gt;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**FDATA**

Formatted measurement data to or from Data Access Map location. Display (access point 2).

**Note:** When querying FDATA, data is received in degrees. When setting phase using FDATA, the command expects the data in radians.

- Corrected data is returned when correction is ON.
- Uncorrected data is returned when correction is OFF.
- Returns TWO numbers per data point for Polar and Smith Chart format. For both formats, the data is the real and imaginary data. To retrieve the resistance and reactance of a trace of data, use the Z-Reflection conversion feature.
- Returns one number per data point for all other formats.
- Format of the read data is same as the displayed format.

**SDATA**

Complex measurement data.

Writes data to Data Access Map location. Raw Measurement (access point 0).

**Note:** Write access is forbidden on the standard channel if factory calibration is on (applies to ENA/PXI/Streamline, not to PNA); to disable factory calibration use SYST:FCOR:CHAN:COUP:STAT
<table>
<thead>
<tr>
<th><strong>OFF command.</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• When writing corrected data, and correction is ON, it will be corrected again, resulting in meaningless data.</td>
</tr>
<tr>
<td>Reads data from Apply Error Terms (access point 1).</td>
</tr>
<tr>
<td>• Returns TWO numbers per data point.</td>
</tr>
<tr>
<td>• Corrected data is returned when correction is ON.</td>
</tr>
<tr>
<td>• Uncorrected data is returned when correction is OFF.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>FMEM</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Formatted memory data to or from Data Access Map location. Memory result (access point 4).</td>
</tr>
<tr>
<td>• Returns TWO numbers per data point for Polar and Smith Chart format.</td>
</tr>
<tr>
<td>• Returns one number per data point for all other formats.</td>
</tr>
<tr>
<td>• Format of the read data is same as the displayed format.</td>
</tr>
<tr>
<td>• Returned data reflects the correction level (ON</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>SMEM</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Complex memory data to or from Data Access Map location. Memory (access point 3).</td>
</tr>
<tr>
<td>• Returns TWO numbers per data point.</td>
</tr>
<tr>
<td>• Returned data reflects the correction level (ON</td>
</tr>
<tr>
<td>• Returned data reflects the</td>
</tr>
</tbody>
</table>
CALCulate<cnum>:MEASure<mnum>:DATA:BUFFer:X? <bufferName>

Applicable Models: All

(Read-only) Retrieves frequency tone data from the modulation distortion measurement.

Parameters

- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <bufferName> (string) – Name of the buffer to be read.

Note: The following names assume a 2-port VNA. Also, by default, the parameter names assume that VNA Port 1 is connected to the DUT input and VNA Port 2 is connected to the DUT output. If the RF Path settings are changed from the default, the names will change accordingly. For example, if the VNA has 4 ports and Port 4 is connected to the DUT output, the names will change to "PIn1", "POut4", "S11", "S41", etc.

Choose from:

- "POut2" - Power Out
- "PIn1" - Power In
- "MSig2" - Modulation Signal Out
- "MDist2" - Modulation Distortion Out
- "MGain21" - Modulation Gain
- "MComp21" - Modulation Compression
- "PGain21" - Power Gain
- "S11" - Linear Input Match

The data was stored into memory.

Examples

CALC:MEAS:DATA:FDATA Data(x)
calculate2:measure2:DATA:sdata data(r,i)

See another example using this command.

Return Type: Block data

Default: Not Applicable
"S21" - Linear Gain

"LPin1" - Linear Power In

"LPOut1" - Linear Reflected Power In

"LPOut2" - Linear Power Out

Examples

CALC:MEAS1:DATA:BUFF:X? "POut2"

Return Type String

Default Not Applicable

CALCulate<cnm>:MEAS<mnms>:DATA:BUFF:Y? <bufferName>

Applicable Models: All

(Read-only) Retrieves trace data (Y data) from the modulation distortion measurement.

Parameters

<cnm> Channel number of the measurement (optional).

<mnms> Measurement number for each measurement.

<bufferName> (string) – Name of the buffer to be read.

Note: The following names assume a 2-port VNA. Also, by default, the parameter names assume that VNA Port 1 is connected to the DUT input and VNA Port 2 is connected to the DUT output. If the RF Path settings are changed from the default, the names will change accordingly. For example, if the VNA has 4 ports and Port 4 is connected to the DUT output, the names will change to "Pin1", "POut4", "S11", "S41", etc.

Choose from:

"POut2" - Power Out

"Pin1" - Power In

"MSig2" - Modulation Signal Out

"MDist2" - Modulation Distortion Out

"MGain21" - Modulation Gain

"MComp21" - Modulation Compression
"PGain21" - Power Gain

"S11" - Linear Input Match

"S21" - Linear Gain

"LPin1" - Linear Power In

"LPOut1" - Linear Reflected Power In

"LPOut2" - Linear Power Out

Examples

```
CALC:MEAS1:DATA:BUFF:Y? "POut2"
```

**Return Type** String

**Default** Not Applicable

CALCulate<cnum>:MEASure<mnum>:DATA:RAW <string>,<dataBlock>

**Applicable Models:** All

(Write-Read) Sets the raw data for the specified raw parameter buffer. For unratioed parameters, the source port separator can be either ',' or '_'. This command applies to a Standard channel only. Application channels will return an error.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<string>` String - (Not case-sensitive) Raw parameter buffer. For example, "S11".
- `<dataBlock>` String - (Not case-sensitive) Raw measurement data.

**Examples**

```
```

```
```

**Query Syntax** CALCulate<ch>:MEASure<mnum>:DATA:RAW? "<string>"

**Return Type** Block data

**Default** Not Applicable

CALCulate<cnum>:MEASure<mnum>:DATA:RAW:CATalog?
Applicable Models: All

(Read-only) Returns the list of raw parameters associated with the measurement <mnum>. These are the raw parameters that are actually used by the error correction; the <mnum> measurement may not strictly depend on all of those. For example, 2 port S-parameters. If the DUT has transmission, then changing raw S11 will modify the corrected S22. If transmission is negligible, S22 is unaffected. The list is returned as strings comma separated. For unratioed parameters, the source port separator is ‘_’ instead of ‘,’ to avoid confusion in tokenization. This command applies to a Standard channel only. Application channels will return an error.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Example**
```
CALC:MEAS1:DATA:RAW:CAT?
```

**Return Type** String

**Default** Not Applicable

---

CALCulate<cnum>:MEASure<mnum>:DATA:SNP? <n>

Applicable Models: All

(Read-only) Reads SnP data from the selected measurement. Learn more about SnP data.

This command is valid ONLY with standard S-parameter measurements.

**Notes**
- This command returns SnP data without header information, and in columns, not in rows as .SnP files. This means that the data returned from this command sends all frequency data, then all Sx1 magnitude or real data, then all Sx1 phase or imaginary data, and so forth.
- To avoid frequency rounding errors, specify FORMat:DATA <Real,64> or <ASCii, 0>

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Amount of data to return. If unspecified, `<n>` is set to 2. The number you specify must be less than or equal to the number of available ports on the VNA.

Choose from:
1 (S1P) returns 1-Port data for the active measurement if the active measurement is a reflection parameter such as S11 or S22. The behavior is UNDEFINED if the active measurement is a transmission parameter such as an S21.

2 (S2P) returns data for the four 2 port parameters associated with the current measurement. Default. Data that is not available is zero-filled.

3 (S3P) returns data for the nine 3 port parameters associated with the current measurement. Data that is not available is zero-filled.

4 (S4P) returns data for the sixteen 4 port parameters associated with the current measurement. Data that is not available is zero-filled.

SnP data can be output using several data formatting options. See MMEM:STOR<e>:TRACe:FORMat:SNP.

See also MMEM:STOR<e>:<snp>

Examples CALC:MEAS<e>:DATA:SNP? 1

Return Type Depends on FORM:DATA.

Default Not Applicable

CALCulate<e>:MEAS<e>:DATA:SNP:PORTs? <"x,y,z"> 

Applicable Models: All

(Read-only) Reads SNP data from the selected measurement for the specified ports. Learn more about SNP data.

This command is valid ONLY with standard S-parameter measurements.

Notes

- This command returns SNP data without header information, and in columns, not in rows as .SnP files. This means that the data returned from this command sends all frequency data, then all Sx1 magnitude or real data, then all Sx1 phase or imaginary data, and so forth.

- To avoid frequency rounding errors, specify FORM:DATA <Real,64> or <ASCii, 0>

- Data that is not available is zero-filled.

- For sweeps with a large number of data points, always follow this command with *OPC? Learn more.
Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<"x,y,z">` Comma or space delimited port numbers for which data is requested, enclosed in quotes.

SNP data can be output using several data formatting options. See `MMEM:STORe:TRACe:FORMat:SNP`.

Examples

```
CALC:MEAS2:DATA:SNP:PORTs? "1,2,4,5,7" 'read data for these ports
```

Return Type

Depends on `FORMat:DATA`

Default Not Applicable

```
CALCulate<cnum>:MEASure<mnum>:DATA:SNP:PORTs:SAVE <"x,y,z">,<filename>[, FAST]
```

Applicable Models: All

(Write-only) Saves SNP data from the selected measurement for the specified ports. Learn more about SNP data.

- The Normal vs Mixed Mode selection is NOT used as it is in the Choose Ports dialog. Instead, data is returned as it is displayed on the trace. If the selected measurement is Mixed Mode (balanced), then balanced data is returned. If the selected measurement is an S-parameter, then S-parameter data is returned.
- This command is valid ONLY with the Standard measurement class (NOT applications).
- Data that is not available is zero-filled.
- For sweeps with a large number of data points, always follow this command with *OPC? Learn more.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<"x,y,z">` String Comma or space delimited port numbers for which data is requested, enclosed in quotes.
- `<filename>` String Path, filename, and suffix of location to store the SNP data, enclosed in quotes. The suffix is not checked for accuracy. If saving 2 ports, specify "filename.s2p"; If saving 4 ports, specify "filename.s4p.", and so forth.

SNP data can be output using several data formatting options. See
MMEM:STORe:TRACe:FORMat:SNP.

[, FAST] Reduce the saving time

:SENS:CORR:CACH:MODE should be set at ON.

The correction must cover all the ports of the SNP port list.

The active measurement must be a corrected S Parameter defined by the port list for the SNP requests. EG: "S33" is not a proper selected measurement with CALC:DATA:SNP:PORTS? <1,2>

Examples

CALC:MEAS2:DATA:SNP:PORTs:Save '1,2,4','D:\MyData.s3p';*OPC?

Return Type

Depends on FORMat:DATA

Default

Not Applicable

CALCulate<cnum>:MEASure<mnum>:DATA:X:AXiS <string>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Write-Read) Sets the X-axis of the selected measurement to a DC Source. This command does not change the default setting for new traces.

Parameters

<cnum> Channel number of the measurement (optional).
<mnum> Measurement number for each measurement.
<string> String - (Not case-sensitive) For all channels EXCEPT DIQ, choose from the following:

"Default" - The default X-axis setting for the selected measurement. For Application measurements, the X-Axis domain is set with specific commands.

"AO1" - Internal DC source #1

"AO2" - Internal DC source #2

Note: For DIQ channels, see CALC:MEAS:X:AXIS:DOMain

For Modulation Distortion Channels (MOD), choose from:

"SA Frequency" - SA display showing the SA frequency settings.
"Power In" - Displays input power sweep.

"Power Out" - Displays output power sweep.

"Measured Carrin" - Measured input band power.

"Measured CarrOut" - Measured output band power.

For Modulation Distortion Converters Channels (MODX), choose from:

"SA Freq In" - SA display showing mixer input range.

"SA Freq Out" - SA display showing mixer output range.

"Power In" - Displays input power sweep.

"Power Out" - Displays output power sweep.

"Measured Carrin" - Measured input band power.

"Measured CarrOut" - Measured output band power.

Examples

CALC:MEAS:DATA:X:AXIS 'Default'
calculate:measure:data:x:axis "AO1"

Query Syntax

CALCulate<ch>:MEASure<mnum>:DATA:X:AXIS?

Return Type

String

Default

"Default"

CALCulate<cnum>:MEASure<mnum>:DATA:X:AXIS:DOMain <string>
**Applicable Models:** N522xB, N523xB, N524xB

*(Write-Read)* Sets and returns the X-Axis domain of the selected DIQ measurement.

**Parameters**

- `<cnum>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.

<table>
<thead>
<tr>
<th>Choose one of these:</th>
<th>Then set X-Axis Source (<em>CALC:MEAS:DATA:X:AXIS</em>) using one of these as the argument.</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Frequency&quot;</td>
<td>&quot;F1&quot;, &quot;F2&quot;, etc.</td>
</tr>
<tr>
<td>&quot;Power&quot;</td>
<td>Source port: &quot;Port 1&quot;, &quot;Port 2&quot;, etc.</td>
</tr>
<tr>
<td>&quot;Phase&quot;</td>
<td>Source port: &quot;Port 1&quot;, &quot;Port 2&quot;, etc.</td>
</tr>
<tr>
<td>&quot;DC&quot;</td>
<td>DC Source: &quot;AO1&quot;, &quot;AO2&quot;</td>
</tr>
<tr>
<td>&quot;Points&quot;</td>
<td>&quot;Points&quot;</td>
</tr>
</tbody>
</table>

**Example**

2. *CALC:MEAS:DATA:X:AXIS "Port 1"*

**Query Syntax**

*CALCulate<ch>:MEASure<mnum>:DATA:X:AXIS:DOMain?*

**Return Type**

String

**Default**

*CALC:MEAS:DATA:X:AXIS:DOMain: "Frequency"
CALC:MEAS:DATA:X:AXIS: "F1"*

*CALCulate<ch>:MEASure<mnum>:DATA:X:AXIS:FIXed:PARameter <string>*
Applicable Models: N524xB models with Option S93070xB

(Write-Read) Sets the X-axis fixed parameter for a power sweep measurement in a Modulation Distortion channel.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<string>` String - (Not case-sensitive) For Modulation Distortion Channels (MOD), choose from:
  - "SA Frequency" - SA display showing the SA frequency settings.
  - For Modulation Distortion Converters Channels (MODX), choose from:
    - "SA Freq In" - SA display showing mixer input range.
    - "SA Freq Out" - SA display showing mixer output range.

Examples

```
calculate:measure2:data:x:axis:fixed:parameter "SA Freq In"
```

Query Syntax


Return Type

String

Default

"SA Frequency"

Applicable Models: N524xB models with Option S93070xB

(Write-Read) Sets the X-axis fixed parameter spectrum analyzer frequency value for a power sweep measurement in a Modulation Distortion channel.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Fixed parameter value.

Examples

```
CALC:MEAS2:DATA:X:AXIS:FIX:PAR:VAL 1.5e9
```
```
calculate:measure2:data:x:axis:fixed:parameter:value 1.5e9
```

Query Syntax

```
```

Return Type

Numeric

Default

Not Applicable

---

CALCulate<cnum>:MEASure<mnum>:DATA:X[:Values]?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Returns the stimulus values for the selected measurement in the current units. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

This command can be used for all Measurement Classes.

Note: To avoid frequency rounding errors, specify FORM:DATA <Real,64> or <ASCii, 0>

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

Examples

```
1. Calc:Par:Sel "MyGCATrace"
2. CALC:MEAS:DATA:X?
```

Return Type

Depends on FORM:DATA command

Default

Not applicable
CALCulate:MEASure:FILter Commands

Controls the gating function used in time domain measurements. The gated range is specified with either (start / stop) or (center / span) commands.

```
CALCulate:MEASure:FILter
 [:GATE]
  | COUPle
  | PARameters
  | TIME
   | CENTER
   | SHAPE
   | SPAN
   | START
   | STATE
  | STOP
 [:TYPE]
```

Click a keyword to view the command details.

**see Also**

- Learn about Gating
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

CALCulate<cnm>:MEAS<mnm>:FILTer[:GATE]:COUPle:PARameters <num>
(Read-Write) Specifies the time domain gating parameters to be coupled. The settings for those parameters will be copied from the selected measurement to all other measurements on the channel.

- To enable Trace Coupling, use SENS:COUP:PAR
- To specify Transform parameters to couple, use CALC:MEAS:TRAN:COUP:PAR

Learn more about Time Domain Trace Coupling

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` (Numeric) Parameters to couple. To specify more than one parameter, add the numbers.
  
  1 - Gating Stimulus (Start, Stop, Center, and Span TIME settings.)
  
  2 - Gating State (ON / OFF)
  
  4 - Gating Shape (Minimum, Normal, Wide, and Maximum)
  
  8 - Gating Type (Bandpass and Notch)

**Examples**

- To couple all parameters:
  
  `CALC:MEAS2:FILT:COUP:PAR 15`
  
- To couple Stimulus and Type:
  

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:FILTer[:GATE]:TIME:CENTer <num>

**Return Type**

Numeric

**Default**

13 (All parameters except 2 - Gating State)
Applicable Models: All

(Read-Write) Sets the gate filter center time.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Center time in seconds; Choose any number between: ± (number of points-1) / frequency span

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

```
CALC:MEAS2:FILT:GATE:TIME:CENT -5 ns
```

Query Syntax

CALCulate<cnum>:MEASure<mnum>:FILTer[:GATE]:TIME:CENTer?

Return Type

Numeric

Default

0

CALCulate<cnum>:MEASure<mnum>:FILTer[:GATE]:TIME:SHAPe <char>

Applicable Models: All

(Read-Write) Sets the gating filter shape when in time domain.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<char>` Choose from
  - MAXimum - the widest gate filter available
  - WIDE -
  - NORMal -
  - MINimum - the narrowest gate filter available

Examples

```
calculate2:measure2:filter:time:shape normal
```

Query Syntax

CALCulate<cnum>:MEASure<mnum>:FILTer[:GATE]:TIME:SHAPe?

Return Type

Character

Default

NORMal
CALCulate<cnum>:MEASure<mnum>:FILTer[:GATE]:TIME:SPAN <num>

Applicable Models: All

(Read-Write) Sets the gate filter span time.

Parameters

- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <num> Time span in seconds; Choose any number between:
  0 and $2 \times \frac{(number\ of\ points-1)\ /\ frequency\ span}$

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

- CALC:MEAS2:FILT:TIME:SPAN 5 ns
- calculate2:measure2:filter:time:span maximum

Query Syntax

CALCulate<cnum>:MEASure<mnum>:FILTer[:GATE]:TIME:SPAN?

Return Type Numeric

Default 20 ns

CALCulate<cnum>:MEASure<mnum>:FILTer[:GATE]:TIME:STARt <num>

Applicable Models: All

(Read-Write) Sets the gate filter start time.

Parameters

- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <num> Start time in seconds; any number between:
  $\pm \frac{(number\ of\ points-1)\ /\ frequency\ span}$

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

- CALC:MEAS2:FILT:TIME:STAR 1e-8
- calculate2:measure2:filter:gate:time:start minimum

Query Syntax

CALCulate<cnum>:MEASure<mnum>:FILTer[:GATE]:TIME:STARt?

Return Type Numeric
### CALCulate<cnum>:MEASure<mnum>:FILTer[:GATE]:TIME:STATe <boolean>

**Applicable Models:** All

*(Read-Write)* Turns gating state ON or OFF.

**Note:** Sweep type must be set to Linear Frequency in order to use Transform Gating.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<boolean>` ON (or 1) - turns gating ON. OFF (or 0) - turns gating OFF.

**Examples**
- `CALC:MEAS2:FILT:TIME:STAT ON`
- `calculate2:measure2:filter:gate:time:state off`

**Query Syntax** `CALCulate<cnum>:MEASure<mnum>:FILTer[:GATE]:TIME:STATe?`

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** OFF

---

### CALCulate<cnum>:MEASure<mnum>:FILTer[:GATE]:TIME:STOP <num>

**Applicable Models:** All

*(Read-Write)* Sets the gate filter stop time.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Stop time in seconds; any number between:
  \[\pm \frac{(\text{number of points}-1)}{\text{frequency span}}\]

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**
- `CALC:MEAS2:FILT:TIME:STOP -1 ns`
- `calculate2:measure2:filter:gate:time:stop maximum`

**Query Syntax** `CALCulate<cnum>:MEASure<mnum>:FILTer[:GATE]:TIME:STOP?`
CALCulate\(<cnum>\):MEASure\(<mnum>\):FILTer\([\cdot]:GATE\)[\cdot]:TIME\([\cdot]:TYPE\) \(<char>\)

**Applicable Models:** All

(Read-Write) Sets the type of gate filter used.

**Parameters**

- \(<cnum>\) Channel number of the measurement (optional).
- \(<mnum>\) Measurement number for each measurement.
- \(<char>\) Choose from:

  - **BPASs** - Includes (passes) the range between the start and stop times.
  - **NOTCh** - Excludes (attenuates) the range between the start and stop times.

**Examples**

```
CALC:MEAS2:FILT:TIME BPAS
calculate2:measure2:filter:gate:time:type notch
```

**Query Syntax**

```
CALCulate\(<cnum>\):MEASure\(<mnum>\):FILTer\([\cdot]:GATE\)[\cdot]:TIME\([\cdot]:TYPE\)?
```

**Return Type** Character

**Default** BPAS
CALCulate:MEASure:FUNCtion Commands

- DATA?
- DOMain
  - USER
  - [:RANGe]
  - STARt
  - STOP
- EXECute
- STATistics
  - [:STATe]

Click a keyword to view the command details.

**see Also**

- Learn about Trace Statistics
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

CALCulate<cn num>:MEASure<mn num>:FUNCtion:DATA?
Applicable Models: All

(Read-only) Returns the trace statistic data for the selected statistic type for the specified channel. Select the type of statistic with CALC:MEAS:FUNC:TYPE.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

Return Type Depends on FORM:DATA

Example
- CALCulate2:MEASure2:FUNCtion:DATA?

Default Not applicable

CALCulate<cnum>:MEASure<mnum>:FUNCtion:DOMain:USER[:RANGe] <range>

Applicable Models: All

(Read-Write) Sets the range used to calculate trace statistics. Each channel has 16 user ranges. The x-axis range is specified with the CALC:MEAS:FUNC:DOM:USER:START and STOP commands.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<range>` Range number. Choose from: 0 to 16
  - 0 is Full Span of the current x-axis range
  - 1 to 16 are user-specified ranges

Examples
- CALC:MEAS2:FUNC:DOM:USER 4
- calculate2:measure2:function:domain:user:range 0

Query Syntax
- CALCulate<cnum>:MEASure<mnum>:FUNCtion:DOMain:USER[:RANGe]?

Return Type Numeric

Default 0 - Full Span

CALCulate<cnum>:MEASure<mnum>:FUNCtion:DOMain:USER:STARt <range>, <start>
Applicable Models: All

(Read-Write) Sets the start of the specified user-domain range.

To apply this range, use `CALC:MEAS:FUNC:DOM:USER`

To set the stop of the range, use `CALC:MEAS:FUNC:DOM:USER:STOP`.

**Note:** This command does the same as `CALC:MEAS:MARK:FUNC:DOM:USER:STAR`

### Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<range>` Range number that will receive the start value. Choose an integer between 1 and 16.
- `<start>` Start value of the specified range. Choose a real number between: the analyzer's Minimum and Maximum x-axis value.

### Examples

- `CALC:MEAS2:FUNC:DOM:USER:STAR 1,1e9`
- `calculate2:measure2:function:domain:user:start 2,2e9`

### Query Syntax

`CALCulate<cnum>:MEASure<mnum>:FUNCtion:DOMain:USER:STARt? <range>`

### Return Type

Numeric

**Default**

The analyzer's Minimum x-axis value

---

CALCulate<cnum>:MEASure<mnum>:FUNCtion:DOMain:USER:STOP <range>, <stop>

Applicable Models: All

(Read-Write) Sets the stop value of the specified user-domain range.

To apply this range, use `CALC:MEAS:FUNC:DOM:USER`.

To set the start of the range, use `CALC:MEAS:MARK:FUNC:DOM:USER:STAR`

**Note:** This command does the same as `CALC:MEAS:MARK:FUNC:DOM:USER:STOP`

### Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
<range> Range number that will receive the stop value. Choose an integer between 1 and 16

<stop> Stop value of the specified range. Choose a real number between: the analyzer's Minimum and Maximum x-axis value.

Examples

```
CALC:MEAS2:FUNC:DOM:USER:STOP 4,5e9
calculate2:measure2:function:domain:user:stop 3,8e9
```

Query Syntax

CALCulate<cnum>:MEASure<mnum>:FUNCtion:DOMain:USER:STOP?

Return Type Numeric

Default The analyzer's Maximum x-axis value

---

CALCulate<cnum>:MEASure<mnum>:FUNCtion:EXECute

Applicable Models: All

(Write-only) For the active trace of specified channel, executes the statistical analysis specified by the CALC:MEAS:FUNC:TYPE command.

Parameters

- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.

Examples

```
CALC:MEAS2:FUNC:EXEC
calculate2:measure2:function:execute
```

Query Syntax Not Applicable

Default Not Applicable

---

CALCulate<cnum>:MEASure<mnum>:FUNCtion:STATistics[:STATE] <ON|OFF>
Applicable Models: All

(Read-Write) Displays and hides the trace statistics (peak-to-peak, mean, standard deviation) on the screen.

The analyzer will display either measurement statistics or Filter Bandwidth statistics; not both.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<ON|OFF>`
  - ON - Displays trace statistics
  - OFF - Hides trace statistics

**Examples**

```
CALC:MEAS2:FUNC:STAT ON
calculate2:measure2:function:statistics:state off
```

**Query Syntax**

`CALCulate<cnum>:MEASure<mnum>:FUNCtion:STATistics[:STATe]?`

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF (0)

---

**CALCulate<cnum>:MEASure<mnum>:FUNCtion:TYPE <char>**

Applicable Models: All

(Read-Write) Sets statistic TYPE that you can then query using `CALC:MEAS:FUNCtion:DATA?`.

**Note:** This command affects only the selected measurement on the specified channel.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<char>` Choose from:
  - `PTPeak` - the difference between the max and min data points on the trace.
  - `STDEV` - standard deviation of all data points on the trace
  - `MEAN` - mean (average) of all data points on the trace
  - `MIN` - lowest data point on the trace
  - `MAX` - highest data point on the trace
<table>
<thead>
<tr>
<th>Examples</th>
<th>CALC:MEAS2:FUNC:TYPE PTP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>calculate2:measure2:function:type stdev</td>
</tr>
<tr>
<td>Query Syntax</td>
<td>CALCulate&lt;cnum&gt;:MEASure&lt;mnum&gt;:FUNCtion:TYPE?</td>
</tr>
<tr>
<td>Return Type</td>
<td>Character</td>
</tr>
<tr>
<td>Default</td>
<td>PTPeak</td>
</tr>
</tbody>
</table>
CALCulate:MEASure:GCData Commands

Reads Gain Compression data from the current Gain Compression acquisition.

```
CALCulate:MEASure:GCData
  DATA
  IMAG
  ITERations
  REAL
```

Click a keyword to view the command details.

Other Gain Compression commands

The calibration commands listed in this topic are supplemental to the Guided Cal commands.

- **CALC:MEAS:DEFINE** - creates a gain compression measurement.
- **SENS:GCSetup** - Most Gain Compression settings.
- **CALC:MEAS:GCMes:ANAL** - Gain Compression Analysis settings
- Gain compression data can also be saved to a *.csv file. Learn how.

See Also

- Learn about Gain Compression Application
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

CALCulate<ch>:MEASure<mnum>:GCData:DATA? <param>

**Applicable Models:** N522xB, N524xB, M9485A

*(Read-Only)* Returns measurement data at all frequency and power data points for GCA SMART sweeps and 2D sweeps.

- When using SMART sweep, ALL data is returned including ALL background iteration sweeps. Use **CALC:MEAS:GCD:ITER** to determine the number of iteration sweeps. The number of data points that are returned is always going to be number of frequency points times the number of iteration sweeps.
- When using 2D sweeps, ALL data is returned. The number of data points returned / freq may vary. Learn
more.

Use CALC:MEAS:DATA? to return just the displayed data results (not the background sweeps).

A compression parameter must be present. Learn more.

The format of the data is the same as the format of the measurement that you select using CALC:MEAS:PAR. If the measurement is scalar, than one number is returned per sweep per data point. If complex (such as Smith Chart format) than both real and imaginary numbers are returned.

If correction is on, corrected data are returned. Otherwise, raw data are returned.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<param>` (String) Parameter to read. NOT Case-sensitive. The specified parameter need NOT be displayed or selected. However, a compression parameter must be present. Learn more.

Choose from:

- "pin" - (CompIn21) Input power at the compression point.
- "pout" - (CompOut21) Output power at the compression point.
- "gain" - (CompGain21) Device gain (S21) at the compression point.
- "inputmatch" - (CompS11) Input match at the compression point.
- "DeltaGain" - (DeltaGain21) Measured Gain (watts) / Ref Gain (watts). Learn more.
- "AI1" and "AI2" - ADC measurements at the specified compression level. Learn more.

Learn more about GCA parameters.

Examples

```plaintext
data = CALC:MEAS2:GCD:DATA? "pin"
data = calculate:measure2:gcdata:data? "pout"
```

Return Type

Array of data

Default Not Applicable

CALCulate<ch>:MEASure<mnum>:GCData:IMAG? <char>, <dpoint>, <param>

Applicable Models: N522xB, N524xB, M9485A
(Read-Only) For a specified data point, returns the imaginary part of the specified Gain Compression data. If correction is on, corrected data are returned. Otherwise, raw data are returned. Can be used with Smart and 2D sweeps.

- For SMART sweep, the number of data points that are returned is always going to be the number of iteration sweeps. Use `CALC:MEAS:GCD:ITER` to determine the number of iteration sweeps.
- For 2D sweeps, the number of data points returned / freq may vary. Learn more.

**Parameters**

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<char>` Choose from:
  - **FREQuency** - for the specified frequency data point, returns all of the measured data for each power stimulus.
  - **POWer** - for the specified power data point, returns all of the measured data for each frequency stimulus.
- `<dPoint>` Data point (FREQ or POWer) for which data is returned.
- `<param>` (String) Parameter to read. NOT Case-sensitive. The specified parameter need NOT be displayed. However, a compression parameter must be present. Learn more.
  - "pin" - (CompIn21) Input power at the compression point.
  - "pout" - (CompOut21) Output power at the compression point.
  - "gain" - (CompGain21) Device gain (S21) at the compression point.
  - "inputmatch" - (CompS11) Input match at the compression point.
  - "DeltaGain" - (DeltaGain21) Measured Gain (watts) / Ref Gain (watts). Learn more.
  - "AI1" and "AI2" - ADC measurements at the specified compression level. Learn more.

**Examples**

For the fifth frequency data point, returns 'Power Output' imaginary (phase) data from all power stimulus values.

For SmartSweep, if there are 30 power sweep points, 30 values are returned.

For 2D sweeps, 30 or 31 power sweep points may be returned. Learn more.
CALCulate<cnum>:MEASure<mnum>:GCData:ITERations?

Applicable Models: N522xB, N524xB, M9485A

(Read-only) In a SMART sweep, returns the max number of iterations that it took for ALL frequencies to converge. Use this number to determine the size of the block data that is returned from Gain Compression SMART sweep data queries.

For a 2D sweep, returns the number of power points.

Parameters

- `<cnum>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.

Examples

```plaintext
data = CALC:MEAS2:GCD:IMAG? FREQ,5,"pout"
```

Return Type

- Array of data
- Default: Not Applicable

CALCulate<ch>:MEASure<mnum>:GCData:REAL? <char>, <dpoint>, <param>

Applicable Models: N522xB, N524xB, M9485A

(Read-Only) For a specified data point, returns the real part of the Gain Compression data. If correction is on, corrected data are returned. Otherwise, raw data are returned. Can be used with Smart and 2D sweeps.

- For SMART sweep, the number of data points that are returned is always going to be the number of iteration sweeps. Use `CALC:MEAS:GCD:ITER` to determine the number of iteration sweeps.

- For 2D sweeps, the number of data points returned / freq may vary. Learn more.

Parameters

- `<ch>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
- `<char>`: Choose from:
- **FREQuency** - for the specified frequency data point, returns all of the measured data for each power stimulus.
- **POWer** - for the specified power data point, returns all of the measured data for each frequency stimulus.

<dPoint> Data point (FREQ or POWer) for which data is returned.

<param> (String) Parameter to read. NOT Case-sensitive. The specified parameter need NOT be displayed. However, a compression parameter must be present. Learn more.

- "pin" - (CompIn21) Input power at the compression point.
- "pout" - (CompOut21) Output power at the compression point.
- "gain" - (CompGain21) Device gain (S21) at the compression point.
- "inputmatch" - (CompS11) Input match at the compression point.
- "DeltaGain" - (DeltaGain21) Measured Gain (watts) / Ref Gain (watts). Learn more.
- "AI1" and "AI2" - ADC measurements at the specified compression level. Learn more.

**Examples**
For the fifth frequency data point, returns 'Power Output' real data from all power stimulus values.
For SmartSweep, if there are 30 power sweep points, 30 values are returned.
For 2D sweeps, 30 or 31 power sweep points may be returned. Learn more.

```
data = CALC:MEAS2:GCD:REAL? FREQ,5,"pout"
```

**Return Type** Array of data

**Default** Not Applicable
CALCulate:MEASure:GCMeas Commands

Sets and reads Gain Compression Analysis controls.

<table>
<thead>
<tr>
<th>CALCulate:MEASure:GCMeas</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANALysis</td>
</tr>
<tr>
<td>CWFrequency</td>
</tr>
<tr>
<td>DIScrete</td>
</tr>
<tr>
<td>[:STATE]</td>
</tr>
<tr>
<td>ENABLe</td>
</tr>
<tr>
<td>XAXis</td>
</tr>
</tbody>
</table>

Click a keyword to view the command details.

Other Gain Compression commands

The calibration commands listed in this topic are supplemental to the Guided Cal commands.

- CALC:MEAS:DEFine - creates a gain compression measurement.
- SENS:GCSetup - Most Gain Compression settings.
- CALC:MEAS:GCDATA - Gain Compression data commands
- Gain compression data can also be saved to a *.csv file. Learn how.

See Also

- Learn about Gain Compression Application
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

CALCulate<cnum>:MEASure<mnum>:GCMeas:ANALysis:CWFRequency <num>
Applicable Models: N522xB, N524xB, M9485A

(Read-Write) Set and return the CW frequency for a compression analysis trace.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` CW frequency in Hz. Choose a frequency within the range of the gain compression channel.

Examples

```
CALC:MEAS2:GCM:ANAL:CWFR 1e9
```
```
calculate2:measure2:gcmeas:analysis:cwfrequency 1e10
```

Query Syntax

CALCulate<cnum>:MEASure<mnum>:GCMeas:ANALysis:CWFRequency?

Return Type Numeric

Default Not Applicable

CALCulate<cnum>:MEASure<mnum>:GCMeas:ANALysis:DISCRete[:STATe] <bool>

Applicable Models: N522xB, N524xB, M9485A

(Read-Write) Sets and returns whether the CW frequency for the compression analysis trace can be set to only the discrete frequencies or provides interpolation.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` ON (or 1) - Discrete data points only.
- OFF (or 0) - Interpolated data points.

Examples

```
CALC:MEAS2:GCM:ANAL:ISD ON
```
```
calculate2:measure2:gcmeas:analysis:isdisfrequency off
```

Query Syntax

CALCulate<cnum>:MEASure<mnum>:GCMeas:ANALysis:ISDisfrequency?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

CALCulate<cnum>:MEASure<mnum>:GCMeas:ANALysis:ENABle <bool>
**Applicable Models:** N522xB, N524xB, M9485A

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>`
  - **ON** (or 1) - Enable compression analysis.
  - **OFF** (or 0) - Disable compression analysis.

**Examples**

```
calc:meas2:gcmeas:analysis:enable on
calculate2:measure2:gcmeas:analysis:enable off
```

**Query Syntax**

```
calc:meas<mnum>:gcmeas:analysis:enable?
```

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

---

**CALCulate<cnum>:MEASure<mnum>:GCMeas:ANALysis:XAXis <char>**

**Applicable Models:** N522xB, N524xB, M9485A

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>`
  - PIN - Input power to the DUT.
  - PSOsource - power from the source.

**Examples**

```
calc:meas2:gcmeas:analysis:xaxis pin
calculate2:measure2:gcmeas:analysis:xaxis psource
```

**Query Syntax**

```
calc:meas<mnum>:gcmeas:analysis:xaxis?
```

**Return Type**

Character

**Default**

PIN
CALCulate:MEASure:GDELay Commands

Controls the Aperture setting used to make Group Delay measurements.

<table>
<thead>
<tr>
<th>CALCulate:MEASure:GDELay</th>
</tr>
</thead>
<tbody>
<tr>
<td>FREQuency</td>
</tr>
<tr>
<td>PERCent</td>
</tr>
<tr>
<td>POINts</td>
</tr>
</tbody>
</table>

Click a keyword to view the command details.

see Also

- Learn about Group Delay Aperture
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

CALCulate:<cnum>:MEASure<mnum>:GDELay:FREQuency <value>

Applicable Models: All

(Read-Write) Sets group delay aperture using a fixed frequency range.

Parameters

- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <value> Frequency range (in Hz) to use for the aperture setting. Choose between the equivalent of two data points and the channel frequency span.

Examples

CALC:MEAS2:GDEL:FREQ 1E6

Query Syntax

CALCulate<cnum>:MEASure<mnum>:GDELay:FREQuency?

Return Type

Numeric

Default

Frequency range that equates to 11 points. This can be changed to two points with a preference setting.

CALCulate:<cnum>:MEASure<mnum>:GDELay:PERCent <value>
Applicable Models: All

(Read-Write) Sets group delay aperture using a percent of the channel frequency span.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<value>` Percent of frequency span to use for the aperture setting. Choose between the equivalent of two data points and 100 percent of the channel frequency span.

**Examples**

- 'set to 25 percent of the channel frequency span
  
  `CALC:MEAS2:GDEL:PERC 25`

**Query Syntax**

`CALCulate<cnum>:MEASure<mnum>:GDELay:PERCent?`

**Return Type**

Numeric

**Default**

Percent of frequency span that equates to 11 points. This can be changed to two points with a preference setting.

---

CALCulate<cnum>:MEASure<mnum>:GDELay:POINts <value>

Applicable Models: All

(Read-Write) Sets group delay aperture using a fixed number of data points.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<value>` Number of data points to use for the aperture setting. Choose between two points and the number of points in the channel.

**Examples**

- 'set to 25 data points
  
  `CALC:MEAS2:GDEL:POIN 25`

**Query Syntax**

`CALCulate<cnum>:MEASure<mnum>:GDELay:POINts?`

**Return Type**

Numeric

**Default**

11 points. This can be changed to two points with a preference setting.
CALCulate:MEASure:LIMit Commands

Controls the limit segments used for pass / fail testing.

<table>
<thead>
<tr>
<th>CALCulate:MEASure:LIMit</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>DISPlay</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>FAIL?</td>
</tr>
<tr>
<td>REPORT</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>SEGment</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>SOUNd</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>[:STATe]</td>
</tr>
</tbody>
</table>

Click a keyword to view the command details.

see Also
CALCulate\(<\text{cnum}\>\):MEASure\(<\text{mnum}\>\):LIMit:DATA \(<\text{block}\>

**Applicable Models:** All

(Read-Write) Sets data for limit segments.

**Parameters**

\(<\text{cnum}\>\) Channel number of the measurement (optional).

\(<\text{mnum}\>\) Measurement number for each measurement.

\(<\text{block}\>\) Data for all limit segments. The following is the data format for 1 segment:

\(\text{Type, BegStim, EndStim, BegResp, EndResp}\)

- **Type** Type of limit segment. Choose from
  - 0 - Off
  - 1 - Max
  - 2 - Min

- **BegStim** Start of X-axis value (freq, power, time)

- **EndStim** End of X-axis value

- **BegResp** Y-axis value that corresponds with Start of X-axis value

- **EndResp** Y-axis value that corresponds with End of X-axis value

**Examples**

The following writes three max limit segments for a bandpass filter.

```
CALC:MEAS2:LIM:DATA 1,3e5,4e9,60,0,1,4e9,7.5e9,0,0,1,7.5e9,9e9,0,-30
```

**Query Syntax**

CALCulate\(<\text{cnum}\>\):MEASure\(<\text{mnum}\>\>:LIMit:DATA?

**Return Type**

Depends on FORM:DATA

**Default**

Maximum limit = 100

CALCulate\(<\text{cnum}\>\):MEASure\(<\text{mnum}\>\>:LIMit:DATA:DELeTe
Applicable Models: All

(Write-only) Deletes all limit line data for the selected measurement on the specified channel.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC2:MEAS2:LIM:DATA:DEL
```

**Query Syntax** Not Applicable

**Default** Not Applicable

---

CALCulate<cnum>:MEASure<mnum>:LIMit:DISPlay[:STATe] <ON | OFF>

Applicable Models: All

(Read-Write) Turns the display of limit segments ON or OFF (if the data trace is turned ON).

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<ON | OFF>`
  - ON (or 1) - turns the display of limit segments ON.
  - OFF (or 0) - turns the display of limit segments OFF.

**Examples**

```
CALC:MEAS2:LIM:DISP:STAT ON
calculate2:limit:display:state off
```

**Query Syntax** CALCulate<cnum>:MEASure<mnum>:LIMit:DISPlay[:STATe]?

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** ON

---

CALCulate<cnum>:MEASure<mnum>:LIMit:FAIL?
Applicable Models: All

(Read-only) Returns the Pass / Fail status of the limit line test. Returns 1 (Fail) if any data point fails for any limit segment.

Limit display (CALC:LIM:DISP) does NOT have to be ON.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

Examples

CALC:MEAS2:LIM:FAIL?

Return Type: Boolean

- 0 is returned when Pass
- 1 is returned when Fail

Default: Not Applicable

CALCulate<cnum>:MEASure<mnum>:LIMit:REPort:ALL? <block>

Applicable Models: All

(Read-only) Returns the test results (stimulus value, limit test result, upper limit value and lower limit value of all measurement points), for the selected trace. This command returns a point by point description of the limit table and pass/fail test result.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<block>` Depends on FORM:DATA

If the number of the measurement points is N,

- `<Block> = <first stimulus>,<test result>,<upper limit>,<lower limit>, ..., <Nth stimulus>,<test result>,<upper limit>,<lower limit>

Where `<test result>` = -1: No limit, 0:Fail, 1:Pass

The following example returns three points:

CALC:MEAS2:LIM:REP:ALL?

+1.00000000000E+009, 'bucket 1 XAxis value
+1.00000000000E+000, 'result (1 == Pass)
Examples

-4.90000009537E+000, 'upper limit
-5.05000019073E+000, 'lower limit
+3.00000000000E+09, 'bucket 2 XAxis value
+1.00000000000E+000,
-4.8499999463E+000,
-5.1999980927E+000,
+5.00000000000E+09, 'bucket 3 Xaxis value
-1.00000000000E+000,
+0.00000000000E+000,
+0.00000000000E+000

Return Type

Variant

Default Depend on the preset status

CALCulate<cnum>:MEASure<mnum>:LIMit:REPort[:DATA]? <block>

Applicable Models: All

(Read-only) Returns the stimulus values (frequency, power level or time) at all the measurement points that failed the limit test, for the selected trace. If there are no failures, a large number is returned (+9.91000000000E+037).

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<block>` Depends on FORM:DATA

If the number of the measurement points that failed the limit test is N, `<block>=<First failed stimulus>, ..., <Nth failed stimulus>.

The following example assumes that there are no failures:

Examples

CALC:MEAS2:LIM:REP:DATA?

+9.91000000000E+037

Return Type

Numeric

Default 9.91E37 (no failures)
CALCulate\(<cnum>\):MEASure\(<mnum>\):LIMit:REPort:POINts?

Applicable Models: All

(Read-only) Reads the number of the measurement points that failed the limit test, for the active trace of selected channel.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>(&lt;cnum&gt;)</td>
</tr>
<tr>
<td>Channel number of the measurement (optional).</td>
</tr>
<tr>
<td>(&lt;mnum&gt;)</td>
</tr>
<tr>
<td>Measurement number for each measurement.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS:LIM:REP:POIN?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
</tr>
<tr>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
</tr>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

CALCulate\(<cnum>\):MEASure\(<mnum>\):LIMit:SEGment\(<snum>\):AMPLitude:STARt \(<num>\)

Applicable Models: All

(Read-Write) Sets the start (beginning) of the Y-axis amplitude (response) value.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>(&lt;cnum&gt;)</td>
</tr>
<tr>
<td>Channel number of the measurement (optional).</td>
</tr>
<tr>
<td>(&lt;mnum&gt;)</td>
</tr>
<tr>
<td>Measurement number for each measurement.</td>
</tr>
<tr>
<td>(&lt;snum&gt;)</td>
</tr>
<tr>
<td>Segment number; if unspecified, value is set to 1.</td>
</tr>
<tr>
<td>(&lt;num&gt;)</td>
</tr>
<tr>
<td>Choose any number between: -500 and 500</td>
</tr>
</tbody>
</table>

Display value is limited to the Maximum and Minimum displayed Y-axis values.

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS2LIM:SEG1:AMPL:STAR 10</td>
</tr>
<tr>
<td>calculate2:measure2:limit:segment2:amplitude:start 10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate(&lt;cnum&gt;):MEASure(&lt;mnum&gt;):LIMit:SEGment(&lt;snum&gt;):AMPLitude:STARt?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numeric</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

CALCulate\(<cnum>\):MEASure\(<mnum>\):LIMit:SEGment\(<snum>\):AMPLitude:STOP \(<num>\)
Applicable Models: All

(Read-Write) Sets the stop (end) of the Y-axis amplitude (response) value.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<snum>` Segment number; if unspecified, value is set to 1.
- `<num>` Choose any number between: -500 and 500

Display value is limited to the Maximum and Minimum displayed Y-axis values.

Examples

- `calculate2:measure2:limit:segment2:amplitude:stop 10`

Query Syntax

- `CALCulate<cnum>:MEASure<mnum>:LIMit:SEGMent<snum>AMPLitude:STOP?`

Return Type

- Numeric
- Default: 0

CALCulate:MEASure<mnum>:LIMit:SEGMent:COUNt?

Applicable Models: All

(Read-only) Returns the number of segments used in a limit test. All segments are counted, whether they are on or not.

Parameters

- `<mnum>` Measurement number for each measurement.

Examples

- `CALC:MEAS2:LIM:SEGM:COUN?`

Return Type

- Numeric
- Default: Not Applicable
Applicable Models: All

(Read-Write) Sets the start (beginning) of the X-axis stimulus value.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<snum>` Segment number; if unspecified, value is set to 1.
- `<num>` Choose any number within the X-axis span of the analyzer.

**Examples**

```
CALC:MEAS:LIM:SEG1:STIM:STAR 10
```

```
calculate2:measure:limit:segment2:stimulus:start 10
```

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:LIMit:SEGment<snum>:STIMulus:STARt?

**Return Type**

Numeric

**Default**

0

CALCulate<cnum>:MEASure<mnum>:LIMit:SEGment<snum>:STIMulus:STOP <num>

Applicable Models: All

(Read-Write) Sets the stop (end) of the X-axis stimulus value.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<snum>` Segment number; if unspecified, value is set to 1.
- `<num>` Choose any number within the X-axis span of the analyzer.

**Examples**

```
CALC:MEAS2:LIM:SEG2:AMPL:STOP 10
```

```
calculate2:measure:limit:segment2:stimulus:stop 10
```

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:LIMit:SEGment<snum>:STIMulus:STOP?

**Return Type**

Numeric

**Default**

0

CALCulate<cnum>:MEASure<mnum>:LIMit:SEGment<snum>:TYPE <char>
Applicable Models: All

(Read-Write) Sets the type of limit segment.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<snum>` Segment number. Choose any number between: 1 and 100
  If unspecified, value is set to 1.
- `<char>` Choose from:
  - **LMAX** - a MAX limit segment. Any response data exceeding the MAX value will fail.
  - **LMIN** - a MIN limit segment. Any response data below the MIN value will fail.
  - **OFF** - the limit segment (display and testing) is turned OFF.

**Examples**

```plaintext
CALC:MEAS2:LIM:SEGM:TYPE LMIN
calculate2:measure2:limit:segment3:type lmax
```

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:LIMit:SEGMent<snum>:TYPE?

**Return Type** Character

**Default** OFF

---

CALCulate<cnum>:MEASure<mnum>:LIMit:SOUNd[:STATe] <ON | OFF>

Applicable Models: All

(Read-Write) Turns limit testing fail sound ON or OFF.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<ON | OFF>`
  - **ON** (or 1) - turns sound ON.
  - **OFF** (or 0) - turns sound OFF.

**Examples**

```plaintext
CALC:MEAS2:LIM:SOUN ON
calculate2:measure2:limit:sound:state off
```

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:LIMit:SOUNd[:STATe]?
CALCulate<cnum>:MEASure<mnum>:LIMit[:STATe] <ON | OFF>

**Applicable Models:** All

(Read-Write) Turns limit segment **testing** ON or OFF.

- Use `CALCulate:MEASure:LIMit:DISPLAY` to turn ON and OFF the **display** of limit segments.
- If using **Global Pass/Fail** status, trigger the VNA AFTER turning Limit testing ON.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<ON | OFF>` **ON** (or 1) - turns limit testing ON.
  **OFF** (or 0) - turns limit testing OFF.

**Examples**
- `CALC:MEAS:LIM:STAT ON`
- `calculate2:measure:limit:state off`

**Query Syntax**
- `CALCulate<cnum>:MEASure<mnum>:LIMit:STATe?`

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** OFF
**CALCulate:MEASure:MARKer Commands**

Controls the marker settings used to remotely output specific data to the computer.

<table>
<thead>
<tr>
<th>CALCulate:MEASure:MARKer:</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOFF</td>
</tr>
<tr>
<td>BUCKet</td>
</tr>
<tr>
<td>BWIDth</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>COUPling</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>DELTa</td>
</tr>
<tr>
<td>DISCrete</td>
</tr>
<tr>
<td>DISTance</td>
</tr>
<tr>
<td>FORMat</td>
</tr>
<tr>
<td>FUNCtion</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
| STARt
| STOP
| EXECute
| MULTI
| EXECute
| PEAK
| EXCursion
| POLarity
| THReshold
| SELect
| TARGET
| TRANSition
| [:VALue]
| TRACKing
| PEAK
| EXCursion
| POLarity
| THReshold
| [:SELect]
| TARGET
| TRANSition
| [:VALue]
| TRACKing

NOTCh
| DATA?
| REF
| [:STATE]
| THReshold
PNOP
  | BACKoff
    | GAIN
    | PIN
    | POUT
  | COMpression
    | MAXimum
  | GAIN
    | MAXimum
  | PIN
    | MAXimum
  | POFFset
  | POUT
    | MAXimum
  | [:STATe]
PSATuration
  | BACKoff
  | COMpression
    | MAXimum
    | SATuration
  | GAIN
    | LINear
    | MAXimum
  | PIN
    | MAXimum
  | POUT
    | MAXimum
  | [:STATe]
Click a keyword to view the command details.

See Also

- Marker Readout number and size commands.
- Learn about Markers
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**CALCulate<cnum>:MEASure<mnum>:MARKer:AOFF**

**Applicable Models:** All

**(Write-only)** Turns all markers off for selected measurement.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS2:MARK:AOFF
calculate2:measure2:marker:aoff
```

**Query Syntax** Not applicable

**Default** Not applicable

**CALCulate<cnum>:MEASure<mnum>:MARKer<n>:BUCKet <num>**
Applicable Models: All

*(Read-Write)* Sets and reads the data point (bucket) number of the trace on which the marker resides. When the markers are interpolated (non-discrete), the returned value is the nearest marker bucket position.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number to move or query. The marker must already exist. If unspecified, `<n>` is set to 1.
- `<num>` Data point (bucket) number. Choose any data point between: 0 and the number of data points minus 1.

**Examples**

```
CALC:MEAS:MARK:BUCK 5
```

```
calculate2:measure2:marker2:bucket 200
```

**Query Syntax**

```
CALCulate<cnum>:MEASure<mnum>:MARKer<n>:BUCKet?
```

**Return Type** Integer

**Default** The first marker is set to the middle of the span. Subsequent markers are set to the bucket number of the previously active marker.

---

**CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:BWIDth:DATA?**

Applicable Models: All

*(Read-only)* Read the bandwidth search result of marker 1 to 15 and reference marker (Mkr :16), for the active trace of selected channel.

If the bandwidth search is impossible, an error occurs when executed and the object is ignored.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; If unspecified, value is set to 1.
  
  Four Character values separated by commas => `{numeric 1}, {numeric 2}, {numeric 3}, {numeric 4}`

  - `{numeric 1}` : Bandwidth
  - `{numeric 2}` : Center point frequency of the 2 cutoff frequency points
Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>calculate2:measure1:marker:bwidth:data?</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:BWIDth:REF <string>

Applicable Models: All

(Read-Write) Set the bandwidth marker function reference to either MARKer or PEAK.

If the reference is set to MARKer, the active marker is not moved; the bandwidth search is computed at the marker's current location.

If the reference is PEAK, the active marker is moved to the maximum or minimum peak on the trace and then bandwidth search is computed.

- If the bandwidth level is negative, the active marker is moved to the maximum peak.
- If the bandwidth level is positive, the active marker is moved to the minimum peak.

Parameters

- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <mkr> Any marker number from 1 to 15; if unspecified, value is set to 1.
- <string> PEAK

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS:MARK:BWID:REF MARK</td>
<td>String</td>
<td>MARKer</td>
</tr>
<tr>
<td>calculate2:measure1:marker:bwidth:ref peak</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:BWIDth[:STATe] <bool>
Applicable Models: All

(Read-Write) Turns ON or OFF the bandwidth search result display, for the active trace of selected channel.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<bool>` Bandwidth search result display:
  - **ON or 1** - Turns ON the bandwidth search result display.
  - **OFF or 0** - Turns OFF the bandwidth search result display.

**Examples**
- CALC:MEAS:MARK:BWID ON
- calculate2:measure1:marker:bwidth:state off

**Query Syntax**
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:BWIDth[:STATe]?

**Return Type**
Boolean

Default
OFF or 0

**CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:BWIDth:THReshold <value><unit>**

Applicable Models: All

(Read-Write) Sets or returns the bandwidth definition value (the value to define the pass-band of the filter) of marker 1 to 15 and reference marker (Mk :16), for the active trace of selected channel.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<value>` Bandwidth definition value (the value to define the pass band of the filter) is between -5E8 to 5E8.
- `<unit>` Varies depending on the data format.
  - Log magnitude (MLOG): dB (decibel)
  - Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): deg (degree)
  - Group delay (GDEL): s (second)
  - Others: No unit

**Examples**
- CALC:MEAS:MARK:BWID:THR -3
- calculate2:measure1:marker:bwidth:threshold -3
**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:BWIDth:THReshold?

**Return Type**

Numeric

Default -3

---

**CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:COUPling:METHod <char>**

**Applicable Models:** All

(Read-Write) Sets and reads the scope of Coupled Markers. This is a global setting that affects all markers. Learn more.

**Note:** This command will not take effect until Coupled Markers is turned on using
CALC:MEAS:MARK:COUP:STATe ON.

**Note:** The preset behavior of Coupled Markers depends on the setting of
SYSTem:PREFerences:ITEM:MCControl, SYSTem:PREFerences:ITEM:MCMethod, and
SYSTem:PREFerences:ITEM:MCPRest.

**Note:** If any or all <cnum>, <mnum>, or <mkr> arguments are omitted, they are assumed to have the value 1.

**Parameters**

- **<cnum>** Must be a valid channel number (unless a measurement number is provided), but marker coupling is not set per channel.
- **<mnum>** Must be a valid measurement number and must be displayed on the screen. Marker coupling is not set per measurement.
- **<mkr>** Not used. The marker number must still be in the range of 1-16, but marker coupling is not set per marker.
- **<char>** CHANnel - Coupling is limited to traces in the same channel.
  - ALL - Coupling occurs across all channels.

**Examples**

CALC:MEAS:MARK:COUP:METH CHAN

calculate:measure:marker:coupling all

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:MARKer:COUPling:METHod?

**Return Type**

Character

Default ALL
CALCulate:MEASure<mnum>:MARKer<mkr>:COUPling[:STATe]<ON|OFF>

**Applicable Models:** All

*(Read-Write)* Sets and reads the state of Coupled Markers (ON and OFF). The scope of coupled markers can be changed with CALC:MEAS:MARK:COUP:METH.

**Note:** If the `<mnum>` or `<mkr>` argument is omitted, they are assumed to have the value 1.

**Parameters**

- `<mnum>`: Must be a valid measurement number and must be displayed on the screen.
- `<mkr>`: Not used. The marker number must still be in the range of 1-16.
- `<ON|OFF>`
  - OFF (0): Turns Coupled Markers OFF
  - ON (1): Turns Coupled Markers ON

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS:MARK:COUP ON</td>
<td>Turns Coupled Markers ON</td>
</tr>
<tr>
<td>calculate:measure1:marker:coupling off</td>
<td>Turns Coupled Markers OFF</td>
</tr>
</tbody>
</table>

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:COUPling[:STATe]?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

---

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:DELTa <ON|OFF>

**Applicable Models:** All

*(Read-Write)* Specifies whether marker is relative to the Reference marker or absolute.

**Note:** The reference marker must already be turned ON with CALC:MARK:REF:STATE.

**Parameters**

- `<cnum>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
- `<mkr>`: Any existing marker number from 1 to 15; if unspecified, value is set to 1.
- `<ON|OFF>`
  - ON (1): Specified marker is a Delta marker
  - OFF (0): Specified marker is an ABSOLUTE marker

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS:MARK:DELT ON</td>
<td>Specified marker is a Delta marker</td>
</tr>
<tr>
<td>calculate2:measure1:marker8:delta off</td>
<td>Specified marker is an ABSOLUTE marker</td>
</tr>
</tbody>
</table>

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:DELTa?

**Return Type**

Boolean (1 = ON, 0 = OFF)
**CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:DISCrete <ON|OFF>**

**Applicable Models:** All

*(Read-Write)* Makes the specified marker display either a calculated value between data points (interpolated data) or the actual data points (discrete data).

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any existing marker number from 1 to 15; if unspecified, value is set to 1.
- `<ON|OFF>`
  - **ON** (or 1) - Specified marker displays the actual data points
  - **OFF** (or 0) - Specified marker displays calculated data between the actual data points.

**Examples**

```
CALC:MEAS:MARK:DISC ON
calculate2:measure2:marker8:discrete off
```

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:DISCrete?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

---

**CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:DISTance <num>**

**Applicable Models:** All

*(Read-Write)* Set or query marker distance on a time domain trace.

The Write command moves the marker to the specified distance value. Once moved, you can read the Y axis value or read the X-axis time value. (Distance is calculated from the X-axis time value.)

The Read command reads the distance of the marker.

If the marker is set as delta, the WRITE and READ data is relative to the reference marker.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any existing marker number from 1 to 15; if unspecified, value is set to 1.
Marker distance in the unit of measure specified with CALC:TRAN:TIME:MARK:UNIT

**Examples**

```
CALC:MEAS:MARK:DIST .1
calculate2:measure1:marker8:distance 5
```

**Query Syntax**

CALCulate\(<cnum>\):MEASure\(<mnum>\):MARKer\(<mkr>\):DISTance?

**Return Type**

Numeric

**Default**

Not Applicable

---

**CALCulate\(<cnum>\):MEASure\(<mnum>\):MARKer\(<mkr>\):FORMat \(<char>\)**

**Applicable Models:** All

(Read-Write) Sets the format of the data that will be returned in a marker data query CALC:MARK:Y? and the displayed value of the marker readout. The selection does not have to be the same as the measurement's display format.

**Parameters**

\(<cnum>\)  Channel number of the measurement (optional).

\(<mnum>\)  Measurement number for each measurement.

\(<mkr>\)  Any marker number from 1 to 15; if unspecified, value is set to 1

\(<char>\)  Choose from:

- **DEFAULT** - The format of the selected measurement
- **MLINear** - Linear magnitude
- **MLOGarithmic** - Logarithmic magnitude
- **IMPedance** - (R+jX)
- **ADMittance** - (G+jB)
- **PHASe** - Phase
- **IMAGinary** - Imaginary part (Im)
- **REAL** - Real part (Re)
- **POLar** - (Re, Im)
- **GDElay** - Group Delay
- **LINPhase** - Linear Magnitude and Phase
**LOGPhase** - Log Magnitude and Phase

**KELVIn** - temperature

**FAHRENheit** - temperature

**CELSius** - temperature

**NOISE** - Noise (available ONLY in IM Spectrum and SA measurement classes).

**Examples**

```
CALC:MEAS:MARK:FORMat MLIN
calculate2:measure:marker8:format Character
```

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:APEak:POLarity?

**Return Type**

Character

**Default**

DEFault

---

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:APEak:POLarity <char>

**Applicable Models:** All

(Read-Write) Sets or returns polarity of the peak search with marker 1 to 15 and reference marker (Mk:16), for the active trace of selected channel.

Learn more about Marker Search

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<char>` Polarity for peak search function to be performed. Choose from:
  
  - "NEGative": Specifies the negative peak.
  - "POSitive": Specifies the positive peak.
  - "BOTH": Specifies both the positive peak and the negative peak.

**Examples**

```
CALC:MEAS:MARK:FUNC:APE:POL NEG
calculate2:measure1:marker6:function:apeak:polarity both
```

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:APEak:POLarity?
CALCulate\textsubscript{<cnum>}:MEASure\textsubscript{<mnum>}:MARKer\textsubscript{<mkr>}:FUNCtion:COMPres\textsubscript{ion}:LEVel \textsubscript{<num>}

**Applicable Models:** All

*(Read-Write)* Sets and read the marker compression level. A compression marker must already exist. Use CALC:MARK ON and CALC:MEAS:MARK:FUNC COMP to create compression markers.

**Parameters**
- \textsubscript{<cnum>} Channel number of the measurement (optional).
- \textsubscript{<mnum>} Measurement number for each measurement.
- \textsubscript{<mkr>} Any existing marker number from 1 to 15; if unspecified, value is set to 1.
- \textsubscript{<num>} Compression level. Choose any number between: -500 dB to 500 dB

Standard gain compression values are positive.

**Examples**
- `CALC:MEAS:MARK:FUNC:COMP:LEV 1`
- `calculate2:measure1:marker:function:compression:level 1.5`

**Query Syntax**
- `CALCulate\textsubscript{<cnum>}:MEASure\textsubscript{<mnum>}:MARKer\textsubscript{<mkr>}:FUNCtion:COMPres\textsubscript{ion}:LEVel?`

**Return Type**
- Numeric
- Default: +1 dB

---

CALCulate\textsubscript{<cnum>}:MEASure\textsubscript{<mnum>}:MARKer\textsubscript{<mkr>}:FUNCtion:COMPres\textsubscript{ion}:PIN?

**Applicable Models:** All

*(Read-only)* Read the input power at the marker compression level. First send CALC:MEAS:MARK:FUNC:EXEC COMP or CALC:MEAS:MARK:FUNC:TRAC ON.

**Parameters**
- \textsubscript{<cnum>} Channel number of the measurement (optional).
- \textsubscript{<mnum>} Measurement number for each measurement.
- \textsubscript{<mkr>} Any existing marker number from 1 to 15; if unspecified, value is set to 1.

**Examples**
- `CALC:MEAS:MARK:FUNC:COMP:PIN?`
- `calculate2:measure1:marker:function:compression:pin?`

**Return Type**
- Numeric
- Default: Not Applicable
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:COMPression:POUT?

Applicable Models: All

(Read-only) Read the output power at the marker compression level. First send CALC:MEAS:MARK:FUNC:EXEC COMP or CALC:MEAS:MARK:FUNC:TRAC ON

Parameters
- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <mkr> Any existing marker number from 1 to 15; if unspecified, value is set to 1.

Examples
- CALC:MEAS:MARK:FUNC:COMP:POUT?
- calculate2:measure1:marker:function:compression:pout?

Return Type Numeric
Default Not Applicable

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:COMPression[:STATe] <bool>

Applicable Models: All

(Read-Write) Turns ON or OFF the compression state.

Parameters
- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <mkr> Any marker number from 1 to 15; if unspecified, value is set to 1.
- <bool> Bandwidth search result display:

  **ON or 1** - Turns ON the compression.

  **OFF or 0** - Turns OFF the compression.

Examples
- CALC:MEAS:MARK:FUNC:COMP:STAT ON
- calculate2:measure1:marker:function:compression:state off

Query Syntax
CALCulate<cnum>:MEASure<mnum>:MARKer:FUNCtion:COMPression[:STATe]?

Return Type Boolean
Default OFF or 0

Applicable Models: All

(Read-Write) Assigns the specified marker to a range number. The x-axis travel of the marker is constrained to the range's span. The span is specified with the CALC:MEAS:MARK:FUNC:DOM:USER:START and STOP commands, unless range 0 is specified which is the full span of the analyzer.

Each channel has 16 user ranges. (Trace statistics use the same ranges.) More than one marker can use a domain range.

Parameters
  <cnum>  Channel number of the measurement (optional).
  <mnum>  Measurement number for each measurement.
  <mkr>   Any marker number from 1 to 15; if unspecified, value is set to 1.
  <range> User span. Choose any Integer from 0 to 16.

  0 is Full Span of the analyzer.

  1 to 16 are available for user-defined x-axis span.

Examples
 calculate2:measure1:marker8:function:domain:user:range 1

Query Syntax

Return Type
  Numeric

Default
  0 - Full Span

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:DOMain:USER:STARt <start>
Applicable Models: All

(Read-Write) Sets the start of the span that the specified marker's x-axis span will be constrained to.


Note: If the marker is assigned to range 0 (full span), the USER:STARt and STOP commands generate an error. You cannot set the STARt and STOP values for "Full Span".

Note: This command does the same as CALC:FUNC:DOM:USER:STAR

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<start>` The analyzer's Minimum x-axis value

Examples
- `calculate2:measure1:marker8:function:domain:user:start 1e12`

Query Syntax
- `CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:DOMain:USER:STARt?`

Return
- Type: Numeric
- Default: The analyzer's Minimum x-axis value


Applicable Models: All

(Read-Write) Sets the stop of the span that the marker's x-axis travel will be constrained to.


Note: If the marker is assigned to range 0 (full span), the USER:STARt and STOP commands generate an error. You cannot set the STARt and STOP values for "Full Span".

Note: This command does the same as CALC:FUNC:DOM:USER:STOP

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
Stop value of x-axis span; Choose any number between the analyzer's MINimum and MAXimum x-axis value.

**Examples**

```
```

```
calculate2:measure1:marker8:function:domain1:user:stop 1e12
```

**Query Syntax**

```
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:DOMain:USER:STOP?
```

**Return Type**

Numeric

**Default**

The analyzer's MAXimum x-axis value.

---

**CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:EXECute <func>**

**Applicable Models:** All

*(Write-only)* Immediately executes (performs) the specified search function.

Learn more about Marker Search

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<func>` The function to be performed. Choose from:

  - MAXimum - finds the highest value.
  - MINimum - finds the lowest value.
  - RPEak - finds the next valid peak to the right.
  - LPEak - finds the next valid peak to the left.
  - NPEak - finds the next highest value among the valid peaks.
  - TARGet - finds the target value to the right, wraps around to the left.
  - LTARget - finds the next target value to the left of the marker.
  - RTARget - finds the next target value to the right of the marker.
  - PEAK - finds the peak value.
  - COMPression - finds the compression level on a Power Swept S21 trace.
  - SPURious - finds spurious signals based on spurious settings.
  - LSPurious - finds the next spurious signal to the left.
• RSPurious - finds the next spurious signal to the right.

Examples

CALC:MEAS:MARK:FUNC:EXEC MAX
calculate2:measure1:marker2:function:execute maximum

Query Syntax
Not Applicable

Default
Not Applicable

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:MULTi:EXECute <func>

Applicable Models: All

(Write-only) Immediately executes (performs) the specified multi search function.

Parameters

<cnun>  Channel number of the measurement (optional).
<mnum>  Measurement number for each measurement.
<mkr>   Any marker number from 1 to 15; if unspecified, value is set to 1.
<func>  The function to be performed. Choose from:

  • **OFF** - function is disabled.
  • **PEAK** - finds the peak value of a multi-peak search.
  • **TARGET** - finds the target value to the right, wraps around to the left.
  • **SPURIOUS** - finds spurious signals based on spurious settings.

Examples

CALC:MEAS:MARK:FUNC:MULT:EXEC PEAK
calculate2:measure1:marker2:function:multi:execute target

Query Syntax
Not Applicable

Default
OFF

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:MULTi:PEAK:EXCursion <num><unit>
Applicable Models: All

(Read-Write) Sets or returns the lower limit of peak excursion value of multi peak search, for the selected channel and selected trace.

Learn more about Marker Search

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<num>` Excursion value. Choose any number between -500 and 500.

Notes: If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

- `<unit>` Varies depending on the data format.
  - Log magnitude (MLOG): dB (decibel)
  - Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): deg (degree)
  - Group delay (GDEL): s (second)
  - Others: No unit

Examples

- `calculate2:measure2:marker8:function:multi:peak:excursion maximum`

Query Syntax

- `CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:MULTi:PEAK:EXCu`

Return Type

- Numeric

Default

- 3

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:MULTi:PEAK:POLarity <func>
Applicable Models: All

(Read-Write) Sets or returns the peak polarity of the multi peak search, for the selected channel and selected trace.

Learn more about Marker Search

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<func>` Polarity for multi peak search function to be performed. Choose from:
  - "NEGative" : Specifies the negative peak.
  - "POSitive" : Specifies the positive peak.
  - "BOTH" : Specifies both the positive peak and the negative peak.

Examples

```
```

Query Syntax

```
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:MULTi:PEAK:POLarity?
```

Return Type

Character

Default

"POSitive"

---

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:MULTi:PEAK:THReshold <num><unit>

Applicable Models: All

(Read-Write) Sets peak threshold for the specified marker. If a peak (using the criteria set with :EXCursor below this reference value, it will not be considered when searching for peaks. This command applies to marker peak searches (Next peak, Peak Right, Peak Left).

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<num>` Excursion value. Choose any number between -500 and 500.

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

- `<unit>` Varies depending on the data format.

---

3638
Log magnitude (MLOG): dB (decibel)
Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): deg (degree)
Group delay (GDEL): s (second)
Others: No unit

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate&lt;cnum&gt;:MEASure&lt;mnum&gt;:MARKer&lt;mkr&gt;:FUNCtion:MULTi:SELect&lt;char&gt;</td>
<td>character</td>
<td>&quot;OFF&quot;</td>
</tr>
</tbody>
</table>

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:MULTi:SELect<char>

Applicable Models: All

(Read-Write) Sets or returns the search type of the multi search, for the selected channel and selected trace.

Learn more about Marker Search

Parameters
- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <mkr> Any marker number from 1 to 15; if unspecified, value is set to 1.
- <char> Select from the following:
  - "OFF ": Turn OFF the multi search function.
  - "PEAK ": Sets the search type to the multi peak search.
  - "TARGet ": Sets the search type to the multi target search.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS:MARK:FUNC:MULT:SEL BOTH</td>
<td>character</td>
<td>&quot;OFF&quot;</td>
</tr>
<tr>
<td>calculate2:measure1:marker6:function:multi:select both</td>
<td>character</td>
<td>&quot;OFF&quot;</td>
</tr>
</tbody>
</table>

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:MULTi:TARGet:TRANSition
Applicable Models: All

(Read-Write) Sets the transition type of the multi target search.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<char>` Transition type of multi target search function to be performed. Choose from:
  - "NEGative": Specifies the negative transition.
  - "POSitive": Specifies the positive transition.
  - "BOTH": Specifies both the positive transition and the negative transition.

Examples
- `calculate2:measure1:marker6:function:multi:target:transition both`

Query Syntax
- `CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:MULTi:TARGet[:VALue] <num><unit>`

Applicable Models: All

(Read-Write) Sets or returns the target value for the specified marker when doing Multi Target Search, for selected channel and selected trace.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<num>` Target value for multi target search to search for.

The range of target value is -5E8 to 5E8.

Notes: If the specified variable is out of the allowable setup range, the minimum value (if t
lower limit of the range is not reached) or the maximum value (if the upper limit of the range exceeded) is set.

<unit> Varies depending on the data format.

- Log magnitude (MLOG): dB (decibel)
- Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): deg (degree)
- Group delay (GDEL): s (second)
- Others: No unit

Examples
CALC:MEAS:MARK:FUNC:MULT:TARG 2.5

Query Syntax
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:MULTi:TARGet[:VAI

Return Type Numeric

Default 0

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:MULTi:TRACking <bool>

Applicable Models: All

(Read-Write) Turns ON or OFF the search tracking capability (function to repeat search for each sweep) of the multi search, for the selected channel and selected trace.

Learn more about Marker Search

Parameters
- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <mkr> Any marker number from 1 to 15; if unspecified, value is set to 1.
- <bool> **ON or 1** - Turns ON the marker search tracking. The specified multi marker will "Track" (find) the selected function every sweep.

**OFF or 0** - Turns OFF the marker search tracking. The specified multi marker will find the selected function **only** when the CALC:MEAS:MARK:FUNC:EXECute command is sent.

Examples
CALC:MEAS:MARK:FUNC:MULT:TRAC ON
calculate2:measure2:marker8:function:multi:tracking off

Query Syntax
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:MULTi:TRACking?

Return Type Boolean
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:PEAK:EXCursion <num><unit>

**Applicable Models:** All

*(Read-Write)* Sets amplitude peak excursion for the specified marker. The Excursion value determines what is considered a "peak". This command applies to marker peak searches (Next peak, Peak Right, Peak Left).

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any existing marker number from 1 to 15; if unspecified, value is set to 1.
- `<num>` Excursion value. Choose any number between -500 and 500.

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

- `<unit>` Varies depending on the data format.
  - Log magnitude (MLOG): dB (decibel)
  - Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): deg (degree)
  - Group delay (GDEL): s (second)
  - Others: No unit

**Examples**

- `CALC:MEAS:MARK:FUNC:PEAK:EXC 10`
- `calculate2:measure2:marker8:function:peak:excursion maximum`

**Query Syntax**

`CALCulate<cnum>:MARKer<mkr>:FUNCtion:PEAK:EXCursion?`

**Return Type**

Numeric

**Default**

3

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:PEAK:POLarity <char>
Applicable Models: All

*(Read-Write)* Sets or returns polarity of the peak search with marker 1 to 15 and reference marker (Mk :16), for the active trace of selected channel.

Learn more about Marker Search

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement (optional).</td>
</tr>
<tr>
<td>&lt;mnum&gt;</td>
<td>Measurement number for each measurement.</td>
</tr>
<tr>
<td>&lt;mkr&gt;</td>
<td>Any marker number from 1 to 15; if unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;char&gt;</td>
<td>Polarity for peak search function to be performed. Choose from:</td>
</tr>
<tr>
<td></td>
<td>&quot;NEGative&quot; : Specifies the negative peak.</td>
</tr>
<tr>
<td></td>
<td>&quot;POSitive&quot; : Specifies the positive peak.</td>
</tr>
<tr>
<td></td>
<td>&quot;BOTH&quot; : Specifies both the positive peak and the negative peak.</td>
</tr>
</tbody>
</table>

**Examples**

```
CALC:MEAS:MARK:FUNC:APE:POL NE
```

calculate2:measure1:marker6:function:apeak:polarity both

**Query Syntax**

```
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:APEak:POLarity?
```

**Return Type**

Character

**Default**

"POSitive"

---

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:PEAK:THReshold <num><unit>

Applicable Models: All

*(Read-Write)* Sets peak threshold for the specified marker. If a peak (using the criteria set with :EXCursion) is below this reference value, it will not be considered when searching for peaks. This command applies to marker peak searches (Next peak, Peak Right, Peak Left).

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement (optional).</td>
</tr>
<tr>
<td>&lt;mnum&gt;</td>
<td>Measurement number for each measurement.</td>
</tr>
<tr>
<td>&lt;mkr&gt;</td>
<td>Any marker number from 1 to 15; if unspecified, value is set to 1</td>
</tr>
<tr>
<td>&lt;num&gt;</td>
<td>Excursion value. Choose any number between -500 and 500.</td>
</tr>
</tbody>
</table>

**Note**: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.
<unit>  
Varies depending on the data format.

- Log magnitude (MLOG): dB (decibel)
- Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): deg (degree)
- Group delay (GDEL): s (second)
- Others: No unit

Examples

Query Syntax
CALCulate<cnum>:MARKer<mkr>:FUNCtion:APEak:THReshold?

Return Type
Numeric

Default
-100

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion[:SELect] <char>

Applicable Models: All

(Read-Write) Sets the search function that the specified marker will perform when executed. Use CALC:MEAS:MARK:FUNC:TRAC ON to automatically execute the search every sweep.

Learn more about Marker Search

Parameters
- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <mkr> Any marker number from 1 to 15; if unspecified, value is set to 1.
- <char> Marker function. Choose from:
  
  - MAXimum - finds the highest value
  - MINimum - finds the lowest value
  - RPEak - finds the next valid peak to the right
  - LPEak - finds the next valid peak to the left
  - NPEak - finds the next highest value among the valid peaks
  - TARGet - finds the target value to the right, wraps around to the left
  - LTARget - finds the next target value to the left of the marker
  - RTARget - finds the next target value to the right of the marker
COMPression - finds the compression level on a power-swept S21 trace.

Examples

```
CALC:MEAS:MARK:FUNC MAX
calculate2:measure1:marker8:function:select 1target
```

Query Syntax
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion[:SELect]?

Return Type
Character

Default
MAXimum

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:TARGet[:VALue]:TRANSition <char>

Applicable Models: All

(Read-Write) Selects the transition type of the target search for specified marker (marker 1 to 15 and reference marker (Mk :16)) of the active trace of selected channel.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<char>` Transition type for search function to be performed. Choose from:
  - "NEGative ": Specifies the negative transition.
  - "POSitive ": Specifies the positive transition.
  - "BOTH ": Specifies both the positive transition and the negative transition.

Examples

```
CALC:MEAS:MARK:FUNC:TARG:TRAN POS
calculate2:measure1:marker8:function:target:value:transition both
```

Query Syntax
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:TARGet[:VALue]:TRANsition

Return Type
Character

Default
"BOTH"

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:TARGet[:VALue]<num><unit>
Applicable Models: All

(Read-Write) Sets the target value for the specified marker when doing Target Searches with 
CALC:MEAS:MARK:FUNC:SEL <TARGET | RTARGET | LTARGET>

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<num>` Target value to search for.

The range of value is between -5E8 to 5E8.

**Notes:** If the specified variable is out of the allowable setup range, the minimum value 
(if the lower limit of the range is not reached) or the maximum value (if the upper limit 
of the range is exceeded) is set.

- `<unit>` Varies depending on the data format.

  - Log magnitude (MLOG): dB (decibels)
  - Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): deg (degree)
  - Group delay (GDEL): s (second)
  - Others: No unit

**Examples**

- `CALC:MEAS:MARK:FUNC:TARG 2.5`
- `calculate2:measure1:marker8:function:target:value -10.3`

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:TARGET[:VALue]?

**Return Type**

- Numeric

**Default**

- 0

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCTION:TRACING <bool>
Applicable Models: All

(Read-Write) Turns ON or OFF the tracking search capability for the specified marker. The tracking function finds the selected search function every sweep. In effect, turning Tracking ON is the same as doing a CALC:MEAS:MARK:FUNC:EXECute command every sweep.

Learn more about Marker Search

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<bool>` ON or 1 - Turns ON the search tracking. The specified marker will "Track" (find) the selected function every sweep.

OFF or 0 - Turns OFF the search tracking. The specified marker will find the selected function only when the CALC:MEAS:MARK:FUNC:EXECute command is sent.

Examples

```
CALC:MEAS:MARK:FUNC:TRAC ON
calculate2:measure1:marker8:function:tracking off
```

Query Syntax

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:FUNCtion:TRACking?

Return Type

Boolean

Default

OFF or 0

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:NOTCh:DATA?

Applicable Models: All

(Read-only) Reads the notch search result of marker 1 to 15 and reference marker (Mk :16), for the active trace of selected channel.

If the notch search is impossible, an error occurs and the command is ignored. In this case, no query response is obtained.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.

Indicates 4-element array data (notch bandwidth search result). Four Character values separated by commas => {Data 1}, {Data 2}, {Data 3}, {Data 4}

- Data(0) : The bandwidth.
- Data(1): Center point frequency of the 2 cutoff frequency points.
- Data(2): The Q value.
- Data(3): Insertion loss

The index of the array starts from 0.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS:MARK:NOTCh:DATA?</td>
</tr>
<tr>
<td>calculate2:measure1:marker:notch:data?</td>
</tr>
</tbody>
</table>

**Return Type**

- Variant
- Not Applicable

---

CALCulate<chnum>:MEASure<mnum>:MARKer<mkr>:NOTCh:REF <string>

**Applicable Models:** All

*(Read-Write)* Set the notch marker reference to either MARKer or PEAK.

If the reference is set to MARKer, the active marker is not moved; the notch search is computed at the marker's current location.

If the reference is set to PEAK, the active marker is moved to the maximum or minimum peak on the trace and then notch search is computed.

- If the notch level is negative, the active marker is moved to the maximum peak.
- If the notch level is positive, the active marker is moved to minimum peak.

**Parameters**

- `<chnum>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
- `<mkr>`: Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<string>`: PEAK

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS:MARK:NOTCh:REF</td>
</tr>
<tr>
<td>calculate2:measure1:marker:notch:ref</td>
</tr>
</tbody>
</table>

**Return Type**

- String
- MARKer

**Default**

MARKer
CALCulate\(<cnum>:MEASure\(<mnum>:MARKer\(<mkr>:NOTCh[:STATe] <bool>

Applicable Models: All

(Read-Write) Turns ON or OFF the notch search result display, for the active trace of selected channel.

Parameters

\(<cnum>\) Channel number of the measurement (optional).
\(<mnum>\) Measurement number for each measurement.
\(<mkr>\) Any marker number from 1 to 15; if unspecified, value is set to 1.
\(<bool>\) Notch search result display. Choose from:

ON or 1 - Turns ON the notch search result display.
OFF or 0 - Turns OFF the notch search result display.

Examples

CALC:MEAS:MARK:NOTC ON
calculate2:measure1:marker:notch:state off

Query Syntax

CALCulate\(<cnum>:MEASure\(<mnum>:MARKer\(<mkr>:NOTCh[:STATe]\)?

Return Type

Boolean

Default

OFF or 0

CALCulate\(<cnum>:MEASure\(<mnum>:MARKer\(<mkr>:NOTCh:THReshold <num><unit>

Applicable Models: All

(Read-Write) Sets or returns the notch definition value of marker 1 to 15 and reference marker (Mk :16), for the active trace of selected channel.

Parameters

\(<cnum>\) Channel number of the measurement (optional).
\(<mnum>\) Measurement number for each measurement.
\(<mkr>\) Any marker number from 1 to 15; if unspecified, value is set to 1.
\(<num>\) The notch definition value range is between -5E8 to 5E8.
\(<unit>\) Varies depending on the data format as follows:

- Amplitude (MLOG): dB (decibel)
- Phase (PHAS), Expanded phase (UPH), Positive phase (PPH): deg (degree)
- Group delay (GDEL): s (second)

Notes: If the specified parameter is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.
**Examples**

```
CALC:MEAS:MARK:NOTC:THR -3
```

```
calculate2:measure1:marker:notch:threshold -3
```

**Query Syntax**

```
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:NOTCh:THReshold?
```

**Return Type**

Numeric

**Default**

-3

---

**CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:BACKoff <num>**

**Applicable Models:** All

(Read-Write) Turns on and sets markers 1, 2, 3, and 4 to calculate various PNOP parameters.

Either this command, or the POFFset command, will initiate the PNOP search markers.

To turn off the PNOP markers, either turn them off individually or turn them All Off.

To search a User Range with the PNOP search, first activate marker 1 and set the desired User Range. Then send CALC:MARK:PNOP:BACK. The user range used with the PNOP search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Backoff value. Choose any number between -500 and 500

**Examples**

```
CALC:MEAS:MARK:PNOP:BACK?
```

```
calculate2:measure1:marker:pnop:backoff 10
```

**Query Syntax**

```
CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:BACKoff?
```

**Return Type**

Numeric

**Default**

0

---

**CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:BACKoff:GAIN?**
**Applicable Models:** All

*(Read-only)* Reads the power backoff gain value from a PNOP marker search.

\[ \text{PBO Gain} = \text{PBO Out} - \text{PBO In} \]


**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS1:MARK:PNOP:BACK:GAIN?
```

**Default** Not applicable

---

**CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:BACKoff:PIN?**

**Applicable Models:** All

*(Read-only)* Reads the power backoff input value from a PNOP marker search.

\[ \text{PBO In} = \text{Marker 2 X-axis} \]


**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS1:MARK:PNOP:BACK:PIN?
```

**Default** Not applicable

---

**CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:BACKoff:POUT?**
**Applicable Models:** All

*(Read-only)* Reads the power backoff output value from a PNOP marker search.

PBO Out = Marker 2 Y-axis


**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS1:MARK:PNOP:BACK:POUT?
```

**Default** Not applicable

---

**CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:COMPression?**

**Applicable Models:** All

*(Read-only)* Reads the PNOP compression value from a PNOP marker search.

Pnop Comp = Pnop Gain - Linear Gain (not shown on marker readout).


**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS1:MARK:PNOP:COMP?
```

**Default** Not applicable

---

**CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:COMPression:MAXimum?**
**Applicable Models:** All

*(Read-only)* Reads the max compression value from a PNOP marker search.

Comp Max = Gain Max - Linear Gain (not shown on marker readout).


**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS1:MARK:PNOP:COMP:MAX?
```

**Default** Not applicable

---

**CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:GAIN?**

**Applicable Models:** All

*(Read-only)* Reads the PNOP gain value from a PNOP marker search.

Pnop Gain = Pnop Out - Pnop In.


**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS:MARK:PNOP:GAIN?
```

**Default** Not applicable

---

**CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:GAIN:MAXimum?**
Applicable Models: All

(Read-only) Reads the max gain from a PNOP marker search.

Gain Max = PMax Out - PMax In


**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS:MARK:PNOP:GAIN:MAX?
```

**Default** Not applicable

---

CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:PIN?

Applicable Models: All

(Read-only) Reads the PNOP input value from a PNOP marker search.

Pnop In = Marker 4 X-axis value


**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS:MARK:PNOP:PIN?
```

**Default** Not applicable

---

CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:PIN:MAXimum?
**Applicable Models:** All

*(Read-only)* Reads the max input power from a PNOP marker search.

\[ \text{PMax In} = \text{Marker 3 X-axis value} \]


**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**
- `CALC:MEAS:MARK:PNOP:PIN:MAX?`
- `CALC:MEAS1:MARK:PNOP:POFF 3`
- `CALC:MEAS2:MARK:PNOP:POFF 10`

**Default** Not applicable

---

CALCulate\(<cnum>\):MEASure\(<mnum>\):MARKer:PNOP:POFFset \(<num>\)

**Applicable Models:** All

*(Read-Write)* Turns on and sets markers 1, 2, 3, and 4 to calculate various PNOP parameters.

Either this command, or the Backoff command, will initiate the PNOP search markers.

To turn off the PNOP markers, either turn them off individually or turn them All Off.

To search a User Range with the PNOP search, first activate marker 1 and set the desired User Range. Then send the `CALC:MARK:PNOP:POFF` command. The user range used with the PNOP search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Power Offset value in dB. Choose any number between \(-500\) and \(500\).

**Examples**
- `CALC:MEAS1:MARK:PNOP:POFF 3`
- `CALC:MEAS2:MARK:PNOP:POFF 10`

**Query Syntax** `CALCulate\(<cnum>\):MEASure\(<mnum>\):MARKer:PNOP:POFFset?`

**Return Type** Numeric

**Default** 0
CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:POUT?

**Applicable Models:** All

*(Read-only)* Reads the output power value of the offset marker from a PNOP marker search.

\[ \text{Pnop Out} = \text{Marker 4 Y-axis value} \]


**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC : MEAS2 : MARK : PNOP : POUT?
```

**Default** Not applicable

CALCulate<cnum>:MEASure<mnum>:MARKer:PNOP:POUT:MAXimum?

**Applicable Models:** All

*(Read-only)* Reads the max output power from a PNOP marker search.

\[ \text{PMax Out} = \text{Marker 3 Y-axis value} \]


**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC : MEAS2 : MARK : PNOP : POUT : MAX?
```

**Default** Not applicable

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:PNOP[:STATE] <ON|OFF>
Applicable Models: All

(Read-Write) Turns the PNOP marker search ON and OFF.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<ON|OFF>`
  - `ON` (or 1) - turns marker ON.
  - `OFF` (or 0) - turns marker OFF.

**Examples**

```
CALC:MEAS1:MARK:PNOP ON
CALC:MEAS2:MARK:PNOP ON
```

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:PNOP:STATe?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

Off

---

CALCulate<cnum>:MEASure<mnum>:MARKer:PSATuration:BACKoff <num>

Applicable Models: All

(Read-Write) Turns on and sets markers 1, 2, and 3 to calculate various Power Saturation parameters. The `<num>` parameter sets and reads the back-off value for a Power Saturation marker search.

To turn off the Power Saturation markers, either turn them off individually or turn them All Off.

To search a User Range with the PSAT search, first activate marker 1 and set the desired User Range. Then send the CALC:MARK:PSAT:BACK command. The user range used with the PSAT search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Backoff value. Choose any number between -500 and 500

**Examples**

```
CALC:MEAS2:MARK:PSAT:BACK 3
CALC:MEAS2:MARK:PSAT:BACKoff 10
```

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:MARKer:PSATuration:BACKoff?
CALCulate<cnum>:MEASure<mnum>:MARKer:PSATuration:COMPression:MAXimum?

**Applicable Models:** All

*(Read-only)* Reads the compression maximum value from a PSAT marker search.

Comp Max = Gain Max - Gain Linear

Use CALC:MARK:PSAT:BACK to initiate a PSAT search.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS2:MARK:PSAT:COMP:MAX?
```

**Default** Not applicable

---

CALCulate<cnum>:MEASure<mnum>:MARKer:PSATuration:COMPression:SATuration?

**Applicable Models:** All

*(Read-only)* Reads the compression saturation value from a PSAT marker search.

Comp Sat = Gain Sat - Gain Linear

Use CALC:MARK:PSAT:BACK to initiate a PSAT search.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS2:MARK:PSAT:COMP:SAT?
```

**Default** Not applicable

---

CALCulate<cnum>:MEASure<mnum>:MARKer:PSATuration:GAIN?
**Applicable Models:** All

*(Read-only)* Reads the saturation gain value from a PSAT marker search.

Gain Sat = Psat Out - Psat In

Use CALC:MARK:PSAT:BACK to initiate a PSAT search.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

<table>
<thead>
<tr>
<th>CALCULATE</th>
<th>MEASURE</th>
<th>MARKER</th>
<th>PSATURATION</th>
<th>GAIN</th>
<th>LINEAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>&lt;mnum&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Default** Not applicable

---

**CALCulate<cnum>:MEASure<mnum>:MARKer:PSATuration:GAIN:MAXimum?**

**Applicable Models:** All

*(Read-only)* Reads the linear gain value from a PSAT marker search.

Gain Linear = Marker 1 - Y-axis value MINUS X-axis value.

Use CALC:MARK:PSAT:BACK to initiate a PSAT search.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

<table>
<thead>
<tr>
<th>CALCULATE</th>
<th>MEASURE</th>
<th>MARKER</th>
<th>PSATURATION</th>
<th>GAIN</th>
<th>LINEAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>&lt;mnum&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Default** Not applicable
**Applicable Models:** All

*(Read-only)* Reads the maximum gain value from a PSAT marker search.

Gain Max = PMax Out - PMax In

Use CALC:MARK:PSAT:BACK to initiate a PSAT search.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CALC:MEAS2:MARK:PSAT:GAIN:MAX?</strong></td>
<td>Gain Max</td>
</tr>
<tr>
<td><strong>CALC:MEAS2:MARK:PSAT:PIN?</strong></td>
<td>Psat In</td>
</tr>
<tr>
<td><strong>CALC:MEAS2:MARK:PSAT:PIN:MAXimum?</strong></td>
<td>Psat Max</td>
</tr>
</tbody>
</table>

**Default** Not applicable

---

**CALCulate<cnum>:MEASure<mnum>:MARKer:PSATuration:PIN?**

**Applicable Models:** All

*(Read-only)* Reads the power saturation input value from a PSAT marker search.

Psat In = Marker 2 X-axis value

Use CALC:MARK:PSAT:BACK to initiate a PSAT search.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CALC:MEAS2:MARK:PSAT:PIN?</strong></td>
<td>Psat In</td>
</tr>
<tr>
<td><strong>CALC:MEAS2:MARK:PSAT:PIN:MAXimum?</strong></td>
<td>Psat Max</td>
</tr>
</tbody>
</table>

**Default** Not applicable

---

**CALCulate<cnum>:MEASure<mnum>:MARKer:PSATuration:PIN:MAXimum?**
Applicable Models: All

(Read-only) Reads the maximum input power from a PSAT marker search.

PMAX In = Marker 3 X-axis value

Use CALC:MARK:PSAT:BACK to initiate a PSAT search.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS2:MARK:PSAT:PIN:MAX?
```

**Default** Not applicable

---

CALCulate<cnum>:MEASure<mnum>:MARKer:PSATuration:POUT?

Applicable Models: All

(Read-only) Reads the back-off output power from a PSAT marker search.

PSat Out = Marker 2 Y-axis value

Use CALC:MARK:PSAT:BACK to initiate a PSAT search.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS2:MARK:PSAT:POUT?
```

**Default** Not applicable

---

CALCulate<cnum>:MEASure<mnum>:MARKer:PSATuration:POUT:MAXimum?
Applicable Models: All

(Read-only) Reads the back-off output power from a PSAT marker search.

PMaxOut = Marker 3 Y-axis value

Use CALC:MARK:PSAT:BACK to initiate a PSAT search.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS2:MARK:PSAT:POUT:MAX?</td>
</tr>
</tbody>
</table>

Default Not applicable

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:PSATuration[:STATe] <ON|OFF>

Applicable Models: All

(Read-Write) Turns the PSAT marker search ON and OFF.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<ON|OFF>` ON (or 1) - turns marker ON.
  
  OFF (or 0) - turns marker OFF.

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS1:MARK:PSAT ON</td>
</tr>
<tr>
<td>calculate2:measure2:marker8:psaturation on</td>
</tr>
</tbody>
</table>

Query Syntax

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:PSATuration:STATe?

Return Type

Boolean (1 = ON, 0 = OFF)

Default Off

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:REFerence[:STATe] <bool>
Applicable Models: All

(Read-Write) Turns ON or OFF the reference marker mode, for the active trace of selected channel. When turned OFF, existing Delta markers revert to absolute markers.

Parameters
- \(<cnum>\): Channel number of the measurement (optional).
- \(<mnum>\): Measurement number for each measurement.
- \(<mkr>\): Any marker number from 1 to 15; if unspecified, value is set to 1.
- \(<bool>\): ON or 1 - Turns reference marker mode ON. OFF or 0 - Turns reference marker mode OFF.

Examples
- `CALC:MEAS:MARK:REF ON`
- `calculate2:measure1:marker:reference:state OFF`

Query Syntax
- `CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:REFerence[:STATe]?`

Return Type
- Boolean

Default
- OFF or 0

CALCulate\(<cnum>:MEASure\(<mnum>:MARKer\(<mkr>:REFerence:X <num>

Applicable Models: All

(Read-Write) Sets and returns the absolute x-axis value of the reference marker.

Parameters
- \(<cnum>\): Channel number of the measurement (optional).
- \(<mnum>\): Measurement number for each measurement.
- \(<mkr>\): Any marker number from 1 to 15; if unspecified, value is set to 1.
- \(<num>\): X-axis value. Choose any number within the operating domain of the reference marker.

Examples
- `CALC:MEAS:MARK:REF:X 1e9`
- `calculate2:measure1:marker:reference:x 1e6`

Query Syntax
- `CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:REFerence:X?`

Return Type
- Numeric

Default
- If the first Marker, turns ON in the middle of the X-axis span. If not, turns ON at the position of the active marker.

CALCulate\(<cnum>:MEASure\(<mnum>:MARKer\(<mkr>:REFerence:Y <num>
Applicable Models: All

(Read-Write) Sets and returns the absolute Y-axis value of the reference marker (Set the reference marker Y position only when the marker is a fixed marker type).

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<num>` Y-axis value. Choose any number within the operating domain of the reference marker.

**Examples**

```
CALC:MEAS:MARK:REF:Y 1e6
calculate2:measure1:marker:reference:y 1e9
```

**Return Type**

Numeric

**Default**

Not Applicable

---

**CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:SET <char>**

Applicable Models: All

(Write-only) Sets the selected instrument setting to assume the value of the specified marker.

Marker Functions CENT, SPAN, STARt, and STOP do not work with channels that are in CW or Segment Sweep mode.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1
- `<char>` Choose from:

  - **CENTer** - changes center frequency to the value of the marker.
  - **SPAN** - changes the sweep span to the span that is defined by the delta marker and the marker that it references. Unavailable if there is no delta marker.
  - **STARt** - changes the start frequency to the value of the marker.
  - **STOP** - changes the stop frequency to the value of the marker.
  - **RLEVel** - changes the reference level to the value of the marker.
  - **DELay** - changes the line length at the receiver input to the phase slope at the active marker stimulus position.
  - **CWFReq** - Sets the CW frequency to the frequency of the active marker. Does NOT
change sweep type. NOT available in CW or Power Sweep. Use this argument to first set
the CW Frequency to a value that is known to be within the current calibrated range, THEN
set Sweep:Type to POWer or CW.

- **SA** - creates an SA channel with a marker at the same CW frequency. Learn more.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Applicable</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:STATe <bool>**

**Applicable Models:** All  

*(Read-Write)* Turns ON or OFF the specified marker. Marker 16 is the Reference Marker. To turn all markers OFF, use **CALC:MEAS:MARK:AOFF**.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<bool>` ON or 1 - Turns marker ON.  
  OFF or 0 - Turns marker OFF.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Applicable</td>
<td>Boolean</td>
<td>OFF or 0</td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:TYPE <char>**
Applicable Models: All

(Read-Write) Sets the type of the specified marker.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1
- `<char>` Choose from:
  - NORMal - a marker that stays on the assigned X-axis position unless moved or searching.
  - FIXed - a marker that will not leave the assigned X or current Y-axis position.

Examples

```
CALC:MEAS:MARK:TYPE NORM
```

```
calculate2:measure1:marker2:type fixed
```

Query Syntax

```
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:TYPE?
```

Return Type

Character

Default

NORMal

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:X <num>

Applicable Models: All

(Read-Write) Sets the marker's X-axis value (frequency, power, or time). If the marker is set as delta, the SET and QUERY data is relative to the reference marker.

Parameters

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<num>` Any X-axis position within the measurement span of the marker.

(When the span value of the sweep range is 0, the range is from 0 to sweep time value.)

Notes: If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

- `<unit>` Hz (hertz), dBm or s (second)

Examples

```
CALC:MEAS:MARK:X 100Mhz
```

```
calculate2:measure1:marker8:x maximum
```
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:X?

Return Type
Numeric

Default
First Marker turns ON in the middle of the X-axis span. Subsequent markers turn ON at the position of the active marker.

(When the span value of the sweep range is 0, the preset value is 0.)

CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:Y?

Applicable Models: All

(Read-only) Reads the marker's Y-axis value. The format of the value depends on the current CALC:MEAS:MARK:FORMAT setting. If the marker is set as delta, the data is relative to the reference marker. The query always returns two numbers:

- Smith and Polar formats - (Real, Imaginary)
- LINPhase and LOGPhase - (Real, Imaginary)
- All other formats - (Value,0)

Note: To accurately read the marker Y-axis value with trace smoothing applied, the requested format must match the displayed format. Otherwise, the returned value is un-smoothed data. For example, to read the smoothed marker value when measuring group delay, both the display format and the marker format must be set to (Group) Delay.

Parameters
- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <mkr> Any marker number from 1 to 15; if unspecified, value is set to 1.

Examples
- CALC:MEAS:MARK:Y?
- calculate2:measure1:marker3:y?

Query Syntax
CALCulate<cnum>:MEASure<mnum>:MARKer<mkr>:Y?

Return Type
Numeric

Default
Not Applicable
CALCulate:MEASure:OFFSet Commands

Allows the data trace magnitude and phase to be offset.

<table>
<thead>
<tr>
<th>CALCulate:MEASure:OFFSet</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAGNitude</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>PHASE</td>
</tr>
</tbody>
</table>

Click a keyword to view the command details.

See Also

- Learn about Magnitude Offset
- Learn about Phase Offset
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

CALCulate<cnum>:MEASure<mnum>:OFFSet:MAGNitude <num>

Applicable Models: All

(Read-Write) Offsets the data trace magnitude by the specified value.

To offset the data trace magnitude to a slope value that changes with frequency, use

CALC:MEAS:OFFS:MAGN:SLOP

Parameters

- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <num> Offset value in dB.

Examples

- CALC:MEAS:OFFS:MAGN:4
calculate1:measure2:offset:magnitude -2

Query Syntax

CALCulate<cnum>:MEASure<mnum>:OFFSet:MAGNitude?

Return Type

- Numeric
  - Default: 0

CALCulate<cnum>:MEASure<mnum>:OFFSet:MAGNitude:SLOPe <num>
Applicable Models: All

(Read-Write) Offsets the data trace magnitude to a value that changes linearly with frequency. The offset slope begins at 0 Hz.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Offset slope value in dB/1GHz.

Examples
- `CALC:MEAS:OFFS:MAGN:SLOP 1 'Offset slope set to 1dB/GHz`
- `calculate1:measure2:offset:magnitude:slope -2 'Offset slope set to -2dB/GHz`

Query Syntax
- `CALC:MEASure<mnum>:OFFSet:MAGNitude:SLOPe?`

Return Type
- Numeric
- Default: 0

CALCulate<cnum>:MEASure<mnum>:OFFSet:PHASe <num>[<char>]

Applicable Models: All

(Read-Write) Sets the phase offset for the selected measurement.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Offset phase value. Choose any number between: -360 and 360
- `<char>` Units for phase. OPTIONAL. Choose either: DEG - Degrees (default)
- RAD - Radians

Examples
- `CALC:MEAS:OFFS:PHAS 10`
- `calculate3:measure2:offset:phase 20rad`

Query Syntax
- `CALC:MEASure<mnum>:OFFSet:PHASe?`

Return Type
- Numeric, returned value always in degrees
- Default: 0 degrees
CALCulate:MEASure:PARameter Commands

Selects a measurement parameter.

```
CALCulate:MEASure
    PARameter
```

Click a keyword to view the command details.

See Also

- Learn about Measurement Parameters
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

```
CALCulate<cnum>:MEASure<mnum>:PARameter <string>
```

**Applicable Models:** All

*(Read-Write)* Set/get a measurement parameter for the specified (cnum/mnum) measurement.

This command replaces the following commands:

- `CALC:CUST:MOD`
- `CALC:PAR:MOD:EXT`
- `CALC:FSIM:BAL:PAR:SBAL[:DEF]`
- `CALC:FSIM:BAL:PAR:SSB[:DEF]`
- `CALC:FSIM:BAL:PAR:BBAL[:DEF]`
- `CALC:FSIM:BAL:PAR:BALS[:DEF]`
- `CALC:FSIM:BAL:PAR:BAL[:DEF]`

**Note:** For Application Measurements see `CALCulate:MEASure:DEFine`

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement (optinal).</td>
</tr>
<tr>
<td>&lt;mnum&gt;</td>
<td>Measurement number for each measurement.</td>
</tr>
</tbody>
</table>
For S-parameters:

Any S-parameter available in the VNA

Single-digit port numbers CAN be separated by "_" (underscore). For example: "S21" or "S2_1"

Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

For ratioed measurements:

Any two VNA physical receivers separated by forward slash '/' followed by comma and source port.

For example: "A/R1, 3"

Learn more about ratioed measurements

See a block diagram showing the receivers in YOUR VNA.

For non-ratioed measurements:

Any VNA physical receiver followed by comma and source port.

For example: "A, 4"

Learn more about unratioed measurements.

See the block diagram showing the receivers in YOUR VNA.

Ratioed and Unratioed measurements can also use logical receiver notation to refer to receivers. This notation makes it easy to refer to receivers with an external test set connected to the VNA. You do not need to know which physical receiver is used for each test port. Learn more.

For ADC measurements:

Any ADC receiver in the VNA followed by a comma, then the source port.

For example: "AI1,2" indicates the Analog Input1 with source port of 2.

Learn more about ADC receiver measurements.

For Balanced Measurements:

For 1 port balanced measurement, choose from:
For Balanced - Single-ended measurement, choose from:

Sdd11, Scd11, Sdc11, Scc11, Ssd21, Ssc21, Sds22, Scs22, Imb, CMMR1, CMMR2

- Imb = - S_{1pos_2}/S_{1neg_2}
- CMMR1 = Ssd21/Ssc21
- CMMR2 = Sds21/Scs21

For Balanced - Single-ended - Single-ended measurement, choose from:

Sdd11, Scd11, Sdc11, Scc11, Ssd21, Ssc21, Ssd31, Ssc31, Sds12, Scs12, Sds13, Ssc13, Sss22, Sss32, Sss23, Sss33, Imbal1, Imbal2, Sds12/Scs12, Sds13/Scs13

For Single-ended - Balanced measurement, choose from:

Sss11, Sds21, Ssc21, Ssd12, Ssc12, Sdd22, Scd22, Sdc22, CMMR1, CMMR2

- Imb = - S_{2pos_1}/S_{2neg_1}
- CMMR1 = Sds21/Ssc21
- CMMR2 = Ssd21/Ssc21

For Balanced - Balanced measurement, choose from:

Sdd11, Sdd21, Sdd12, Sdd22, Scd11, Scd21, Sdc12, Sdc22, Scc11, Scc21, Scc12, Scc22, Imb1, Imb2, CMMR

- Imb1 = -(S_{1pos_2pos} - S_{1pos_2neg})/(S_{1neg_2pos} - S_{1neg_2neg})
- Imb2 = -(S_{2pos_1pos} - S_{2pos_1neg})/(S_{2neg_1pos} - S_{2neg_1neg})
- CMMR = - Sdd21/Scc21

For Single-ended - Single-ended - Balanced measurement, choose from:

Sss11, Sss21, Sss12, Sss22, Ssd31, Ssc31, Sds32, Ssc32, Ssd13, Ssc13, Ssd23, Ssc23, Sdd33, Ssc33, Sdc33, CMMR1, CMMR2

- Imb1 = -(S_{1pos_2pos} - S_{1pos_2neg})/(S_{1neg_2pos} - S_{1neg_2neg})
- Imb2 = \(-\frac{S_{2pos_1pos} - S_{2pos_1neg}}{S_{2neg_1pos} - S_{2neg_1neg}}\)
- Imb3 = \(-\frac{S_{3pos_1}}{S_{3neg_1}}\)
- Imb4 = \(-\frac{S_{3pos_2}}{S_{3neg_2}}\)
- CMMR1 = \(\frac{S_{ds31}}{Scs31}\)
- CMMR2 = \(\frac{S_{ds32}}{Scs32}\)

**Note:** The right definition for SSB imbalance is added as Imb3, 4. The definition for SSB Imb1, 2 seem a mistake, but keep it remained for backward compatibility.

Choose from the following (click or scroll down to view valid measurement parameters for each measurement class)

- "Standard"
- "Active Hot Parameters"
- "Vector Mixer/Converter"
- "Scalar Mixer/Converter"
- "Gain Compression"
- "Gain Compression Converters"
- Modulation Distortion
- "Noise Figure Cold Source"
- "Noise Figure Converters"
- Phase Noise
- "Swept IMD"
- "IM Spectrum"
- "Swept IMD Converters"
- "IM Spectrum Converters"
- Impedance Measurement
- "Differential I/Q"
- "Spectrum Analyzer"

*(variant)* Measurement names to create:
<table>
<thead>
<tr>
<th>Meas Class</th>
<th>Measurement Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Standard&quot;</td>
<td>&quot;S11&quot;, &quot;S21&quot;, and so forth</td>
<td>S-parameter name</td>
</tr>
<tr>
<td></td>
<td>&quot;A_1&quot;, &quot;A_2&quot;, and so forth</td>
<td>Unratioed parameter names with notation: &quot;receiver_source port&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See balanced parameter names</td>
</tr>
<tr>
<td>&quot;Active Hot Parameters&quot;</td>
<td>Port 1 is the Source Port (DUT input). Port 3 or Port 2 can be chosen as the output of the DUT.</td>
<td>Learn about Active Hot parameters</td>
</tr>
<tr>
<td></td>
<td>&quot;HotS11&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;HotS31&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;HotS13&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;HotS33&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;IPwr&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;OPwr&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Gamma&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Pmax&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Xs(3,3)&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Xt(3,3)&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Xf(3,1)&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;DeltaOPwr&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;Vector&quot;</td>
<td>For output port Y (input port must be 1):</td>
<td>Learn about VMC parameters</td>
</tr>
<tr>
<td></td>
<td>&quot;S11&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;VCY1&quot;</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** Input and output ports are set up using the Mixer Setup dialog. If the ports are not set up...
### Mixer/Converter

"SYY" using the Mixer Setup dialog, then ports 1 and 2 are the default input and output ports and the only ports that can be used.

### For input port X and output port Y:

- "SCXY"
- "SCXY"
- "SXX"
- "SYY"
- "Ipwr"
- "RevIPwr"
- "Opwr"
- "RevOPwr"

**Note:** Input and output ports are set up using the Mixer Setup dialog. If the ports are not set up using the Mixer Setup dialog, then ports 1 and 2 are the default input and output ports and the only ports that can be used.

### Gain Compression

**Learn more**

**GCA and GCX:**

- "CompIn21" Input power at the compression point.
- "CompOut21" Output power at the compression point.
- "CompGain21" Gain at the compression point.
- "CompS11" Input Match at the compression point
- "RefS21" Linear Gain
- "DeltaGain21" CompGain21 -Linear Gain
- "S11", "S21", "S12", "S22" Standard S-parameters; measured at port 1 and port 2

**GCX - All Gain Compression parameters (except S21 and S12) plus the following:**

- "S11"
- "SC21"
- "SC12"
| "S22" | Mixer parameters |
| "Ipwr" |
| "RevIPwr" |
| "Opwr" |
| "RevOPwr" |

**Modulation Distortion:**

| "POut2" | Power Out |
| "PIn1" | Power In |
| "MSig2" | Modulation Signal Out |
| "MDist2" | Modulation Distortion Out |
| "MGain21" | Modulation Gain |
| "MComp21" | Modulation Compression |
| "PGain21" | Power Gain |
| "S11" | Linear Input Match |
| "S21" | Linear Gain |
| "LPIn1" | Linear Input Match |
| "LPOut1" | Linear Reflected Power In |
| "LPOut2" | Linear Power Out |
| "A", "b1" | Port 1 test port receiver |
| "B", "b2" | Port 2 test port receiver |
| "R1", "a1" | Port 1 reference receiver |
| "R2", "a2" | Port 2 reference receiver |
| "CarrIn1" | Input Band Power |
| "CarrOut2" | Output Band Power |
| "CarrGain21" | Band Power Gain |
| "ACPIn1" | ACP at input |
| "ACPOut2" | ACP at output |
| "ACPDist21" | ACP distortion, Added by DUT |
| "EVMDistEq21" | EVM Equalized Distortion, Added by DUT |
| "EVMDistUn21" | EVM Unequalized Distortion, Added by DUT |

**Noise Figure AND NFX:**

<p>| &quot;NF&quot; | Noise figure |</p>
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;ENR&quot;</td>
<td>Validate noise source measurements.</td>
</tr>
<tr>
<td>&quot;T-Eff&quot;</td>
<td>Effective noise temperature.</td>
</tr>
<tr>
<td>&quot;DUTRNP&quot;</td>
<td>DUT noise power ratio. (Noise power expressed in Kelvin divided by 290).</td>
</tr>
<tr>
<td>&quot;DUTRNPI&quot;</td>
<td>System noise power ratio</td>
</tr>
<tr>
<td>&quot;SYSRNP&quot;</td>
<td>System noise power ratio</td>
</tr>
<tr>
<td>&quot;SYSRNPI&quot;</td>
<td>System noise power ratio</td>
</tr>
<tr>
<td>&quot;DUTNPD&quot;</td>
<td>DUT noise power density. (Noise power expressed in dBm/Hz).</td>
</tr>
<tr>
<td>&quot;DUTNPDI&quot;</td>
<td>System noise power density</td>
</tr>
<tr>
<td>&quot;SYSNPD&quot;</td>
<td>System noise power density</td>
</tr>
<tr>
<td>&quot;SYSNPDI&quot;</td>
<td>System noise power density</td>
</tr>
<tr>
<td>&quot;OvrRng&quot;</td>
<td>Indication that the noise receiver is being over powered.</td>
</tr>
<tr>
<td>&quot;T-Rcvr&quot;</td>
<td>Temperature reading (in Kelvin) of the noise receiver board.</td>
</tr>
</tbody>
</table>

**Noise Figure ONLY - NOT NFX:**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;A_1&quot;,&quot;A_2&quot; ...and so forth.</td>
<td>Unratioed parameters; with notation: &quot;receiver, source port&quot;</td>
</tr>
<tr>
<td>&quot;GammaOpt&quot;</td>
<td>Optimum Complex Reflection Coefficient</td>
</tr>
<tr>
<td>&quot;Rn&quot;</td>
<td>Noise Resistance</td>
</tr>
<tr>
<td>&quot;NFMn&quot;</td>
<td>Minimum noise figure that occurs at GammaOpt</td>
</tr>
</tbody>
</table>

**NFX ONLY:**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;S11&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;SC21&quot;</td>
<td>&quot;SC12&quot;</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>Mixer parameters</td>
<td></td>
</tr>
</tbody>
</table>

| "ALO1"," BLO1" | Test port receiver at LO1 frequency |
| "R1_1", "B_2" | Unratioed parameters with notation: |
| and so forth. | "receiver_source port" |

<table>
<thead>
<tr>
<th>Phase Noise</th>
<th>PN</th>
<th>Phase Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn more</td>
<td>AM</td>
<td>AM Noise</td>
</tr>
</tbody>
</table>

There are over 150 possible Swept IMD parameters, too many to list here.

Build the parameters with the Swept IMD Parameter dialog, then copy the parameter name to the remote command.

The following are a few example parameters:

<table>
<thead>
<tr>
<th>&quot;Swept IMD&quot;</th>
<th>&quot;PwrMainLo&quot;</th>
<th>Absolute power of the Low tone at the DUT output.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>&quot;Swept IMD Converters&quot;</th>
<th>&quot;IM3&quot;</th>
<th>Power of the third product relative to the average power of the f1 and f2 tones measured at the DUT output.</th>
</tr>
</thead>
</table>

<p>| &quot;OIP3&quot; | Theoretical power level at which the third product will be the same power level as the |</p>
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;IM Spectrum&quot;</td>
<td>Average of the main tones at the output of the DUT.</td>
</tr>
<tr>
<td>&quot;Input&quot;</td>
<td>View signals IN to the DUT (R1 receiver).</td>
</tr>
<tr>
<td>&quot;Output&quot;</td>
<td>View signals OUT of the DUT and into VNA port 2 (B receiver).</td>
</tr>
<tr>
<td>&quot;Reflection&quot;</td>
<td>View signals reflected off the DUT input and back into VNA port 1 (A receiver)</td>
</tr>
<tr>
<td>&quot;IMx Spectrum Converters&quot;</td>
<td>View signals OUT of the DUT and into VNA port 2 (B receiver)</td>
</tr>
<tr>
<td>Impedance Measurement</td>
<td>Impedance</td>
</tr>
<tr>
<td>(E5080B Only)</td>
<td>Admittance</td>
</tr>
<tr>
<td>&quot;Z&quot;</td>
<td>Equivalent Series Inductance</td>
</tr>
<tr>
<td>&quot;Y&quot;</td>
<td>Inductance</td>
</tr>
<tr>
<td>&quot;Ls&quot;</td>
<td>Equivalent Parallel Inductance</td>
</tr>
<tr>
<td>&quot;Lp&quot;</td>
<td>Capacitance</td>
</tr>
<tr>
<td>&quot;Cs&quot;</td>
<td>Equivalent Series Capacitance</td>
</tr>
<tr>
<td>&quot;Cp&quot;</td>
<td>Capacitance</td>
</tr>
<tr>
<td>&quot;Rs&quot;</td>
<td>Equivalent Series Resistance</td>
</tr>
<tr>
<td>&quot;Rp&quot;</td>
<td>Resistance</td>
</tr>
<tr>
<td>&quot;Q&quot;</td>
<td>Equivalent Parallel Resistance</td>
</tr>
<tr>
<td>&quot;D&quot;</td>
<td>Q Value (Quality Factor)</td>
</tr>
<tr>
<td>&quot;Differential I/Q&quot;</td>
<td>Create custom parameters using Sens:DIQ:Par:Def,</td>
</tr>
<tr>
<td></td>
<td>Dissipation Factor</td>
</tr>
<tr>
<td>Learn more</td>
<td>then specify your custom parameter name here. The following are default parameters:</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td>&quot;IPwrF1&quot; Input Power over F1 range</td>
</tr>
<tr>
<td></td>
<td>&quot;OPwrF1&quot; Output Power over F1 range</td>
</tr>
<tr>
<td></td>
<td>&quot;GainF1&quot; Gain over F1 range</td>
</tr>
<tr>
<td>&quot;Spectrum Analyzer&quot;</td>
<td>&quot;a&lt;n&gt;&quot; Reference receiver</td>
</tr>
<tr>
<td>Learn more</td>
<td>&quot;b&lt;n&gt;&quot; Test port receiver</td>
</tr>
<tr>
<td></td>
<td>where &lt;n&gt; is the port number to measure</td>
</tr>
<tr>
<td></td>
<td>&quot;ImageReject&lt;n&gt;&quot;</td>
</tr>
<tr>
<td></td>
<td>where &lt;n&gt; is the image reject trace</td>
</tr>
</tbody>
</table>

**Examples**

```
CALC:MEAS:PAR "Sdd11"
calculate2:measure2:parameter "Sdd11"
CALC:MEAS2:DEF 'PN:Phase Noise' 'Defines a Phase Noise measurement but doesn't display.'
DISP:MEAS2:FEED 1 'Displays Phase Noise measurement in window number 1.'
CALC:MEAS2:PAR 'AM' 'Changes the Phase Noise parameter to AM in window number 1.'
```

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:PARameter?

**Return Type**

String

**Default**

"S11"
CALCulate:MEASure:RLIMit Commands

These commands are for setting up ripple tests.

```plaintext
CALCulate:MEASure:RLIMit
   DATA
   DISPlay
      | LINE
      | STATe
      | SELECT
      | TYPE
   FAIL
   REPORT
      | DATA
   STATe
```

Click a keyword to view the command details.

**see Also**

- Learn about Ripple tests
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

CALCulate<cnum>:MEASure<mnum>:RLIMit:DATA <data>
Applicable Models: All

(Read-Write) Sets or returns the ripple limit table for the active trace of selected channel.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<data>` Indicates the array data (for ripple line) of \(1 + \text{Num (number of limit lines)} \times 4\). Where \(n\) is an integer between 1 and Num.
  
  - Data(0) : The number of limit lines you want to set. Specify an integer ranging 0 to 12. When the number of limit lines is set to 0 (clears the limit table), the variable Data is only required with Data(0).
  
  - Data(nx4-3) : The type of the n-th line.

  Specify an integer 0 to 1 as follows.
  
  0: OFF
  
  1: ON

  - Data(nx4-2) : The value on the horizontal axis (frequency/power/time) of the start point of the n-th line.

  - Data(nx4-1) : The value on the horizontal axis (frequency/power/time) of the end point of the n-th line.

  - Data(nx4) : The ripple line value (dB) of the n-th line.

  The index of the array starts from 0.

**Examples**

```
CALC: MEAS: RLIM: DATA
calculate2:measure2:rlimit: data
```

**Query Syntax**
CALCulate<cnun>:MEASure<mnum>:RLIMit:DATA?

**Return Type**
Variant type Array

**Default**
OFF
Applicable Models: All

(Read-Write) Turns ON/OFF the ripple limit line display, for the active trace of selected channel.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` ON or 1 - Turns limit testing ON.
  OFF or 0 - Turns limit testing OFF.

Examples
```
CALC:MEAS:RLIM:DISP:LINE:STAT ON
```
```
calculate2:measure2:rlimit:display:line:state off
```

Query Syntax
```
CALCulate<cnum>:MEASure<mnum>:RLIMit:DISPly:LINE:STATe?
```

Return Type
Boolean

Default
OFF

CALCulate<cnum>:MEASure<mnum>:RLIMit:DISPly:SELect <band>

Applicable Models: All

(Read-Write) Sets or gets the ripple limit band for ripple value display for selected channel.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<band>` 1 to 12

Examples
```
CALC:MEAS:RLIM:DISP:RIPP:SEL
```
```
calculate2:measure2:rlimit:display:ripple:select
```

Query Syntax
```
CALCulate<cnum>:MEASure<mnum>:RLIMit:DISPlay:SELect?
```

Return Type
Numeric

Default
1

CALCulate<cnum>:MEASure<mnum>:RLIMit:DISPlay:TYPE <typ>
Applicable Models: All

(Read-Write) Sets/gets the display type of ripple value for the active trace of selected channel.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<typ>` Select from the following:
  - "OFF": Specifies the display OFF.
  - "ABSolute": Specifies the absolute value for display type.
  - "MARgin": Specifies the margin for display type.

Examples
```
CALC:MEAS:RLIM:DISP:TYPE
```
```
calculate2:measure2:rlimit:display:type
```

Query Syntax
```
CALCulate<cnum>:MEASure<mnum>:RLIMit:DISPLay:TYPE?
```
Return Type
Boolean
Default
OFF

CALCulate<cnum>:MEASure<mnum>:RLIMit:FAIL

Applicable Models: All

(Read-only) Read the ripple test result for the active trace.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` Boolean
  - 0 is returned when Pass
  - 1 is returned when Fail

Examples
```
CALC:MEAS:RLIM:FAIL?
```
```
calculate2:measure2:rlimit:FAIL?
```

Query Syntax
```
CALCulate<cnum>:MEASure<mnum>:RLIMit:FAIL?
```
Return Type
Boolean
Default
Not Applicable

CALCulate<cnum>:MEASure<mnum>:RLIMit:REPort:DATA
Applicable Models: All

(Read-only) Reads the ripple value of the ripple test for the active trace.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<data>` `{numeric 1} ... {numeric NOP×3+1}<newline><^END>`

NOP is the number of measurement points.

- `{numeric 1}`: Number of ripple limit line
- `{numeric n×3-1}`: Number of ripple limit bands.
- `{numeric n×3}`: Ripple value.
- `{numeric n×3+1}`: Ripple test result (1: Fail, 0: Pass)

**Examples**
```
CALC:MEAS:RLIM:REP:DATA?
calculate2:measure2:rlimit:report:data?
```

**Query Syntax**
```
CALCulate<cnum>:MEASure<mnum>:RLIMit:STATe?
```

**Return Type** Variant

**Default** OFF

CALCulate<cnum>:MEASure<mnum>:RLIMit:STATe <bool>

Applicable Models: All

(Read-Write) Turns ON/OFF the ripple test function for the active trace of selected channel.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` ON (or 1) - turns limit testing ON.
  OFF (or 0) - turns limit testing OFF.

**Examples**
```
CALC:MEAS:RLIM:STAT ON
calculate2:measure2:rlimit:state off
```

**Query Syntax**
```
CALCulate<cnum>:MEASure<mnum>:RLIMit:STATe?
```

**Return Type** Boolean

**Default** OFF
CALCulate:MEASure:SA Commands

Controls the marker settings used in the SA application.

<table>
<thead>
<tr>
<th>CALCulate:MEASure:SA:MARKer:</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAND</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>BDENsity</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>BPOWer</td>
</tr>
</tbody>
</table>
Click on a keyword to view the command details.

See Also

- Marker Readout number and size commands.
- Learn about Markers
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**Important:** Learn about programming the reference marker.

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BAND:FUNCTION <enum>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xA/B, S9x090A/B) or N524xB models with Option S93070xB

*(Read-Write)* Set and read the SA marker function type.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<enum>` Choose from:
  - MNOI - Marker noise.
  - BPOW - Band power.
  - BDEN - Band density.
  - OFF

<table>
<thead>
<tr>
<th>Examples</th>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
</table>

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BAND:IBW <num>
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B) or N524xB models with Option S93070xB

(Read-Write) Sets and reads the integration bandwidth marker. The IBW is used to determine total signal power within a specified frequency span. For example, to calculate the total power of a signal composed of 100 tones spaced 1 MHz apart over a 100 MHz span, the signal power would be integrated over an IBW of 100 MHz.

Parameters
- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<num>` Choose an integration bandwidth.

Examples

```
CALC:MEAS2:SA:MARK:BAND:IBW 100e6
```

Query Syntax
```
CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BAND:IBW?
```

Return Type: Numeric

Default: Not Applicable


Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B) or N524xB models with Option S93070xB

(Read-Write) Sets and reads the ACPR density marker.

Parameters
- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - Turn band density marker OFF.
  - 1 - ON - Turn band density marker ON.

Examples
```
'Select the measurement
CALC2:PAR:SEL "M2SA_CH2_A"

'Create marker3 on that measurement
```
CALC2:MARK3 ON
'Make it a band density noise marker

**Query Syntax**  CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:ACPR[:STATe]?

**Return Type**  Boolean

**Default**  0

**CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:BW <num>**

**Applicable Models:** All with Spectrum Analysis Options (59x09xxA/B, 59x090A/B)

(Read-Write) Sets and reads the bandwidth of the band density marker.

**Parameters**

- `<ch>`  Channel number of the measurement (optional).
- `<mnum>`  Measurement number for each measurement.
- `<n>`  Marker number. If unspecified, `<n>` is set to 1.
- `<num>`  Choose a bandwidth.

**Examples**  CALC:MEAS2:SA:MARK:BDEN:BW 1e6

**Query Syntax**  CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:BW?

**Return Type**  Numeric

**Default**  1 MHz

**CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:DATA?**
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-only) Returns the band density level in dBm/Hz from the band density marker.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Band density marker number. If unspecified, `<n>` is set to 1.

Examples

```
CALC:MEAS2:SA:MARK:BDEN:DATA?
calculate2:measure2:sa:marker2:bdensity:data?
```

Return Type Numeric

Default Not applicable

---

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:EQSPan <num>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-Write) Sets and reads the frequency span used by Power Density to normalize the power.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<num>` Choose a span.

Examples

```
CALC:MEAS:SA:MARK:BDEN:EQSPan 1e6
```

Query Syntax CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:EQSPan?

Return Type Numeric

Default 1 MHz

---

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:NOISe[:STATe] <bool>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

*(Read-Write)* Sets and reads the state of the band density noise marker.

**Parameters**

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Turn band density noise marker OFF.
  - **1 - ON** - Turn band density noise marker ON.

**Examples**

'Select the measurement

```
CALC2:PAR:SEL "M2SA_CH2_A"
```

'Create marker3 on that measurement

```
CALC2:MARK3 ON
```

'Make it a band density noise marker

```
CALC:MEAS:SA:MARK:BDEN:NOIS:STAT 1
```

**Query Syntax**

```
CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:NOISe?
```

**Return Type**

Boolean

**Default**

0

```
```
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B) or N524xB models with Option S93070xB

(Read-Write) Sets and reads the NPR density marker.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - Turn band density marker OFF.
  - 1 - ON - Turn band density marker ON.

Examples

- 'Select the measurement
  CALC2:PAR:SEL "M2SA_CH2_A"

- 'Create marker3 on that measurement
  CALC2:MARK3 ON

- 'Make it a band density noise marker
  CALC2:MEAS:SA:MARKer<n>:BDENsity:NPR:STAT 1

Query Syntax

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:NPR[:STATe]?

Return Type

Boolean

Default

0

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:POWer[:STATe] <bool>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

*(Read-Write)* Sets and reads the state of the band power density marker.

**Parameters**

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<bool>` Choose from:
  - 0 - **OFF** - Turn band power density marker OFF.
  - 1 - **ON** - Turn band power density marker ON.

**Examples**

'Select the measurement

CALC2:PAR:SEL "M2SA_CH2_A"

'Create marker3 on that measurement

CALC2:MARK3 ON

'Make it a band density noise marker

CALC:MEAS:SA:MARK:BDEN:POW:STAT 1

**Query Syntax**

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:POWer?

**Return Type**

Boolean

**Default**

0

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:POWer:BW <num>
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-Write) Sets and reads the bandwidth of the band power density marker.

**Parameters**
- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<num>` Choose a bandwidth.

**Examples**
```
```

**Query Syntax**
```
CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:POWer:BW?
```

**Return Type** Numeric

**Default** 1 MHz

---

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:TONE:BW <num>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-Write) Sets and reads the bandwidth of the band tone density marker.

**Parameters**
- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<num>` Choose a bandwidth.

**Examples**
```
```

**Query Syntax**
```
CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:TONE:BW?
```

**Return Type** Numeric

**Default** 1 MHz

---

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:TONE[:STATe] <bool>
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-Write) Sets and reads the state of the band tone density marker.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Turn band tone density marker OFF.
  - **1 - ON** - Turn band tone density marker ON.

Examples

```
'Select the measurement
CALC2:PAR:SEL "M2SA_CH2_A"

'Create marker3 on that measurement
CALC2:MARK3 ON

'Make it a band density noise marker
CALC:MEAS:SA:MARK:BDEN:TONE:STAT 1
```

Query Syntax

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:TONE?

Return Type

Boolean

Default

0

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:TONE:TSPacing <num>
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-Write) Sets and reads the spacing of the band tone density marker.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<num>` Choose a spacing value.

Examples

```
```

Query Syntax

```
CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BDENsity:TONE:TSPacing?
```

Return Type Numeric

Default 100 MHz

---

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BPOWer:DATA?

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-only) Returns the band power level from the band power marker.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Band power marker number.

Examples

```
CALC:MEAS2:SA:MARK:BPOWER:DATA?
calculate2:measure2:sa:marker2:bpower:data?
```

Default Not applicable

---

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BPOWer:SPAN <num>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xA/B, S9x090A/B)

*(Read-Write)* Sets and reads the frequency span of the band power marker. This area is marked by two vertical dotted lines on the screen and the marker's y-axis value is set to the measured power value. Noise and power on the same marker share the same span.

**Parameters**
- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number.
- `<num>` Choose a frequency span within the frequency range of the analyzer.

**Examples**
```
CALC:MEAS2:SA:MARK:BPOW:SPAN 1e6
```

**Query Syntax**
```
CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BPOWer:SPAN?
```

**Return Type** Numeric

**Default** 1 MHz

---

**CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BPOWer[:STATe] <bool>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xA/B, S9x090A/B)

*(Read-Write)* Sets and reads the state of the band power marker. This command makes a band power marker from a generic marker. The generic marker must first be created using:

```
CALC:MEAS:MARK:STATe
```

**Parameters**
- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number.
- `<bool>` Choose from:
  - 0 - OFF - Turn band power marker OFF.
  - 1 - ON - Turn band power marker ON.

**Examples**
```
'Create marker3 on the specified measurement
CALC2:MEAS2:MARK3 ON

'Make it a band power marker
CALC:MEAS2:SA:MARK:BPOW:STAT 1
```
CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:BPOWer?

Return Type: Boolean
Default: 0

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:OCCBand:CENTer?

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-only) Returns the occupied bandwidth center frequency.

Parameters
- `<ch>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
- `<n>`: Marker number.

Examples
- `CALC:MEAS:SA:MARK:OCCB:CENT?`

Default: Not applicable

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:OCCBand:PERCent <num>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-Write) Sets and returns the percentage of the band power to search for.

Parameters
- `<ch>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
- `<n>`: Marker number.
- `<num>`: Percentage value.

Examples

Query Syntax: CALCulate<ch>:SA:MARKer<n>:OCCBand:PERCent?
Return Type: Numeric
Default: 99.0
CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:OCCBand:POWer?

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

*(Read-only)* Returns the occupied bandwidth power.

**Parameters**
- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number.

**Examples**
- `CALC:MEAS:SA:MARK:OCCB:POW?`
- `calculate2:measure:sa:marker2:occband:power?`

**Default** Not applicable

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:OCCBand:SPAN?

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

*(Read-only)* Returns the occupied bandwidth span.

**Parameters**
- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number.

**Examples**
- `CALC:MEAS:SA:MARK:OCCB:SPAN?`

**Default** Not applicable

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:OCCBand[:STATe] <bool>
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-Write) Sets and returns the occupied bandwidth on/off state.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number.
- `<bool>` Choose from:
  - **0 - OFF** - Turns occupied bandwidth OFF.
  - **1 - ON** - Turns occupied bandwidth ON.

Examples

```
CALC:MEAS:SA:MARK:OCCB:STAT 1
```

Query Syntax

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:SEARch:ACPR

Default

- **0**

Note: If occupied band state is turned ON, then Band Power or Band Noise is turned OFF.

---

CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:SEARch:ACPR

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Write-only) Executes the search ACPR density marker.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.

Examples

```
```

Query Syntax

Not Applicable

Default

- Not Applicable
CALCulate<ch>:MEASure<mnum>:SA:MARKer<n>:SEARch:NPR

Applicable Models: All with Spectrum Analysis Options (S9x09xA/B, S9x090A/B)

(Write-only) Executes the search NPR density marker.

Parameters

<ch> Channel number of the measurement (optional).

<mnum> Measurement number for each measurement.

<n> Marker number. If unspecified, <n> is set to 1.

Examples

CALC:MEAS:SA:MARK:SEAR:NPR


Query Syntax Not Applicable

Default Not Applicable
CALCulate:MEASure:SMOothing Commands

Controls point-to-point smoothing. Smoothing is a noise reduction technique that averages adjacent data points in a measurement trace. Choose the amount of smoothing by specifying either the number of points or the aperture. Smoothing is not the same as CALC:AVERage which averages each data point over a number of sweeps.

```
CALCulate:MEASure:SMOothing
  APERture
  POINts
  [:STATe]
```

Click a keyword to view the command details.

See Also

- Learn about Smoothing
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

CALCulate<cnum>:MEASure<mnum>:SMOothing:APERture <num>

**Applicable Models:** All

*(Read-Write)* Sets the amount of smoothing as a percentage of the number of data points in the channel.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Percentage value. Choose any number between: 1 and 25

**Examples**
- `CALC:MEAS:SMO:APER 2`
- `calculate2:measure2:smoothing:aperture 20.7`

**Query Syntax**
- `CALCulate<cnum>:MEASure<mnum>:SMOothing:APERture?`

**Return Type**
- Numeric
- Default: 1.5

CALCulate<cnum>:MEASure<mnum>:SMOothing:POINts <num>
Applicable Models: All

([Read-Write]) Sets the number of adjacent data points to average.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Number of points from 1 point to maximum of 25% of data points in the channel. For example: if number of points in a data trace = 401, the maximum value for points = 100. The points value is always rounded to the closest odd number.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS:SMO:POIN 50</td>
<td>calculate2:measure2:smoothing:points 21</td>
</tr>
</tbody>
</table>

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:SMOothing:POINts?

**Return Type**

Numeric

**Default**

3

CALCulate<cnum>:MEASure<mnum>:SMOothing[:STATe] <bool>

Applicable Models: All

([Read-Write]) Turns data smoothing ON or OFF.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` ON or 1 - Turns smoothing ON. OFF or 0 - Turns smoothing OFF.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS:SMO ON</td>
<td>calculate2:measure2:smoothing:state off</td>
</tr>
</tbody>
</table>

**Query Syntax**

CALCulate<cnum>:MEASure<mnum>:SMOothing[:STATe]

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF
### CALCulate:MEASure:TRANsform Commands

Specifies the settings for time domain transform.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COUPle</td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Click a keyword to view the command details.

#### See Also

- Learn about Time Domain
- Synchronizing the Analyzer and Controller
CALCulate<cnum>:MEASure<mnum>:TRANsform:COUPle:PARameters <num>

Applicable Models: All

(Read-Write) Specifies the time domain transform parameters to be coupled. The settings for those parameters will be copied from the selected measurement to all other measurements on the channel.

- To turn coupling ON and OFF, use SENS:COUP:PAR
- To specify Gating parameters to couple, use CALC:MEAS:FILT:COUP:PAR

Learn more about Time Domain Trace Coupling

Parameters
- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <num> (Numeric) Parameters to couple. To specify more than one parameter, add the numbers.
  1 - Transform Stimulus (Start, Stop, Center, and Span TIME settings.)
  2 - Transform State (ON / OFF)
  4 - Transform Window (Kaiser Beta / Impulse Width)
  8 - Transform Mode (Low Pass Impulse, Low Pass Step, Band Pass)
  16 - Transform Distance Marker Units

Examples
- To couple all parameters:
  
  ```
  CALC:MEAS:TRAN:COUP:PAR 31
  ```
  
- To couple Stimulus and Mode:
  
  ```
  calculate2:measure2:transform:couple:parameters 9
  ```

Query Syntax
CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:ALIgnment <enum>

Return Type
Numeric

Default
29 (All parameters except 2 - Transform State)
Applicable Models: All

(Read-Write) Sets the way the PNA computes the DC value of the frequency-domain measurement. The correct DC value is required for inverse-FFT accuracy, and if not estimated properly, can cause distortions in the time-domain measurement in the form of an undesired slope in the waveform.

Parameters

\(<\text{cnum}>\) Channel number of the measurement (optional).
\(<\text{mnum}>\) Measurement number for each measurement.
\(<\text{enum}>\) Choose from:

**LEGacy** - The DC value is extrapolated using three data points. The transform offset is calculated using the delay of the first frequency point. This is the same algorithm used in the HP 8510 network analyzer.

**NORMalize** - The DC value is extrapolated using three data points. The transform offset is set to zero at \(t=0\) minus six rise-times. This mode requires that a good S-parameter calibration has been performed, which can be verified by observing a flat time-domain response at \(t=0\) when measuring a load located at the physical point corresponding to \(t=0\). Normalize mode is principally used to help stabilize the time-domain trace at time \(t=0\) to 50 ohms, to remove bouncing of the response at \(t=0\). This method is similar to that used with PLTS, and is very useful in determining the time-domain-transform response of transmission lines and printed-circuit-board characteristics.

Examples

```
CALC:MEAS:TRAN:TIME:ALIG NORM
calculate2:measure2:transform:time:alignment?
```

Return Type

Enumeration

Default

NORMalize

CALCulate\(<\text{cnum}>\):MEASure\(<\text{mnum}>\):TRANsform:TIME:CENTer \(<\text{num}>\)
Applicable Models: All

(Read-Write) Sets the center time for time domain measurements.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Center time in seconds; any number between: ± (number of points-1) / frequency span

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples
```
CALC:MEAS:TRAN:TIME:CENT 1e-8
calculate2:measure2:transform:time:center 15 ps
```

Query Syntax
```
CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:CENTer?
```

Return Type
Numeric

Default
0

\textbf{CALCulate}<cnum>:MEASure<mnum>:TRANsform:TIME:CLIP <bool> \textbf{Applicable Models: All}

(Read-Write) Turns the start/stop time limits ON or OFF. When ON, limits the start/stop times to avoid aliasing.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` ON (or 1) - turns start/stop time limits ON.
  OFF (or 0) - turns start/stop time limits OFF.

Examples
```
CALC:MEAS:TRAN:TIME:CLIP ON
calculate2:measure2:transform:time:clip off
```

Query Syntax
```
CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:CLIP?
```

Return Type
Boolean (1 = ON, 0 = OFF)

Default
ON

\textbf{CALCulate}<cnum>:MEASure<mnum>:TRANsform:TIME:IMPulse:WIDTh <num>
Applicable Models: All

(Read-Write) Sets the impulse width for the transform window.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Impulse width in seconds; Choose any number between: .6 / frequency span and 1.39 / frequency span

Examples
- CALC:MEAS:TRAN:TIME:IMP:WIDTh 10

Query Syntax
- CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:IMPulse:WIDTh?

Return Type
- Numeric
- Default: .98 / Default Span

CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:KBESsel <num>

Applicable Models: All

(Read-Write) Sets the parametric window for the Kaiser Bessel window.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Window width for Kaiser Bessel in seconds; Choose any number between: 0.0 and 13.0

Examples
- CALC:MEAS:TRAN:TIME:KBES 10
- calculate2:measure2:transform:time:kbessel 13

Query Syntax
- CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:KBESsel?

Return Type
- Numeric
- Default: 6

CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:LPFREQuency
(Write-only) Sets the start frequencies in LowPass Mode.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

Examples
```
CALC:MEAS:TRAN:TIME:LPFR
```
```
calculate2:measure2:transform:time:lpfrequency
```

Query Syntax Not Applicable
Default Not Applicable

---

**CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:MARKer:MODE <char>**

Applicable Models: All

(Read-Write) Specifies the measurement type in order to determine the correct marker distance.

- Select Auto for S-Parameter measurements.
- Select Reflection or Transmission for arbitrary ratio or unratioed measurements.

This setting affects the display of ALL markers for only the ACTIVE measurement.

Learn more about Distance Markers.

Parameters
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<char>` Choose from:
  - AUTO - If the active measurement is an S-Parameter, automatically chooses reflection or transmission. If non S-Parameter measurements, reflection is chosen.
  - REFLection - Displays the distance from the source to the receiver divided by two (to compensate for the return trip.)
  - TRANsmission - Displays the distance from the source to the receiver.

Examples
```
CALC:MEAS:TRAN:TIME:MARK:MODE REFL
```
```
calculate2:measure2:transform:time:marker:mode auto
```

Query Syntax
```
CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:MARKer:MODE?
```

Return Type Character
Default AUTO
CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:MARKer:UNIT <char>

**Applicable Models:** All

*(Read-Write)* Specifies the unit of measure for the display of marker distance values. This setting affects the display of ALL markers for only the ACTIVE measurement (unless Distance Maker Units are coupled using CALC:MEAS:TRAN:COUP:PAR).

Learn more about [Distance Markers](#).

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<char>` Choose from:
  - METRs
  - FEET
  - INCHes

**Examples**
- `calculate2:measure2:transform:time:marker:unit feet`

**Query Syntax**
- `CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:MARKer:UNIT?`

**Return Type**
- Character

**Default**
- METRs

---

CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:SPAN <num>

**Applicable Models:** All

*(Read-Write)* Sets the span time for time domain measurements.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Span time in seconds; any number between: 0 and 2 * [(number of points-1) / frequency span]

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See [SCPI Syntax](#) for more information.

**Examples**
- `CALC:MEAS:TRAN:TIME:SPAN 1e-8`
- `calculate2:measure2:transform:time:span maximum`

**Query Syntax**
- `CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:SPAN?`
CALCulate\(<cnum>\):MEASure\(<mnum>\):TRANsform:TIME:STARt \(<num>\)

**Applicable Models:** All

*(Read-Write)* Sets the start time for time domain measurements.

**Parameters**

- \(<cnum>\) Channel number of the measurement (optional).
- \(<mnum>\) Measurement number for each measurement.
- \(<num>\) Start time in seconds; any number between:
  \[\pm \frac{(\text{number of points}-1)}{\text{frequency span}}\]

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate(&lt;cnum&gt;):MEASure(&lt;mnum&gt;):TRANsform:TIME:STARt?</td>
</tr>
</tbody>
</table>

**Return Type** Numeric

**Default** 20 ns

---

CALCulate\(<cnum>\):MEASure\(<mnum>\):TRANsform:TIME:STATe \(<bool>\)

**Applicable Models:** All

*(Read-Write)* Turns the time domain transform capability ON or OFF.

**Note:** *Sweep type* must be set to Linear Frequency in order to use Time Domain Transform.

**Parameters**

- \(<cnum>\) Channel number of the measurement (optional).
- \(<mnum>\) Measurement number for each measurement.
- \(<bool>\) ON (or 1) - turns time domain ON.
  OFF (or 0) - turns time domain OFF.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MEAS:TRAN:TIME:STAT ON</td>
</tr>
<tr>
<td>calculate2:measure2:transform:time:stat off</td>
</tr>
</tbody>
</table>

**Return Type** Boolean (1 = ON, 0 = OFF)
CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:STEP:RTIMe <num>

Applicable Models: All

(Read-Write) Sets the step rise time for the transform window.

Parameters
- <cnum>: Channel number of the measurement (optional).
- <mnum>: Measurement number for each measurement.
- <num>: Rise time in seconds; Choose any number between:
  - 0.45 / frequency span and 1.48 / frequency span

Examples
- calculate2:measure2:transform:time:step:rtime 15 ps

Query Syntax
- CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:STEP:RTIMe?

Return Type
- Numeric
- Default: 0.99 / Default Span

CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:STOP <num>

Applicable Models: All

(Read-Write) Sets the stop time for time domain measurements.

Parameters
- <cnum>: Channel number of the measurement (optional).
- <mnum>: Measurement number for each measurement.
- <num>: Stop time in seconds; any number between:
  - ± (number of points-1) / frequency span

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples
- CALC:MEAS:TRAN:TIME:STOP 1e-8
- calculate2:measure2:transform:time:stop maximum

Query Syntax
- CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME:STOP?

Return Type
- Numeric
- Default: 10 ns

CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME[:TYPE] <char>
**Applicable Models:** All

(Read-Write) Sets the type of time domain measurement.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<char>` Type of measurement. Choose from:
  - **BPASs** - Set transform mode to band pass.
  - **LPSTep** - Set transform mode to low pass step.
  - **LPIMpulse** - Set transform mode to low pass impulse.

**Examples**

Learn about these settings.

```
CALC:MEAS:TRAN:TIME BPAS
```

```
calculate2:measure2:transform:time:type bpas
```

**Query Syntax**

```
CALCulate<cnum>:MEASure<mnum>:TRANsform:TIME[:TYPE]?
```

**Return Type**

Character

**Default**

BPAS
CALCulate:MEASure:X Commands

Controls the display of X-axis for various measurements.

```
CALCulate:MEASure:X
  AXIS
    | DOMain
    | FIXed
    | PARameter
    | VALue
    | UNIT?
[:,VALues]
```

Click a keyword to view the command details.

See Also

- Synchronizing the Analyzer and Controller
- SCPI Command Tree

CALCulate<ch>:MEASure<mnum>:X:AXIS <string>

**Applicable Models:** All

(Write-Read) Sets the X-axis of the selected measurement. This command does not change the default setting for new traces.

**Parameters**

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<string>` String - (Not case-sensitive) For all channels EXCEPT DIQ, choose from the following:
  
  - "Default" - The default X-axis setting for the selected measurement. For Application measurements, the X-Axis domain is set with specific commands.
  
  - "AO1" - Internal DC source #1
"AO2" - Internal DC source #2

**Note:** For DIQ channels, see `CALC:MEAS:X:AXIS:DOMain`

For Modulation Distortion Channels (MOD), choose from:

"**SA Frequency**" - SA display showing the SA frequency settings.

"**Power In**" - Displays input power sweep.

"**Power Out**" - Displays output power sweep.

"**Measured Carrin**" - Measured input band power.

"**Measured CarrOut**" - Measured output band power.

For Modulation Distortion Converters Channels (MODX), choose from:

"**SA Freq In**" - SA display showing mixer input range.

"**SA Freq Out**" - SA display showing mixer output range.

"**Power In**" - Displays input power sweep.

"**Power Out**" - Displays output power sweep.

"**Measured Carrin**" - Measured input band power.

"**Measured CarrOut**" - Measured output band power.

**Examples**

```plaintext
CALC:MEAS2:X:AXIS 'Default'
calculate:measure2:x:axis "AO1"
```

**Query Syntax**

`CALCulate<ch>:MEASure<mnum>:X:AXIS?`

**Return Type**

String

**Default**

"Default"

`CALCulate<ch>:MEASure<mnum>:X:AXIS:DOM ain <string>`
Applicable Models: All

(Write-Read) Sets and returns the X-Axis domain of the selected DIQ measurement.

Parameters

<ch> Channel number of the measurement (optional).
<mnum> Measurement number for each measurement.

<table>
<thead>
<tr>
<th>Choose one of these:</th>
<th>Then set X-Axis Source (CALC:MEAS:X:AXIS) using one of these as the argument.</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Frequency&quot;</td>
<td>&quot;F1&quot;, &quot;F2&quot;, etc.</td>
</tr>
<tr>
<td>&quot;Power&quot;</td>
<td>Source port: &quot;Port 1&quot;, &quot;Port 2&quot;, etc.</td>
</tr>
<tr>
<td>&quot;Phase&quot;</td>
<td>Source port: &quot;Port 1&quot;, &quot;Port 2&quot;, etc.</td>
</tr>
<tr>
<td>&quot;DC&quot;</td>
<td>DC Source: &quot;AO1&quot;, &quot;AO2&quot;</td>
</tr>
<tr>
<td>&quot;Points&quot;</td>
<td>&quot;Points&quot;</td>
</tr>
</tbody>
</table>

Example

1. CALC:MEAS2:X:AXIS:DOM "Power"
2. CALC:MEAS2:X:AXIS "Port 1"

Query Syntax

CALCulate<ch>:MEASure<mnum>:X:AXIS:DOMAIN?

Return Type

String

Default

CALC:MEAS:X:AXIS:DOMAIN: "Frequency"
CALC:MEAS:X:AXIS: "F1"

CALCulate<ch>:MEASure<mnum>:X:AXIS:FIXed:PARameter <string>
Applicable Models: N524xB models with Option S93070xB

(Write-Read) Sets the X-axis fixed parameter for a power sweep measurement in a Modulation Distortion channel.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<string>` String - (Not case-sensitive) For Modulation Distortion Channels (MOD), choose from:
  - "SA Frequency" - SA display showing the SA frequency settings.

For Modulation Distortion Converters Channels (MODX), choose from:
  - "SA Freq In" - SA display showing mixer input range.
  - "SA Freq Out" - SA display showing mixer output range.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>calculate:measure2:x:axis:fixed:parameter &quot;SA Freq In&quot;</td>
<td></td>
<td>&quot;SA Frequency&quot;</td>
</tr>
</tbody>
</table>

Applicable Models: N524xB models with Option S93070xB

(Write-Read) Sets the X-axis fixed parameter spectrum analyzer frequency value for a power sweep measurement in a Modulation Distortion channel.

**Parameters**

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<num>` Fixed parameter value.

**Examples**

```
calc:meas2:x:axis:fix:par:val 1.5e9
```

**Query Syntax**

```
```

**Return Type**

- **Numeric**
- **Default** Not Applicable

---

CALCulate<ch>:MEASure<mnum>:X:AXIS:UNIT?

**Applicable Models:** All

(Read-only) Returns the current units of the X-axis (FREQuency | POWer | PHASe | DC | POINts | DEFault).

This command can be used for all Measurement Classes.

**Parameters**

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

**Examples**

```
CALC:MEAS2:X:AXIS:UNIT?
```

**Return Type**

- **Enumeration**
- **Default** FREQuency

---

CALCulate<ch>:MEASure<mnum>:X:[VALues]?
Applicable Models: All

(Read-only) Returns the stimulus values for the selected measurement in the current units.

This command can be used for all Measurement Classes.

Note: To avoid frequency rounding errors, specify FORM:DATA <Real,64> or <ASCii, 0>

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.

Examples

1. Calc:MEAS2:Par:Sel "MyGCATrace"
2. CALC:MEAS2:X?

Return Type

Depends on FORM:DATA command

Default Not applicable
CALCulate:Measure:UNCertainty Commands

Controls the trace settings for Dynamic Uncertainty for S-Parameters.

<table>
<thead>
<tr>
<th>CALCulate:MEASure:UNCertainty</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISPlay</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>MODE</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>SAVE</td>
</tr>
</tbody>
</table>

See Also

- See other Dynamic Uncertainty commands
- Learn more about Dynamic Uncertainty

CALCulate<cnum>:MEASure<mnum>:UNCertainty:DISPlay:CFACtor <value>

Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the coverage factor (sigma) value to apply to the displayed uncertainty for the selected measurement trace. Coverage Factor corresponds to the level of confidence used in computing the specified measurement uncertainties.

Parameters

- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <value> Integer. Choose from: 

### Coverage Factor and Approximate Confidence Level

<table>
<thead>
<tr>
<th>Coverage Factor</th>
<th>Approximate confidence level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>67%</td>
</tr>
<tr>
<td>2</td>
<td>95%</td>
</tr>
<tr>
<td>3</td>
<td>99%</td>
</tr>
<tr>
<td>4</td>
<td>&gt;99%</td>
</tr>
</tbody>
</table>

### Examples

```
CALC1:MEAS:UNC:DISP:CFAC 2
```

### Query Syntax

`CALCulate<cnum>:MEASure<mnum>:UNCertainty:DISPlay:CFACtor?`

### Return Type

Numeric

### Default

2

### Calculation<cnm>:MEASure<mnm>:UNCertainty:DISPlay:TYPE <char>

**Applicable Models:** N522xB, N523xB, N524xB

*(Write-Read)* Sets and returns the display type for uncertainties for the selected measurement trace.

**Parameters**

- `<cnum>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
- `<char>`: Display type. Choose from:
  - **NONE** - Display the trace without uncertainties.
  - **MAXimum** – Display the trace as the uncertainty maximum (measured or memory data + upper limit uncertainty values). Not supported with Smith Chart or Polar display format.
  - **MINimum** - Display the trace as the uncertainty minimum (measured or memory data - lower limit uncertainty values). Not supported with Smith Chart or Polar display format.
  - **BAR** – Display the uncertainties as “error bars” around the trace. Not supported with Smith Chart or Polar display format.
  - **SHADe** – Display the uncertainties as a shaded region around the trace. Not supported with Smith Chart or Polar display format.
  - **ELLipse** – Display the uncertainties in ellipse form. Supported only in Smith Chart or Polar display format.
CALCulate<cnum>:MEASure<mnum>:UNCertainty:DISPlay:TYPE?  

Return Type: Character  
Default: NONE

**CALCulate<cnum>:MEASure<mnum>:UNCertainty:MODE:CABLe:REPeat <bool>**

**Applicable Models:** N522xB, N523xB, N524xB

*(Write-Read)* Sets and returns whether the cable/connection repeatability contribution is currently included in the uncertainty values for the selected measurement trace.

**Parameters**
- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` Choose from:
  - OFF (or 0) - Cable repeatability is NOT included.
  - ON (or 1) - Cable repeatability IS included.

**Examples** CALC1:MEAS:UNC:MODE:CABL:REP ON

**Query Syntax** CALCulate<cnum>:MEASure<mnum>:UNCertainty:MODE:CABLE:REPeat?  
**Return Type** Boolean  
**Default** ON

CALCulate<cnum>:MEASure<mnum>:UNCertainty:MODE:ETERm <bool>
Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns whether the uncertainties associated with the correction error terms are being included in the uncertainty values for the selected measurement trace.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` Choose from:
  - OFF (or 0) - Error term uncertainty contribution is NOT included.
  - ON (or 1) - Error term uncertainty contribution IS included.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC1:MEAS:UNC:MODE:ETERm?</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

**Default** ON

---

CALCulate<cnum>:MEASure<mnum>:UNCertainty:MODE:NOISe <bool>

Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns whether the noise contribution is currently included in the uncertainty values for the selected measurement trace.

**Parameters**

- `<cnum>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` Choose from:
  - OFF (or 0) - Noise contribution is NOT included.
  - ON (or 1) - Noise contribution IS included.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC1:MEAS:UNC:MODE:NOISe?</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

**Default** ON
CALCulate<cnum>:MEASure<mnum>:UNCertainty:SAVE <"x,y,z">,<filename>

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Saves all three uncertainty data formats from the active measurement for the specified ports.

- This command is valid ONLY with Dynamic Uncertainty Calibration applied to the active measurement.
- Data that is not available is zero-filled.
- For sweeps with a large number of data points, always follow this command with *OPC? Learn more.

Parameters
- <cnum> Channel number of the measurement (optional).
- <mnum> Measurement number for each measurement.
- <"x,y,z"> String Comma or space delimited port numbers for which data is requested, enclosed in quotes.
- <filename> String Path, filename, and suffix of location to store the uncertainty data, enclosed in quotes. The suffix is not checked for accuracy.

- (*.u*p) S-parameter Uncertainty File - If saving 2 ports, specify "filename.u2p"; If saving 4 ports, specify "filename.u4p.", and so forth.
- (*.dsd) S-parameter Data Standard Definition file - Data for Only one or two ports is allowed.
- (*.sdatcv) METAS S-parameter Covariance File.

Examples
All three examples save the active uncertainty measurement on channel 1
'Saves ports 2 and 3 to .u2p file
CALC1:MEAS:UNC:SAVE "2,3","C:\myData.u2p"
'Saves ports 1 and 2 to .dsd file
CALC1:MEAS:UNC:SAVE "1,2","C:\myData.dsd"
'Saves ports 1 through 4 to .sdatcv file
CALC1:MEAS:UNC:SAVE "1,2,3,4","C:\myData.sdatcv"

Query Syntax Not Applicable
Default  Not Applicable
**Calculate:Mixer Command**

This command is **Superseded** by the `CALCulate:MEASure:MIXer:XAXis` command.

---

CALCulate<ch>:MIXer:XAXis <char>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets or returns the swept parameter to display on the X-axis for the selected FCA and GCX measurement.

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using `Calc:Par:MNUM` or `Calc:Par:Select`. Learn more.

**Parameters**

- `<ch>`  Any existing channel number. If unspecified, value is set to 1.
- `<char>` Parameter to display on the X-axis. Choose from:
  - **INPUT**  - Input frequency span
  - **OUTPUT**  - Output frequency span
  - **LO_1**  - First LO frequency span
  - **LO_2**  - Second LO frequency span

**Examples**

```
CALC:MIX:XAX INPUT
calc2:mixer:xaxis output
```

See an example that creates, selects, and calibrates an SMC and VMC measurement using SCPI.

**Query Syntax**  CALCulate<ch>:MIXer:XAXis?

**Return Type**  Character

- **Default**  OUTPUT
Calculate:Normalize Commands

Specifies the normalization features used for a receiver power calibration.

These commands are Superseded (Sept 2004).

See the replacement commands in a new Receiver Power Cal example.

CALCulate: NORMalize

[IMMediate]  STATe  INTerpolation  [STATe]

Click on a keyword to view the command details.

See Also

- Example Programs
- Learn about Receiver Cal
- SCPI Command Tree

Save and recall your receiver power calibration (which use .CST file commands):

- SENS:CORR:CSET:SAVE
- SENS:CORR:CSET[:SEL]

Or use these two commands and specify either .STA or .CST file extensions:

- MMEM:LOAD
- MMEM:STOR

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select

CALCulate<cnum>:NORMalize[:IMMediate]  Superseded
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Note: This command is replaced with SENS:CORR:COLL:METH RPOWer and SENS:CORR:COLL[:ACQ] POWer

See an example of a Receiver Power Calibration.

(Write only) Stores the selected measurement’s data to that measurement’s “divisor” buffer for use by the Normalization data processing algorithm. This command is not compatible with ratioed measurements such as S-parameters. It is intended for receiver power calibration when the selected measurement is of an unratioed power type.

See Critical Note

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
</tbody>
</table>

Examples

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:NORM</td>
</tr>
<tr>
<td>calculate1:normalize:immediate</td>
</tr>
</tbody>
</table>

Query Syntax

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

Default

<table>
<thead>
<tr>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

CALCulate<cnum>:NORMalize:STATe <ON | OFF>  Superseded

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Note: This command is replaced with SENS:CORR[:STATe] ON|OFF

(Read-Write) Specifies whether or not normalization is applied to the measurement. Normalization is enabled only for measurements of unratioed power where it serves as a receiver power calibration.

See Critical Note

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
<tr>
<td>&lt;ON</td>
<td>OFF&gt;</td>
</tr>
</tbody>
</table>

Examples

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:NORM:STAT ON</td>
</tr>
<tr>
<td>calculate2:normalize:state off</td>
</tr>
</tbody>
</table>

Query Syntax

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate&lt;cnum&gt;:NORMalize:STATe?</td>
</tr>
</tbody>
</table>
CALCulate<cnm>:NORMalize:INTerpolate[:STATe] <ON | OFF>  **Superseded**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

**Note:** This command is replaced with SENS:CORR:INT[:STATe] ON|OFF

(Read-Write) Turns normalization interpolation ON or OFF. Normalization is enabled only for measurements of unratioed power, where it serves as a receiver power calibration.

See Critical Note

**Parameters**

- `<cnm>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnm>` is set to 1.
- `<ON | OFF>` **ON (or 1)** – turns interpolation ON.
  
  **OFF (or 0)** – turns interpolation OFF.

**Examples**

- `CALC:NORM:INT ON`
- `calculate2:normalize:interpolate:state off`

**Query Syntax**

CALCulate<cnm>:NORMalize:INTerpolate[:STATe]?

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** OFF
Calculate:Offset Commands

Allows the data trace magnitude and phase to be offset.

These commands are **Superseded** by the CALCulate:MEASure:OFFSet commands.

```
CALCulate:OFFSet
    MAGNitude  PHASE
        SLOPe
```

Click on a keyword to view the command details.

**See Also**

- Example Programs
- Learn about Magnitude Offset
- Learn about Phase Offset
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:NUM or Calc:Par:Select. Learn more.

```
CALCulate<cnum>:OFFSet:MAGNitude <num>
```
Applicable Models: All

(Read-Write) Offsets the data trace magnitude by the specified value.

To offset the data trace magnitude to a slope value that changes with frequency, use

CALC:OFFS:MAGN:SLOP

See Critical Note

Parameters

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<num>** Offset value in dB.

Examples

- CALC:OFFS:MAGN:4
calculate:offset:magnitude -2

Query Syntax

CALCulate<cnum>:OFFSet:MAGNitude?

Return Type

Numeric

Default

0

CALCulate<cnum>:OFFSet:MAGNitude:SLOPe <num>

Applicable Models: All

(Read-Write) Offsets the data trace magnitude to a value that changes linearly with frequency. The offset slope begins at 0 Hz.

See Critical Note

Parameters

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<num>** Offset slope value in dB/ 1GHz.

Examples

- CALC:OFFS:MAGN:SLOP 1 'Offset slope set to 1dB/1GHz
calculate:offset:magnitude:slope -2 'Offset slope set to -2dB/1GHz

Query Syntax

CALCulate<cnum>:OFFSet:MAGNitude:SLOPe?

Return Type

Numeric

Default

0

CALCulate<cnum>:OFFSet:PHASe <num>[<char>]

3732
Applicable Models: All

(Read-Write) Sets the phase offset for the selected measurement.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<num> Offset phase value. Choose any number between: -360 and 360

<char> Units for phase. OPTIONAL. Choose either:
  DEG - Degrees (default)
  RAD - Radians

Examples

```
CALC:OFFS:PHAS 10
calculate:offset:phase 20rad
```

Query Syntax  CALCulate:OFFSet:PHASe?

Return Type  Numeric, returned value always in degrees

Default  0 degrees
Calculate:Parameter Commands

Lists, creates, selects, and deletes measurements.

For application measurements, use Calc:Custom commands.

```
CALCulate:PARameter:
    CATalog
       | EXTended
    COUNT
    DEFINE
       | EXTended
    DELETE
       | ALL
    MNUMBER
       | [SELECT]
    MODify
       | EXTended
    SELECT
    TAG
       | NEXT?
    TNUMBER?
    WNUMBER?
```

Click on a keyword to view the command details.

Blue commands are superseded.

See Also

- Example Programs
- Learn about Measurement Parameters
- Synchronizing the Analyzer and Controller
Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

CALCulate<cnum>:PARameter:CATalog? <enum> Superseded

Applicable Models: All

Note: This command is replaced with CALC:PAR:CAT:EXTended? which lists parameters with "_" instead of "," allowing the list to be parsed easily. This command will continue to work.

(Read-only) Returns the names and parameters of existing measurements for the specified channel.

Note: For Balanced Measurements: CALC:PAR:CAT? may have an unexpected behavior. Learn more.

See Critical Note

Parameters

- **<cnum>** Channel number of the measurements to be listed. If unspecified, <cnum> is set to 1.
- **<enum>** Choose from:
  - **NORMal** - This is the default if no parameter is specified. If a trace title is defined in a standard channel, then the "name" returned is the same as the trace title. For non standard channels, the "name" returned is the underlying parameter name, regardless of whether the user has turned on a trace title or not.
  - **DISPlay** - If a trace title is defined, then the "name" returned is the same as the trace title.
  - **DEFine** - The "name" returned is always the same as the underlying parameter name, regardless of whether the trace title is turned on or not.

Examples

```
CALC:PAR:CAT? DISP
calculate2:parameter:catalog?
```

Return Type

String - "<measurement name>,<parameter>,[<measurement name>,<parameter>...]"

Default

"CH1_S11_1,S11"

CALCulate<cnum>:PARameter:CATalog:EXTended? <enum>
Applicable Models: All

(Read-only) Returns the names and parameters of existing measurements for the specified channel. This command lists receiver parameters with "_" such that R1,1 is reported as R1_1. This makes the returned string a true "comma-delimited" list all the time.

The returned string of this command is easily parsed and used to create measurements using the CALC:PAR:EXT command.

**Parameters**

- `<cnum>` Channel number of the measurements to be listed. If unspecified, `<cnum>` is set to 1.
- `<enum>` Choose from:
  - **NORMal** - This is the default if no parameter is specified. If a trace title is defined in a standard channel, then the "name" returned is the same as the trace title. For non standard channels, the "name" returned is the underlying parameter name, regardless of whether the user has turned on a trace title or not.
  - **DISPlay** - If a trace title is defined, then the "name" returned is the same as the trace title.
  - **DEFine** - The "name" returned is always the same as the underlying parameter name, regardless of whether the trace title is turned on or not.

**Examples**

```
CALC:PAR:CAT:EXT? DEF
```

```
calculate2:parameter:catalog:extended?
```

**Return Type** String - "<measurement name>,<parameter>[,<measurement name>,<parameter>...]"

**Default** "CH1_S11_1,S11"

CALCulate<cnum>:PARarameter:COUNt <numOfTraces>,[<measClass>]
Applicable Models: All

(Read-Write) Sets or gets the number of traces of selected channel.

Requirements:

- Requires that window [n] exist. So to use the command properly you need to create window [n] before this command can succeed.
- If window[n] exists but is already occupied by another channel’s measurement, an error is returned “Duplicate trace number”.
- If window[n] does not exist, an error is returned: “Window number not found”.
- If the command succeeds, it will always delete measurements.
calc[n]:par:count <m> will delete existing measurements in channel [n] and create <m> copies of S11 in window [n].

Parameters

- `<cnum>` Channel number of the measurements to be listed. If unspecified, `<cnum>` is set to 1.
- `<measClass>` Measurement class name.
- `<numOfTraces>` Number of traces on the selected channel. Varies depending on the upper limit setting for the channel/trace number.

Note: This command always deletes existing measurements and replaces them with S11.

Examples

- 'Create a standard S-parameter channel with one trace
  disp:wind4:state on
calc4:par:count 1

- 'Create a Gain Compression channel with one trace
  disp:wind4:state on
calc4:par:count 1,"Gain Compression"

Query Syntax

- Numeric
- Default 1

CALCulate<cnum>:PARameter[:DEFine] <Mname>,<param>[,port] Superseded

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Note: This command is replaced with CALC:PAR:DEFine:EXTended. This command will continue to work for up to 4-port parameters.
(Write-only) Creates a measurement but does NOT display it.

There is no limit to the number of measurements that can be created. However, there is a limit to the number of measurements that can be displayed. See Traces, Channels, and Windows on the VNA.

- Use \texttt{DISP:WIND:STATe} to create a window if it doesn't already exist.
- Use \texttt{DISP:WIND<wnum>:TRAC<tnum>:FEED <Mname>} to display the measurement.

For Application Measurements see \texttt{CALC:CUST:DEF}

You must select the measurement (\texttt{CALC<cnum>:PAR:SEL <mname>}) before making additional settings.

See Critical Note

\textbf{Parameters}

\begin{itemize}
\item \texttt{<cnum>} Channel number of the new measurement. If unspecified, value is set to 1.
\item \texttt{<Mname>} Name of the measurement. Any non-empty, unique string, enclosed in quotes.
\item \texttt{<param>} Parameter to be measured. Quotes are optional.
\end{itemize}

\textbf{For S-parameters:}

Any S-parameter available in the VNA

\textbf{For ratioed measurements:}

Any two receivers that are available in the VNA. (See the \texttt{block diagram} showing the receivers in YOUR VNA.)

For example: AR1 (this means A/R1)

\textbf{For non-ratioed measurements:}

Any receiver that is available in the VNA. (See the \texttt{block diagram} showing the receivers in YOUR VNA.)

For example: A

\textbf{For Balanced Measurements:}

First create an S-parameter measurement, then change the measurement using \texttt{CALC:FSIM:BAL} commands. See an example.

\textbf{For Applications} see \texttt{CALC:CUST:DEF}.
[port] Optional argument;

For multi-port reflection S-parameter measurements: specifies the VNA port which will provide the load for the calibration. This argument is ignored if a transmission S-parameter is specified.

For all non S-parameter measurements: specifies the source port for the measurement.

Examples

CALC4:PAR 'ch4_S33',S33,2 'Defines an S33 measurement with a load on port2 of the analyzer.'

calculate2:parameter:define 'ch1_a', a, 1 'unratioed meas'

calculate2:parameter:define 'ch1_a', arl,1 'ratioed meas'

Query Syntax

Not Applicable; see Calc:Par:Cat?

Default

Not Applicable

CALCulate<cnum>:PARameter[:DEFine]:EXTended <Mname>,<param>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Note: This command replaces CALC:PAR:DEF as it allows the creating of measurements using external multiport testsets.

(Write-only) Creates a measurement but does NOT display it.

There is no limit to the number of measurements that can be created. However, there is a limit to the number of measurements that can be displayed. See Traces, Channels, and Windows on the VNA.

- Use DISP:WIND:STATe to create a window if it doesn't already exist.
- Use DISP:WIND<wnum>:TRAC<tnum>:FEED <Mname> to display the measurement.

Note: For Application Measurements see CALC:CUST:DEF

You must select the measurement using CALC:PAR:SELect before making additional settings.

See Critical Note

Parameters

<cnum> Channel number of the new measurement. If unspecified, value is set to 1.

<Mname> (String) Name of the measurement. Any non-empty, unique string, enclosed in quotes.
<param> (String ) Measurement Parameter to create. Case sensitive.

**For S-parameters:**

Any S-parameter available in the VNA

Single-digit port numbers CAN be separated by "_" (underscore). For example: "S21" or "S2_1"

Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

**For ratioed measurements:**

Any two VNA physical receivers separated by forward slash '/' followed by comma and source port.

For example: "A/R1, 3"

Learn more about ratioed measurements

See a block diagram showing the receivers in YOUR VNA.

**For non-ratioed measurements:**

Any VNA physical receiver followed by comma and source port.

For example: "A, 4"

Learn more about unratioed measurements.

See the block diagram showing the receivers in YOUR VNA.

Ratioed and Unratioed measurements can also use logical receiver notation to refer to receivers. This notation makes it easy to refer to receivers with an external test set connected to the VNA. You do not need to know which physical receiver is used for each test port. Learn more.

**For ADC measurements:**

Any ADC receiver in the VNA followed by a comma, then the source port.

For example: "AI1,2" indicates the Analog Input1 with source port of 2.

Learn more about ADC receiver measurements.

**For Balanced Measurements:**
First create an S-parameter measurement, then change the measurement using `CALC:FSIM:BAL "define" commands. See an example.

**Note:** For Application Measurements see `CALC:CUST:DEF`

**Examples**

```plaintext
CALC4:PAR:EXT 'ch4_S33', 'S33' 'Defines an S33 measurement
```

```plaintext
calculate2:parameter:define:extended 'ch1_a', 'b9, 1' 'logical receiver notation for unratioed meas of test port 9 receiver with source port 1.
```

```plaintext
calculate2:parameter:define:extended 'ch1_a', 'b9/a10,1' 'logical receiver notation for ratioed meas of test port 9 receiver divided by the reference receiver for port 10 using source port 1
```

**Query Syntax**  Not Applicable; see `Calc:Par:Cat`?

**Default**  Not Applicable

**CALCulate<cnum>:PARameter:DELete[:NAME] <Mname>**

**Applicable Models:** All

(Write-only) Deletes the specified measurement.

See Critical Note

**Parameters**

- `<cnum>`  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<Mname>`  String - Name of the measurement

**Examples**

```plaintext
CALC:PAR:DEL 'TEST'
calculate2:parameter:delete 'test'
```

**Query Syntax**  Not Applicable

**Default**  Not Applicable

**CALCulate:PARameter:DELeTe:ALL**
Applicable Models: All

(Write-only) Deletes all measurements on the VNA.

See Critical Note

### Parameters

<table>
<thead>
<tr>
<th>Examples</th>
<th>CALC:PAR:DEL:ALL</th>
</tr>
</thead>
</table>

### Query Syntax

Not Applicable

Default

Not Applicable

CALCulate\(<cnum>\):PARameter:MNUMber[:SELect] \(<n>[,fast]\)

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the selected measurement for the channel using the Tr#. Most CALC:
commands require that this, or CALC:PAR:SEL, be sent before a setting change is made to that measurement. Each channel can have one selected measurement.

#### Parameters

| \(<cnum>\) | Channel number of the measurement to be selected. If unspecified, \(<cnum>\) is set to 1. |
| \(<n>\) | Numeric - Measurement number. These are the same numbers you see in the “Tr1”, “Tr2” annotation next to the parameter name on the VNA screen. |
| [fast] | Optional. The VNA display is NOT updated. Therefore, do not use this argument when an operator is using the VNA display. Otherwise, sending this argument results in much faster sweep speeds. There is NO other reason to NOT send this argument. |

#### Examples

| CALC:PAR:MNUM 2 |
| calculate2:parameter:mnumber:select 3,fast |

#### Query Syntax

CALCulate\(<cnum>\):PARameter:MNUMber[:SELect]? |

There is NO query available to determine if the FAST argument has been set.

#### Return Type

Numeric

Default

1 (Trace number when factory preset is performed)

CALCulate\(<cnum>\):PARameter:MODify <param> Superseded
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Note: This command is replaced with \texttt{CALC:PAR:MOD:EXT}. This command will continue to work for up to 4 port parameters.

(Write-only) Modifies a standard measurement using the same arguments as \texttt{CALC:PAR:DEF}. To modify an FCA measurement, use \texttt{CALC:CUST:MOD}.

See Critical Note

Parameters
\begin{itemize}
    \item \texttt{<cnum>} Channel number of the measurement. The selected measurement on that channel will be changed. If unspecified, \texttt{<cnum>} is set to 1.
    \item \texttt{<param>} Measurement parameter to change to. Use the same \texttt{<param>} arguments as \texttt{CALC:PAR:DEF}.
\end{itemize}

Examples
\begin{verbatim}
SYST:PRESET
CALC:PAR:DEF "MyMeas", S11
CALC:PAR:SEL "MyMeas"
CALC:PAR:MOD AR1 'changes the selected S11 measurement to an A/R1 measurement'
\end{verbatim}

Query Syntax
- Not Applicable

Default
- Not Applicable

\texttt{CALCulate<cnum>:PARameter:MODify:EXTended <param>}

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Note: This command replaces \texttt{CALC:PAR:MOD} as it allows modification of measurements using external multiport testsets.

(Write-only) Modifies a standard measurement using the same arguments as \texttt{CALC:PAR:DEF:EXT}.

To modify an Application measurement, use \texttt{CALC:CUST:MOD}.

See Critical Note

Parameters
\begin{itemize}
    \item \texttt{<cnum>} Channel number of the measurement. The selected measurement on that channel will be changed. If unspecified, \texttt{<cnum>} is set to 1.
\end{itemize}
<param>  (String) New measurement parameter. Use the same <param> arguments as CALC:PAR:DEF:EXT.

Examples

SYST:PRESET
CALC:PAR:DEF:EXT "MyMeas", "S10_1"
CALC:PAR:SEL "MyMeas"
CALC:PAR:MOD:EXT "a4b4,1"  'changes the selected S10_1 measurement to an a4/b4 measurement with source port 1

Query Syntax

Not Applicable

Default

Not Applicable

CALCulate<cnum>:PARameter:TAG:NEXT?

Applicable Models:  E5080A, M9485A

(Read-only)

Parameters

<cnum>  Channel number of the measurement. The selected measurement on that channel will be changed. If unspecified, <cnum> is set to 1.

Examples

CALC:PAR:TAG:NEXT

Query Syntax

Not Applicable

Default

Not Applicable

CALCulate<cnum>:PARameter:SELect <Mname>[,.fast]
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the selected measurement. Most CALC: commands require that this command be sent before a setting change is made. One measurement on each channel can be selected at the same time.

- Use `CALC:PAR:MNUM` to select a measurement by Tr# number. Learn more.
- To obtain a list of currently named measurements, use `CALC:PAR:CAT?`

### Parameters

- `<cnum>` Channel number of the measurement to be selected. If unspecified, `<cnum>` is set to 1.
- `<Mname>` String - Name of the measurement. CASE-SENSITIVE. Do NOT include the parameter name that is returned with Calc:Par:Cat?
- `[fast]` Optional. The VNA display is NOT updated. Therefore, do not use this argument when an operator is using the VNA display. Otherwise, sending this argument results in much faster sweep speeds. There is NO other reason to NOT send this argument.

### Examples

```
CALC:PAR:SEL 'TEST'
calculate2:parameter:select 'test',fast
```

### Query Syntax

```
CALCulate:PARameter:SELection?
```

There is NO query available to determine if the FAST argument has been set.

### Return Type

String

**Default**

"CH1_S11_1" (Trace name when factory preset is performed)

---

**CALCulate<cnum>:PARameter:TAG:NEXT?**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Returns a string that is guaranteed to be unique and valid for use with `CALC:PAR:DEF`.

### Parameters

- `<cnum>` Channel number of the trace. If unspecified, `<cnum>` is set to 1.

### Examples

```
CALC:PAR:TAG:NEXT?
calculate2:parameter:tag:next?
```

### Return Type

String

**Default**

Not Applicable
CALCulate<cnum>:PARameter:TNUMber?

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-only)* Returns the trace number of the selected trace. Select a trace using Calc:Par:Select.

**Parameters**

\(<\text{cnum}>\)    Channel number of the trace. If unspecified, \(<\text{cnum}>\) is set to 1.

**Examples**

```
CALC:PAR:TNUM?
calculate2:parameter:tnumber?
```

**Return Type** Numeric

**Default** Not Applicable

---

CALCulate<cnum>:PARameter:WNUMber?

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-only)* Returns the window number of the selected trace. Select a trace using Calc:Par:Select.

**Parameters**

\(<\text{cnum}>\)    Channel number of the selected trace. If unspecified, \(<\text{cnum}>\) is set to 1.

**Examples**

```
CALC:PAR:WNUM?
calculate2:parameter:wnumber?
```

**Return Type** Numeric

**Default** Not Applicable
Calculate:RData? Command

This command is **Superseded** by the **CALCulate:MEASure:RDA**Ta command.

---

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using **Calc:Par:MNUM** or **Calc:Par:Select**. Learn more.

CALCulate<cnum>:RDATA? <char>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Returns receiver data for the selected measurement. To query measurement data, see **CALC:DATA?**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>` Choose from any physical receiver in the VNA.

For example: "A"

Also, **REF** - returns data for either R1 or R2 data depending on the source port of the selected measurement.

See the block diagram showing the receivers in YOUR VNA.

**Note:** Logical receiver notation is NOT allowed with this command. Learn more.

**Example**

```
GPIB.Write "INITiate:CONTinuous OFF"
GPIB.Write "INITiate:IMMediate;*wai"
GPIB.Write "CALCulate:RDATA? A"
```

```
GPIB.Write "CALCulate:RDATA? REF"
```

**Return Type** Depends on **FORM:DATA** - Two numbers per data point

**Default** Not Applicable

**Notes:**

Generally when you query the analyzer for data, you expect that the number of data values returned will be consistent with the number of points in the sweep.

However, if you query **receiver** data while the instrument is sweeping, the returned values may contain
zeros. For example, if your request for receiver data is handled on the 45th point of a 201 point sweep, the first 45 values will be valid data, and the remainder will contain complex zero.

This can be avoided by synchronizing this request with the end of a sweep or putting the channel in hold mode.

Learn about Unratioed Measurements
**CALCulate:SA:MARKer commands**

Controls the marker settings used in the SA application.

These commands are **Superseded** by the **CALCulate:MEASure:SA** commands.

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BDENsity</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>ACPR</strong></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>BW</strong></td>
</tr>
<tr>
<td></td>
<td><strong>DATA?</strong></td>
</tr>
<tr>
<td></td>
<td><strong>EQSPan</strong></td>
</tr>
<tr>
<td></td>
<td><strong>NOISe</strong></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>NPR</strong></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>POWer</strong></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>TONE</strong></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>TSPacing</strong></td>
</tr>
<tr>
<td><strong>BPOWer</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>DATA?</strong></td>
</tr>
</tbody>
</table>
Click on a keyword to view the command details.

See Also

- Marker Readout number and size commands.
- Learn about Markers
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

If you use Calc:Par:Cat? to return the list of current measurements, the returned string will be similar to ‘CH1_B_1,B’.

To select this measurement as a parameter for SA, you need to send Calc:Par:Sel ‘CH1_B_1’.

Moreover, most of the following commands will return ‘+202, “Parameter not valid” if they are called with marker number n, and marker number n is not currently turned ON.

**Important:** Learn about programming the reference marker.

**Note:** For all band power marker family (this includes BNOise, BPOWer, OCCBand markers) and
when measuring wideband repetitive modulated signals, there is basically 2 approaches to get good measurements:

- Either run the coherent mode of SA if the modulated test signal repetition period or tone spacing is known and a frequency reference connection is made between the signal source and the PNA (usually the 10 MHz ref signal BNC). See SA Setup Coherence.

- Or make use of a RBW that is equal or smaller than 1/20 of the modulated test signal tone spacing (rule of thumb). The test signal tone spacing is 1/ the test signal duration. For example, if the ARB file that generates the test signal replays each 100 microseconds, it means the tone spacing is 10 kHz. So, we recommend to set the RBW to 500 Hz or below.

CALCulate<ch>:SA:MARKer<n>:BDENsity:ACPR[:STATe] <bool>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B) or N524xB models with Option S93070xB

(Read-Write) Sets and reads the ACPR density marker.

**See Critical Note**

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<ch>` is set to 1.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - Turn band density marker OFF.
  - 1 - ON - Turn band density marker ON.

**Examples**

'Select the measurement
CALC2:PAR:SEL "M2SA_CH2_A"

'Create marker3 on that measurement
CALC2:MARK3 ON

'Make it a band density noise marker
CALC2:SA:MARK3:BDEN:ACPR:STAT 1

**Query Syntax** CALCulate<ch>:SA:MARKer<n>:BDENsity:ACPR[:STATe]?

**Return Type** Boolean
CALCulate<ch>:SA:MARKer<n>:BDENsity:BW <num>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

*(Read-Write)* Sets and reads the bandwidth of the band density marker.

**See Critical Note**

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<ch>` is set to 1.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<num>` Choose a bandwidth.

**Examples**

```
CALC:SA:MARK:BDEN:BW 1e6
```

**Query Syntax**

```
CALCulate<ch>:SA:MARKer<n>:BDENsity:BW?
```

**Return Type** Numeric

**Default** 1 MHz

---

CALCulate<ch>:SA:MARKer<n>:BDENsity:DATA?

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

*(Read-only)* Returns the band density level in dBm/Hz from the band density marker.

**See Critical Note**

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that SA channel. If unspecified, `<ch>` is set to 1.
- `<n>` Band density marker number. If unspecified, `<n>` is set to 1.

**Examples**

```
CALC:SA:MARK:BDEN:DATA?
calculate2:sa:marker2:bdensity:data?
```

**Return Type** Numeric

**Default** Not applicable

---

3752
**CALCulate<ch>:SA:MARKer<n>:BDENsity:EQSPan <num>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-Write) Sets and reads the frequency span used by Power Density to normalize the power.

**See Critical Note**

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<ch>` is set to 1.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<num>` Choose a span.

**Examples**

```
CALC:SA:MARK:BDEN:EQSPan 1e6
```

**Query Syntax**

CALCulate<ch>:SA:MARKer<n>:BDENsity:EQSPan?

**Return Type**

Numeric

**Default**

1 MHz

---

**CALCulate<ch>:SA:MARKer<n>:BDENsity:NOISe[:STATe] <bool>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-Write) Sets and reads the state of the band density noise marker.

**See Critical Note**

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<ch>` is set to 1.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - Turn band density noise marker OFF.
  - 1 - ON - Turn band density noise marker ON.

**Examples**

```
'Select the measurement
CALC2:PAR:SEL "M2SA_CH2_A"

'Create marker3 on that measurement
```
CALC2:MARK3 ON

'Make it a band density noise marker

CALC:SA:MARK:BDEN:NOIS:STAT 1

**Query Syntax**
CALCulate<ch>:SA:MARKer<n>:BDENsity:NOISe?

**Return Type**
Boolean

**Default**
0

---

**CALCulate<ch>:SA:MARKer<n>:BDENsity:NPR[:STATe] <bool>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B) or N524xB models with Option S93070xB

(Read-Write) Sets and reads the NPR density marker.

**See Critical Note**

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<ch>` is set to 1.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<bool>` Choose from:

  - **0 - OFF** - Turn band density marker OFF.
  - **1 - ON** - Turn band density marker ON.

**Examples**

'Select the measurement
CALC2:PAR:SEL "M2SA_CH2_A"

'Create marker3 on that measurement
CALC2:MARK3 ON

'Make it a band density noise marker
CALC2:SA:MARK3:BDEN:NPR:STAT 1

**Query Syntax**
CALCulate<ch>:SA:MARKer<n>:BDENsity:NPR[:STATe]?

**Return Type**
Boolean

**Default**
0

---

3754
CALCulate<ch>:SA:MARKer<n>:BDENsity:POWer:BW <num>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-Write) Sets and reads the bandwidth of the band power density marker.

See Critical Note

Parameters

<ch> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <ch> is set to 1.

<n> Marker number. If unspecified, <n> is set to 1.

<num> Choose a bandwidth.

Examples


Query Syntax

CALCulate<ch>:SA:MARKer<n>:BDENsity:POWer:BW?

Return Type

Numeric

Default

1 MHz

CALCulate<ch>:SA:MARKer<n>:BDENsity:POWer[:STATe] <bool>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-Write) Sets and reads the state of the band power density marker.

See Critical Note

Parameters

<ch> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <ch> is set to 1.

<n> Marker number. If unspecified, <n> is set to 1.

<bool> Choose from:

0 - OFF - Turn band power density marker OFF.

1 - ON - Turn band power density marker ON.

Examples

'Select the measurement

CALC2:PAR:SEL "M2SA_CH2_A"

'Create marker3 on that measurement
'Make it a band density noise marker

CALC:SA:MARK:BDEN:POW:STAT 1

**Query Syntax**

CALCulate<ch>:SA:MARKer<n>:BDENsity:POWer?

**Return Type**

Boolean

**Default**

0

**CALCulate<ch>:SA:MARKer<n>:BDENsity:TONE:BW <num>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

*(Read-Write)* Sets and reads the bandwidth of the band tone density marker.

**See Critical Note**

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<ch>` is set to 1.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<num>` Choose a bandwidth.

**Examples**

CALC:SA:MARK:BDEN:TONE:BW 1e6

**Query Syntax**

CALCulate<ch>:SA:MARKer<n>:BDENsity:TONE:BW?

**Return Type**

Numeric

**Default**

1 MHz

**CALCulate<ch>:SA:MARKer<n>:BDENsity:TONE[:STATe] <bool>**
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-Write) Sets and reads the state of the band tone density marker.

See Critical Note

Parameters

<ch> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <ch> is set to 1.

<n> Marker number. If unspecified, <n> is set to 1.

<bool> Choose from:

0 - OFF - Turn band tone density marker OFF.

1 - ON - Turn band tone density marker ON.

Examples

'Select the measurement
CALC2:PAR:SEL "M2SA_CH2_A"

'Create marker3 on that measurement
CALC2:MARK3 ON

'Make it a band density noise marker
CALC:SA:MARK:BDEN:TONE:STAT 1

Query Syntax
CALCulate<ch>:SA:MARKer<n>:BDENsity:TONE?

Return Type
Boolean

Default
0
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

**(Read-Write)** Sets and reads the spacing of the band tone density marker.

**See Critical Note**

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<ch>` is set to 1.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<num>` Choose a spacing value.

**Examples**

`CALC:SA:MARK:BDEN:TONE:TSP 100e6`

**Query Syntax**

`CALCulate<ch>:SA:MARKer<n>:BDENsity:TONE:TSPacing?`

**Return Type** Numeric

**Default** 100 MHz

---

**CALCulate<ch>:SA:MARKer<n>:BPOWer:DATA?**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

**(Read-only)** Returns the band power level from the band power marker.

**See Critical Note**

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that SA channel. If unspecified, `<ch>` is set to 1.
- `<n>` Band power marker number. If unspecified, `<n>` is set to 1.

**Examples**

`CALC:SA:MARK:BPOW:DATA?`

`calculate2:sa:marker2:bpower:DATA?`

**Default** Not applicable

---

**CALCulate<ch>:SA:MARKer<n>:BPOWer:SPAN <num>**
**CALCulate<ch>:SA:MARKer<n>:BPOWer[:STATe] <bool>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

*(Read-Write)* Sets and reads the state of the band power marker. This command makes a band power marker from a generic marker. The generic marker must first be created using: `CALC:MARK:STATe`

See Critical Note

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<ch>` is set to 1.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - Turn band power marker OFF.
  - 1 - ON - Turn band power marker ON.

**Examples**

```
'Select the measurement

CALC2:PAR:SEL "M2SA_CH2_A"
```
Create marker3 on that measurement

CALC2:MARK3 ON

Make it a band power marker

CALC:SA:MARK:BPOW:STAT 1

Query Syntax

CALCulate<ch>:SA:MARKer<n>:BPOWer?

Return Type

Boolean

Default

0

CALCulate<ch>:SA:MARKer<n>:OCCBand:CENTer?

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

(Read-only) Returns the occupied bandwidth center frequency.

See Critical Note

Parameters

<ch> Channel number of the measurement. There must be a selected measurement on that SA channel. If unspecified, <ch> is set to 1.

<n> Marker number. If unspecified, <n> is set to 1.

Examples

CALC:SA:MARK:OCCB:CENT?

calculate2:sa:marker2:occband:cent?

Default Not applicable

CALCulate<ch>:SA:MARKer<n>:OCCBand:PERCent <num>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

*(Read-Write)* Sets and returns the percentage of the band power to search for.

See Critical Note

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<ch>` is set to 1.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<num>` Percentage value.

**Examples**

```
calculate2:sa:marker2:occband:percent 99
```

**Query Syntax**

```
CALCulate<ch>:SA:MARKer<n>:OCCBand:PERCent?
```

**Return Type**

Numeric

**Default**

99.0

---

**CALCulate<ch>:SA:MARKer<n>:OCCBand:POWer?**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

*(Read-only)* Returns the occupied bandwidth power.

See Critical Note

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that SA channel. If unspecified, `<ch>` is set to 1.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.

**Examples**

```
CALC:SA:MARK:OCCB:POW?
calculate2:sa:marker2:occband:power?
```

**Default**

Not applicable

---

**CALCulate<ch>:SA:MARKer<n>:OCCBand:SPAN?**
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

**(Read-only)** Returns the occupied bandwidth span.

See Critical Note

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that SA channel. If unspecified, `<ch>` is set to 1.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.

**Examples**

```
CALC:SA:MARK:OCCB:SPAN?
calculate2:sa:marker2:occband:span?
```

**Default** Not applicable

---

**CALCulate<ch>:SA:MARKer<n>:OCCBand[:STATe] <bool>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x090A/B)

**(Read-Write)** Sets and returns the occupied bandwidth on/off state.

See Critical Note

**Parameters**

- `<ch>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<ch>` is set to 1.
- `<n>` Marker number. If unspecified, `<n>` is set to 1.
- `<bool>` Choose from:
  - **0** - OFF - Turns occupied bandwidth OFF.
  - **1** - ON - Turns occupied bandwidth ON.

**Examples**

```
CALC:SA:MARK:OCCB:STAT 1
calculate2:sa:marker2:occband:state 1
```

**Query Syntax** CALCulate<ch>:SA:MARKer<n>:OCCBand[:STATe]?

**Return Type** Boolean
**Note:** If occupied band state is turned ON, then Band Power or Band Noise is turned OFF.
Calculate: Smoothing Commands

Controls point-to-point smoothing. Smoothing is a noise reduction technique that averages adjacent data points in a measurement trace. Choose the amount of smoothing by specifying either the number of points or the aperture. Smoothing is not the same as CALC:AVERage which averages each data point over a number of sweeps.

These commands are Superseded by the CALCulate:MEASure:SMoothing commands.

![CALCulate:SMoothing Diagram]

Click on a keyword to view the command details.

See Also

- Example Programs
- Learn about Smoothing
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

CALCulate<cnum>:SMOothing:APERture <num>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the amount of smoothing as a percentage of the number of data points in the channel.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Percentage value. Choose any number between: 1 and 25

**Examples**

```plaintext
CALC:SMO:APER 2

calculate2:smoothing:aperture 20.7
```

**Query Syntax**

CALCulate<cnum>:SMOothing:APERture?

**Return Type** Numeric

**Default** 1.5

**CALCulate<cnum>:SMOothing:POINts <num>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the number of adjacent data points to average.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Number of points from 1 point to maximum of 25% of data points in the channel. For example: if number of points in a data trace = 401, the maximum value for points = 100. The points value is always rounded to the closest odd number.

**Examples**

```plaintext
CALC:SMO:POIN 50

calculate2:smoothing:points 21
```

**Query Syntax**

CALCulate<cnum>:SMOothing:POINts?

**Return Type** Numeric

**Default** 3
CALCulate<cnum>:SMOothing[:STATe] <ON | OFF>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Turns data smoothing ON or OFF.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<ON | OFF> ON (or 1) - turns smoothing ON.
           OFF (or 0) - turns smoothing OFF.

Examples

CALC:SMO ON
calculate2:smoothing:state off

Query Syntax CALCulate<cnum>:SMOothing[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF
CALulate:TDR Commands

These commands control the setup and execution of TDR measurements.

<table>
<thead>
<tr>
<th>CALCulate:TDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLocate</td>
</tr>
<tr>
<td>DEEM</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>DEVice</td>
</tr>
<tr>
<td>EMPHasis</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>EQUalization</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>EYE</td>
</tr>
<tr>
<td>STATE</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>ACTIVE</td>
</tr>
<tr>
<td>DTIME</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>FORMAT</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>PEEILING</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>TIME</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Click on a red keyword to view the command details.

See Also

- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**CALCulate<cnum>:TDR:ALLocate <enum>**

**Applicable Models:** All with TDR Options (S9x011A/B)

**(Read-Write)** This command sets the type of parameter and format allocation for each trace.

**Parameters**

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<enum>` Parameters to display. Choose from:
  
  **SPARameters** - Display all S-parameter data.
  
  **TPARameters** - Display all Time domain data.
  
  **MIXed** - Display a mix of commonly measured time-domain and S-parameter data.

**Examples**

```
CALC:TDR:ALL SPAR
```

```
calculate2:tdr:allocate spparameters
```

**Query Syntax**

```
CALCulate<cnum>:TDR:ALLocate?
```

**Return Type**

String

**Default**

MIXed
CALCulate<cnum>:TDR:DEVice <enum>

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the DUT topology.

**Parameters**

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<enum>` Topology. Choose from:
  - **SEND1** - Single-ended, 1-port.
  - **SEND2** - Single-ended, 2-port.
  - **DIF1** - Differential, 1-port.
  - **SEND4** - Single-ended, 4-port.
  - **DIF2** - Differential, 2-port.

**Examples**

```
CALC:TDR:DEV SEND2
calculate2:tdr:device:send2
```

**Query Syntax**

CALCulate<cnum>:TDR:DEVice?

**Return Type** String

**Default** SEND1
**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the filename of the s4p de-embedding user file.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<pnum>`: Port number for balance port. If unspecified, `<pnum>` is set to 1.
- `<file>`: File Name. This file is saved as a 4-port touchstone file with the .s4p extensions.

**Examples**
- `CALC:TDR:DEEM:BPOR1:FIL "test.s4p"`
- `calculate2:tdr:deem:bport2:filename "test.s4p"`

**Query Syntax**
- `CALCulate<cnum>:TDR:DEEM:BPOR<pnum>:FILename?`

**Return Type**
- `String`

**Default**
- " " (Empty String)

---

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets s4p de-embedding function state ON/OFF.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<pnum>`: Port number for balance port. If unspecified, `<pnum>` is set to 1.
- `<bool>`: ON or 1 - Turns s4p de-embedding function ON.
  - OFF or 0 - Turns s4p de-embedding function OFF.

**Examples**
- `CALC:TDR:DEEM:BPOR1:STAT ON`
- `calculate2:tdr:deem:bport2:state off`

**Query Syntax**
- `CALCulate<cnum>:TDR:DEEM:BPOR<pnum>:STATe?`

**Return Type**
- `Boolean`

**Default**
- OFF

---

```bash
CALCulate<cnum>:TDR:DEEM:PORT<pnum>:FILENAME <file>
```
**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) This command sets the filename of the s2p de-embedding user file.

**Parameters**

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<pnum>` Port number. If unspecified, `<pnum>` is set to 1.
- `<file>` File Name. This file is saved as a 2-port touchstone file with the .s2p extensions.

**Examples**

```
CALC:TDR:DEEM:PORT1:FIL "test.s2p"
calculate2:tdr:deem:port2:filename "test.s2p"
```

**Query Syntax**

CALCulate<cnum>:TDR:DEEM:PORT<pnum>:FILENAME?

**Return Type**

String

**Default**

"" (Empty String)

---

**CALCulate<cnum>:TDR:DEEM:PORT<pnum>:STATe <bool>**

**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) This command sets s2p de-embedding function state ON/OFF.

**Parameters**

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<pnum>` Port number. If unspecified, `<pnum>` is set to 1.
- `<bool>` ON or 1 - Turns s2p de-embedding function ON.
  OFF or 0 - Turns s2p de-embedding function OFF.

**Examples**

```
CALC:TDR:DEEM:PORT1:STAT ON
calculate2:tdr:deem:port2:state off
```

**Query Syntax**

CALCulate<cnum>:TDR:DEEM:PORT<pnum>:STATe?

**Return Type**

Boolean

**Default**

OFF

---

**CALCulate<cnum>:TDR:DEEM:STATe <bool>**
**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) This command sets de-embedding function state ON/OFF.

**Parameters**

- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<bool>`: ON or 1 - Turns de-embedding function ON. OFF or 0 - Turns de-embedding function OFF.

**Examples**

```
CALC:TDR:DEEM:STAT ON
calculate2:tdr:deem:state off
```

**Query Syntax**

```
CALCulate<cnum>:TDR:DEEM::STATe?
```

**Return Type**

Boolean

**Default**

OFF

---

**CALCulate<cnum>:TDR:EMPHasis:CURSor:POST1 <value>**

**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) This command sets the emphasis post1 level.

**Parameters**

- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>`: Post1 level value. The range is -20 dB to +20 dB.

**Examples**

```
CALC:TDR:EMPH:CURS:POST1 3
calculate2:tdr:emphasis:cursor:post1 3
```

**Query Syntax**

```
CALCulate<cnum>:TDR:EMPH:CURS:POST1?
```

**Return Type**

Double

**Default**

0

---

**CALCulate<cnum>:TDR:EMPHasis:CURSor:POST2 <value>**
**CALCulate<cnump>:TDR:EMPHasis:CURSor:POST2 3**

**CALCulate<cnump>:TDR:EMPHasis:CURSor:PRE1 2**

**CALCulate<cnump>:TDR:EMPHasis:STATe ON**
CALCulate<cnum>:TDR:EQUalization:CTLE:DC <value>

Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the equalization CTLE (Continuous Time Linear Equalization) DC gain parameter.

Parameters
- <cnum> Channel number of the measurement. If unspecified, <cnum> is set to 1.
- <value> CTLE value. The range is 0 to 10.

Examples
- CALC:TDR:EQU:CTLE:DC 0.077
- calculate2:tdr:equalization:ctle:dc 0.077

Query Syntax
- CALCulate<cnum>:TDR:EQUalization:CTLE:DC?

Return Type
- Double

Default
- 0.667

CALCulate<cnum>:TDR:EQUalization:CTLE:POLE1 <value>

Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the equalization CTLE (Continuous Time Linear Equalization) Pole1 parameter.

Parameters
- <cnum> Channel number of the measurement. If unspecified, <cnum> is set to 1.
- <value> CTLE value. The range is 0 to 76E9 in Hz.

Examples
- CALC:TDR:EQU:CTLE:POLE1 2E9
- calculate2:tdr:equalization:ctle:pole1 2e9

Query Syntax
- CALCulate<cnum>:TDR:EQUalization:CTLE:POLE1?

Return Type
- Double

Default
- 1.95E9

CALCulate<cnum>:TDR:EQUalization:CTLE:POLE2 <value>
Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the equalization CTLE (Continuous Time Linear Equalization) Pole2 parameter.

Parameters
- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>` CTLE value. The range is 0 to 76E9 in Hz.

Examples
- `CALC:TDR:EQU:CTLE:POLE2 2E9`
- `calculate2:tdr:equalization:ctle:pole2 2e9`

Query Syntax
- `CALCulate<cnum>:TDR:EQUalization:CTLE:POLE2?`

Return Type
- Double

Default
- 5E9

CALCulate<cnum>:TDR:EQUalization:CTLE:ZERO1 <value>

Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the equalization CTLE (Continuous Time Linear Equalization) zero parameter.

Parameters
- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>` CTLE value. The range is 0 to 76E9 in Hz.

Examples
- `CALC:TDR:EQU:CTLE:ZERO1 650E6`
- `calculate2:tdr:equalization:ctle:zero1 650e6`

Query Syntax
- `CALCulate<cnum>:TDR:EQUalization:CTLE:ZERO1?`

Return Type
- Double

Default
- 650E6

CALCulate<cnum>:TDR:EQUalization:FILENAME <file>
**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the filename of the equalization equation user file. It is necessary to select the file using CALCulate:TDR:EQUalization:TYPE and turn ON the :CALCulate:TDR:EQUalization:STATe. This file is saved with .csv extension.

Specify the file name with the extension. When you use directory names (folder names) and file name, separate them with "\" (back slash), or "/" (slash).

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<file>`: Filename up to 254 characters.

**Examples**
- `CALC:TDR:EQU:FIL "C:\folder\User.csv"`
- `calculate2:tdr:equalization:filename "C:\folder\User.csv"`

**Query Syntax**
- `CALCulate<cnum>:TDR:EQUalization:FILename?`

**Return Type**
- String

**Default**
- " " (Empty String)

**CALCulate<cnum>:TDR:EQUalization:STATe <bool>**

*Applicable Models:* All with TDR Options (S9x011A/B)

*(Read-Write)* Turns the equalization function state ON or OFF.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<bool>`: ON or 1 - Turns equalization ON. OFF or 0 - Turns equalization OFF.

**Examples**
- `CALC:TDR:EQU:STAT ON`
- `calculate2:tdr:equalization:state off`

**Query Syntax**
- `CALCulate<cnum>:TDR:EQUalization:STATe?`

**Return Type**
- Boolean

**Default**
- OFF

**CALCulate<cnum>:TDR:EQUalization:TYPE <enum>**
**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the equalization type.

**Parameters**

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<enum>` Equalization type. Choose from:
  - **EQUation** - Option to enter values for equation calculation.
  - **USER** - Load user file.

**Examples**

- `CALC:TDR:EQU:TYPE EQU`
- `calculate2:tdr:equalization:type equation`

**Query Syntax**

`CALCulate<cnum>:TDR:EQUalization:TYPE?`

**Return Type** Double

**Default** EQUation

---

**CALCulate<cnum>:TDR:EYE:EXECute**

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Write-only)* This command performs the calculation for the simulated eye diagram for the active trace.

**Parameters**

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

**Example**

- `CALC:TDR:EYE:STAT ON`
- `CALC:PAR:MNUM:SEL 3`
- `CALC:TDR:EYE:EXEC`

---

**CALCulate<cnum>:TDR:EYE:INPut:BPATtern:LENGth <value>**
**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) This command sets the bits’ power of 2 for a PRBS pattern. This value is used only when the selected bit pattern type is PRBS.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>`: Bit pattern power of 2. The range is 3 to 15 (resolution is 1).

**Examples**
- `CALC:TDR:EYE:INP:BPAT:TYPE PRBS`
- `CALC:TDR:EYE:INP:BPAT:LENG 7`

**Query Syntax**
- `CALCulate<cnum>:TDR:EYE:INPut:BPATtern:LENGth?`

**Return Type**
- Integer

**Default**
- 7

---

**CALCulate<cnum>:TDR:EYE:INPut:BPATtern:TYPE <enum>**

**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) This command sets the bit pattern type for the simulated eye function.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<enum>`: Eye pattern type. Choose from:
  - **PRBS** - Pseudo-Random Bit Sequence.
  - **K285** - K 28.5.
  - **USER** - Custom user-defined pattern.
  - **STAT** - Statistical calculation.

**Examples**

**Query Syntax**
- `CALCulate<cnum>:TDR:EYE:INPut:BPATtern:TYPE?`

**Return Type**
- String

**Default**
- PRBS

---

**CALCulate<cnum>:TDR:EYE:INPut:DRATe <value>**
Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the bit rate in bits/sec for the simulated eye function.

Parameters
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>`: Bit rate in bits/sec. The range is 1.21M to 60.8G.

Examples
- `CALC:TDR:EYE:INP:DRAT 1.1E9`
- `calculate2:tdr:eye:input:drate 1.1e9`

Query Syntax: `CALCulate<cnum>:TDR:EYE:INPut:DRATe?`

Return Type: Double

Default: 1G

CALCulate<cnum>:TDR:EYE:INPut:JITTer:DLIMit <value>

Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the display limit value.

Parameters
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>`: Display limit value. The range is 0 to 1.

Examples
- `CALC:TDR:EYE:INP:JITT:DLIM 0`
- `calculate2:tdr:eye:input:jitter:dlimit 0`

Query Syntax: `CALCulate<cnum>:TDR:EYE:INPut:JITTer:DLIMit?`

Return Type: Double

Default: 1E-9

CALCulate<cnum>:TDR:EYE:INPut:JITTer:PERiodic:FREQuency <value>

Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the periodic jitter frequency. This value is used only when the periodic jitter function type is selected.

Parameters
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>`: Periodic jitter frequency in Hz. The range is 0 to TDR eye jitter max frequency.

Examples
- `CALC:TDR:EYE:INP:JITT:PER:FREQ 0`
- `calculate2:tdr:eye:input:jitter:periodic:frequency 0`


Return Type: Double

Default: 500E3
CALCulate\(<\text{cnum}>\)::TDR:EYE:INPut:JITTer:PERiodic:MAGNitude \(<\text{value}>\)

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the periodic jitter magnitude in rms. This value is used only when periodic jitter function type is selected.

**Parameters**
- \(<\text{cnum}>\): Channel number of the measurement. If unspecified, \(<\text{cnum}>\) is set to 1.
- \(<\text{value}>\): Periodic jitter magnitude. The range is 0 to 1 UI.

**Examples**
- **CALC::TDR::EYE::INP::JITT::TYPE::PER**
- **CALC::TDR::EYE::INP::JITT::PER::MAGN 0.5**

**Query Syntax**

```
CALCulate\(<\text{cnum}>\)::TDR:EYE:INPut:JITTer:PERiodic:MAGNitude? 
```

**Return Type**

Double

**Default**

0

CALCulate\(<\text{cnum}>\)::TDR:EYE:INPut:JITTer:RANDom:MAGNitude \(<\text{value}>\)

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the random jitter magnitude in rms. This value is used only when random jitter function type is selected.

**Parameters**
- \(<\text{cnum}>\): Channel number of the measurement. If unspecified, \(<\text{cnum}>\) is set to 1.
- \(<\text{value}>\): Random jitter magnitude. The range is 0 to 0.25 UI.

**Examples**
- **CALC::TDR::EYE::INP::JITT::TYPE::RAND**
- **CALC::TDR::EYE::INP::JITT::RAND::MAGN 0.05**

**Query Syntax**

```
CALCulate\(<\text{cnum}>\)::TDR:EYE:INPut:JITTer:RANDom:MAGNitude? 
```

**Return Type**

Double

**Default**

0

CALCulate\(<\text{cnum}>\)::TDR:EYE:INPut:JITTer:STATe \(<\text{bool}>\)
**Applicable Models:** All with TDR Options (S9x011A/B)

**(Read-Write)** Turns the jitter function state with simulated eye ON or OFF.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<bool>`: ON or 1 - Turns jitter ON.
  OFF or 0 - Turns jitter OFF.

**Examples**
```
CALC:TDR:EYE:INP:JITT:STAT ON  
calculate2:tdr:eye:input:jitter:state off
```

**Query Syntax**
```
CALCulate<cnum>:TDR:EYE:INPut:JITTer:STATe?
```

**Return Type**
Boolean

**Default**
OFF

---

**CALCulate<cnum>:TDR:EYE:INPut:JITTer:TYPE <enum>**

**Applicable Models:** All with TDR Options (S9x011A/B)

**(Read-Write)** This command sets the jitter function type for the simulated eye function.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<enum>`: Jitter type. Choose from:
  - **RANDom** - Random jitter function.
  - **PERiodic** - Periodic jitter function.

**Examples**
```
CALC:TDR:EYE:INP:JITT:TYPE RAND  
calculate2:tdr:eye:input:jitter:type random
```

**Query Syntax**
```
CALCulate<cnum>:TDR:EYE:INPut:JITTer:TYPE?
```

**Return Type**
String

**Default**
PERiodic

---

**CALCulate<cnum>:TDR:EYE:INPut:OLEVel <value>**
**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the voltage level for bit "1" for the simulated eye function.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>`: Level for bit "1". The range is -5 V to +5 V.

**Examples**
- `CALC:TDR:EYE:INP:OLEV 100E-3`
- `calculate2:tdr:eye:input:olevel 100e-3`

**Query Syntax**
- `CALCulate<cnum>:TDR:EYE:INPut:OLEVel?`

**Return Type**
- Double

**Default**
- 0.2

---

**CALCulate<cnum>:TDR:EYE:INPut:RTIMe:DATA <value>**

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the rise time value for the simulated eye function.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>`: Rise time for simulated eye function in seconds.

**Examples**
- `CALC:TDR:EYE:INP:RTIM:DATA 90E-12`
- `calculate2:tdr:eye:input:rtime:data 90e-12`

**Query Syntax**
- `CALCulate<cnum>:TDR:EYE:INPut:RTIMe:DATA?`

**Return Type**
- Double

**Default**
- 35E-12

---

**CALCulate<cnum>:TDR:EYE:INPut:RTIMe:THReshold <enum>**
**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the rise time threshold for the simulated eye.

**Parameters**

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<enum>` Threshold levels. Choose from:
  - **T1_9** - 10% to 90%.
  - **T2_8** - 20% to 80%.

**Examples**

```
CALC:TDR:EYE:INP:RTIM:THR T1_9
```
```
calculate2:tdr:eye:input:rtime:threshold t1_9
```

**Query Syntax**

`CALCulate<cnum>:TDR:EYE:INPut:RTIMe:THReshold?`

**Return Type** String

**Default** T1_9

---

**CALCulate<cnum>:TDR:EYE:INPut:ZLEVel <value>**

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the voltage level for bit "0" for the simulated eye function.

**Parameters**

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>` Voltage level for bit "0". The range is -5 V to +5 V.

**Examples**

```
CALC:TDR:EYE:INP:ZLEV 100E-3
```
```
calculate2:tdr:eye:input:zlevel 100e-3
```

**Query Syntax**

`CALCulate<cnum>:TDR:EYE:INPut:ZLEVEL?`

**Return Type** Double

**Default** 0

---

**CALCulate<cnum>:TDR:EYE:MASK:FAIL?**
**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-only) This command returns the mask test result.

**Parameters**

- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

**Examples**

- `CALC:TDR:EYE:MASK:FAIL?`
- `calculate2:tdr:eye:mask:fail?`

**Return Type**

Boolean

- ON or 1 - Mask test fail.
- OFF or 0 - Mask test pass.

**Default**

Not Applicable

---

**CALCulate<cnum>:TDR:EYE:MASK:STATe <bool>**

**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) Turns the mask test function state with simulated eye ON or OFF.

**Parameters**

- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<bool>`: ON or 1 - Turns mask test ON.  
  OFF or 0 - Turns mask test OFF.

**Examples**

- `CALC:TDR:EYE:MASK:STAT ON`
- `calculate2:tdr:eye:mask:state off`

**Query Syntax**

`CALCulate<cnum>:TDR:EYE:MASK:STATe?`

**Return Type**

Boolean

**Default**

OFF

---

**CALCulate<cnum>:TDR:EYE:RESults:DATA?**

**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-only) This command returns the results of the eye measurement. There are 18 values returned. The minimum and maximum values are returned in addition to the displayed results (16 values) on the TDR application GUI.

**Parameters**

- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

**Examples**

- `CALC:TDR:EYE:RESults:DATA?`
- `calculate2:tdr:eye:results:data?`

**Return Type**

Variant Array

**Default**

Not Applicable
CALCulate<cnum>:TDR:EYE:RESults:DISPlay:STATe <bool>

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* Turns the overlay ON or OFF.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<bool>`: ON or 1 - Turns overlay ON.
  
  OFF or 0 - Turns overlay OFF.

**Examples**
- `CALC:TDR:EYE:RES:DISP:STAT ON`
- `CALCulate2:tdr:eye:results:display:state off`

**Query Syntax**
- `CALCulate<cnum>:TDR:EYE:RESulsts:DISPlay:STATe?`

**Return Type**
- Boolean

**Default**
- ON

CALCulate<cnum>:TDR:EYE:RESults:THReshold <enum>

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the rise time threshold level for the results of eye measurement.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<enum>`: Threshold levels. Choose from:
  - **T1_9** - 10% to 90%.
  - **T2_8** - 20% to 80%.

**Examples**
- `CALC:TDR:EYE:RES:THR T1_9`
- `CALCulate2:tdr:eye:results:threshold t1_9`

**Query Syntax**
- `CALCulate<cnum>:TDR:EYE:RESulsts:THReshold?`

**Return Type**
- String

**Default**
- T1_9

CALCulate<cnum>:TDR:EYE:STATe <bool>
**Applicable Models:** All with TDR Options (S9x011A/B)

**(Read-Write)** Turns the Eye/Mask window ON or OFF.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<bool>`: ON or 1 - Turns Eye/Mask window ON.
  OFF or 0 - Turns Eye/Mask window OFF.

**Examples**
- `CALC:TDR:EYE:STAT ON`
- `calculate2:tdr:eye:state off`

**Query Syntax**
- `CALCulate<cnum>:TDR:EYE:STATe?`

**Return Type**
- Boolean
- Default: OFF

**CALCulate<cnum>:TDR:MEASure[1-256]:ACTive:MARKer <value>**

**Applicable Models:** All with TDR Options (S9x011A/B)

**(Read-Write)** This command sets active marker number.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>`: Marker number to activate. The range is 0 to 10.

**Examples**
- `CALC:TDR:MEAS1:ACT:MARK 1`
- `calculate2:tdr:measure1:active:marker 1`

**Query Syntax**
- `CALCulate<cnum>:TDR:MEASure[1-256]:ACTive:MARKer?`

**Return Type**
- Integer
- Default: 0

**CALCulate<cnum>:TDR:MEASure[1-256]:DTIMe:DATA?**

**Applicable Models:** All with TDR Options (S9x011A/B)

**(Read-only)** This command returns the delta time result value. You can get the result even if `CALCulate:TDR:MEASure:DTIMe:STATe` is off.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

**Examples**
- `CALC:TDR:MEAS1:DTIM:DATA?`
- `calculate2:tdr:measure1:dtime:data?`

**Return Type**
- Double
- Default: Not Applicable
**CALCulate\(<\text{cnum}\>):TDR:MEASure[1-256]:DTIMe:POSition \(<\text{value}\>)**

**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) This command sets delta time reference position.

**Parameters**

- **\(<\text{cnum}\)>** Channel number of the measurement. If unspecified, \(<\text{cnum}\)> is set to 1.
- **\(<\text{value}\)>** Delta time reference position. The range is 0 to 100.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:TDR:MEAS1:DTIM:POS 0</code></td>
<td><code>calculate2:tdr:measure1:dtime:position 0</code></td>
</tr>
</tbody>
</table>

**Query Syntax**

`CALCulate\(<\text{cnum}\>):TDR:MEASure[1-256]:DTIMe:POSition?`

**Return Type**

Double

**Default**

50

---

**CALCulate\(<\text{cnum}\>):TDR:MEASure[1-256]:DTIMe:STATe \(<\text{bool}\>)**

**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) Turns the delta time marker in the marker search ON or OFF.

**Parameters**

- **\(<\text{cnum}\)>** Channel number of the measurement. If unspecified, \(<\text{cnum}\)> is set to 1.
- **\(<\text{bool}\)>** ON or 1 - Turns delta time marker ON.
  
  OFF or 0 - Turns delta time marker OFF.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

**Query Syntax**

`CALCulate\(<\text{cnum}\>):TDR:MEASure[1-256]:DTIMe:STATe?`

**Return Type**

Boolean

**Default**

OFF

---

**CALCulate\(<\text{cnum}\>):TDR:MEASure[1-256]:DTIMe:TARGet \(<\text{value}\>)**
**Applicable Models:** All with TDR Options (S9x011A/B)

**(Read-Write)** This command sets the target trace number for the delta time function. The MEASure[1-256] is the trace number starting point for delta time. The <value> is the trace number stopping point for delta time.

**Parameters**
- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>` Trace number stopping point for delta time. The range is 1 to 16.

**Examples**
- `CALC:TDR:MEAS1:DTIM:TARG 5`
- `calculate2:tdr:measure1:dtime:target 5`

**Query Syntax**
- `CALCulate<cnum>:TDR:MEASure[1-256]:DTIMe:TARGet?`

**Return Type**
- Integer

**Default**
- 1

---

**CALCulate<cnum>:TDR:MEASure[1-256]:FORMat <enum>**

**Applicable Models:** All with TDR Options (S9x011A/B)

**(Read-Write)** This command sets the trace format.

**Parameters**
- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<enum>` Trace format.

For S-Parameter measurements, choose from:
- MLINear
- MLOGarithmic
- PHASe
- UPHase
- IMAGinary
- REAL
- POLar
- SMITh
- SADMittance
For Time Domain measurements, choose from:

- IMPedance
- VOLT
- MLOGarithmic
- MLINear
- REAL

Examples:

```
CALC:TDR:MEAS1:FORM IMP
calculate2:tdr:measure1:format impedance
```

Query Syntax:

```
CALCulate<cnum>:TDR:MEASure[1-256]:FORMat?
```

Return Type:

String

Default:

MLINear

```
CALCulate<cnum>:TDR:MEASure[1-256]:MARKer:REference[:STATe] <ON | OFF>
```
**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* Turns the reference marker ON or OFF.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<ON | OFF>` **ON** (or 1) - turns reference marker ON  
  **OFF** (or 0) - turns reference marker ON

**Examples**

```
CALC:TDR:MEAS1:MARK:REF ON
```

**Query Syntax**

`CALCulate<cnum>:TDR:MEASure[1-256]:MARKer:REFerence[:STATe]?

**Return Type**

Boolean

**Default**

Off

---

**CALCulate<cnum>:TDR:MEASure[1-256]:MARKer<mkr>:STATe <ON|OFF>**

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* Turns the specified marker ON or OFF.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 15; if unspecified, value is set to 1.
- `<ON|OFF>` **ON** (or 1) - turns marker ON.  
  **OFF** (or 0) - turns marker OFF.

**Examples**

```
CALC:TDR:MEAS1:MARK ON
calculate2:tdr:measure1:marker8 on
```

**Query Syntax**

`CALCulate<cnum>:TDR:MEASure[1-256]:MARKer<mkr>:STATe?`

**Return Type**

Boolean

**Default**

Off

---

`CALCulate<cnum>:TDR:MEASure[1-256]:PARameter <string>`
Applicable Models: All with TDR Options (S9x011A/B)

**Parameters**

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<sting>` Measurement parameter.

For S-Parameter measurements:

- Sxy
- Sddy
- Sdcxy
- Scdxy
- Sccxy
- X: 1 to 4
- Y: 1 to 4

For Time Domain measurements:

- Txy
- Tddy
- Tdcxy
- Tcdxy
- Tccxy

**Examples**

```
CALC:TDR:MEAS1:PAR T11
calculate2:tdr:measure1:parameter t11
```

**Query Syntax**

CALCulate<cnum>:TDR:MEASure[1-256]:PARameter?

**Return Type**

String

**Default**

S11

CALCulate<cnum>:TDR:MEASure[1-256]:PEELing:STATe <bool>
**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) This command sets state for the peeling function.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<bool>`: ON or 1 - Turns peeling function ON.
  OFF or 0 - Turns peeling function OFF.

**Examples**
- `CALC:TDR:MEAS1:PEEL:STAT ON`
- `calculate2:tdr:measure1:peeling:state off`

**Query Syntax**
`CALCulate<cnum>:TDR:MEASure[1-256]:PEELing:STATe?`

**Return Type**
Boolean

**Default**
OFF

**CALCulate<cnum>:TDR:MEASure[1-256]:SMOothing:STATe <bool>**

**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) This command sets state for the smoothing function.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<bool>`: ON or 1 - Turns smoothing function ON.
  OFF or 0 - Turns smoothing function OFF.

**Examples**
- `CALC:TDR:MEAS1:SMO:STAT ON`
- `calculate2:tdr:measure1:smoothing:state off`

**Query Syntax**
`CALCulate<cnum>:TDR:MEASure[1-256]:SMOothing:STATe?`

**Return Type**
Boolean

**Default**
OFF

**CALCulate<cnum>:TDR:MEASure[1-256]:TIME:IMPulse:WIDTh <value>**

**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) This command sets the impulse width value for the transform function.

**Parameters**
- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>`: Transform function impulse width value.

**Examples**
- `CALC:TDR:MEAS1:TIME:IMP:WIDT 17E-12`
- `calculate2:tdr:measure1:time:impulse:width 17e-12`

**Query Syntax**
`CALCulate<cnum>:TDR:MEASure[1-256]:TIME:IMPulse:WIDTh?`

**Return Type**
Double

**Default**
0
CALCulate<cnum>:TDR:MEASure[1-256]:TIME:STEP:COUPle <bool>

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command enables/disables rise time coupling.

**Parameters**
- **<cnum>** Channel number of the measurement. If unspecified, <cnum> is set to 1.
- **<bool>**
  - ON or 1 - Turns rise time coupling ON.
  - OFF or 0 - Turns rise time coupling OFF.

**Examples**

```
CALC:TDR:MEAS:TIME:STEP:COUP ON
calculate2:tdr:meas:time:step:couple_off
```

**Query Syntax**
CALCulate<cnum>:TDR:MEAS:TIME:STEP:COUPle?

**Return Type** Boolean

**Default** ON

---

CALCulate<cnum>:TDR:MEASure[1-256]:TIME:STEP:RTIMe <value>

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets rise time value for the transform function.

**Parameters**
- **<cnum>** Channel number of the measurement. If unspecified, <cnum> is set to 1.
- **<value>** Transform function rise time value in seconds.

**Examples**

```
CALC:TDR:MEAS1:TIME:STEP:RTIM 0
calculate2:tdr:measure1:time:step:rtime 0
```

**Query Syntax**
CALCulate<cnum>:TDR:MEASure[1-256]:TIME:STEP:RTIMe?

**Return Type** Double

**Default** 0

---

CALCulate<cnum>:TDR:MEASure[1-256]:TIME:STEP:RTIMe:THReshold <enum>
**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the rise time threshold level for the results of eye measurement.

**Parameters**
- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<enum>` Transform function rise time threshold levels. Choose from:
  - **T1_9** - 10% to 90%.
  - **T2_8** - 20% to 80%.

**Examples**
```
CALC:TDR:MEAS1:TIME:STEP:RTIM:THR T1_9
```
```
calculate2:tdr:measure1:time:step:rtime:threshold t1_9
```

**Query Syntax**
```
CALCulate<cnum>:TDR:MEASure[1-256]:TIME:STEP:RTIMe:THReshold?
```

**Return Type** String

**Default** T1_9

---

**CALCulate<cnum>:TDR:MEASure[1-256]:TIME:TYPE <enum>**

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the stimulus type for the transform function.

**Parameters**
- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<enum>` Transform function stimulus type. Choose from:
  - **LPSTep** - Low pass step.
  - **LPIMpulse** - Low pass impulse.

**Examples**
```
CALC:TDR:MEAS1:TIME:TYPE LPST
```
```
calculate2:tdr:measure1:time:type lpstep
```

**Query Syntax**
```
CALCulate<cnum>:TDR:MEASure[1-256]:TIME:TYPE?
```

**Return Type** String

**Default** LPSTep

---

**CALCulate<cnum>:TDR:MEASure[1-256]:TTIMe:DATA?**
**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-only)* This command returns the rise time result value for marker search. You can get the data even if CALCulate:TDR:MEASure:TTIMe:STATe is off.

**Parameters**
- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

**Examples**
- `CALC:TDR:MEAS1:TTIM:DATA?`
- `calculate2:tdr:measure1:ttime:data?`

**Return Type** Double

**Default** Not Applicable

**CALCulate<cnum>:TDR:MEASure[1-256]:TTIMe:STATe <bool>**

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* This command displays the rise time marker.

**Parameters**
- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<bool>` ON or 1 - Turns rise time marker ON.
- OFF or 0 - Turns rise time marker OFF.

**Examples**
- `CALC:TDR:MEAS1:TTIM:STAT ON`
- `calculate2:tdr:measure1:ttime:state off`

**Query Syntax** `CALCulate<cnum>:TDR:MEASure[1-256]:TTIMe:STATe?`

**Return Type** Boolean

**Default** OFF

**CALCulate<cnum>:TDR:MEASure[1-256]:TTIMe:THReshold <enum>**
**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) This command sets the rise time threshold for the rise time in the marker search function.

**Parameters**

- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<enum>`: Rise time threshold levels. Choose from:
  - **T1_9**: 10% to 90%.
  - **T2_8**: 20% to 80%.

**Examples**

```
CALC:TDR:MEAS1:TTIM:THR T1_9
```

```
calculate2:tdr:measure1:ttime:threshold t1_9
```

**Query Syntax**

```
CALCulate<cnum>:TDR:MEASure[1-256]:TTIme:THReshold?
```

**Return Type**

String

**Default**

T1_9

---

**CALCulate<cnum>:TDR:TIME:COUPle <bool>**

**Applicable Models:** All with TDR Options (S9x011A/B)

(Read-Write) This command enables/disables time coupling.

**Parameters**

- `<cnum>`: Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<bool>`: ON or 1 - Turns time coupling ON.
  OFF or 0 - Turns time coupling OFF.

**Examples**

```
CALC:TDR:TIME:COUP ON
calculate2:tdr:time:couple off
```

**Query Syntax**

```
CALCulate<cnum>:TDR:TIME:COUPlE?
```

**Return Type**

Boolean

**Default**

ON

---

**CALCulate<cnum>:TDR:TIME:STEP:AMPLitude <value>**
Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets or gets the step amplitude value.

Parameters

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>` Step amplitude value in Hz, dBm, or seconds. The range is 0.001 to 5.

Examples

```
CALC:TDR:TIME:STEP:AMPL 200ms
calculate2:tdr:time:step:amplitude .05s
```

Query Syntax

CALCulate<cnum>:TDR:TIME:STEP:AMPLitude?

Return Type

Double

Default

0.2
CALC:HOLD Commands

Controls the Trace Hold settings.

These commands are Superseded by the CALCulate:MEASure:HOLD commands.

**CALCulate:HOLD**

<table>
<thead>
<tr>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLEar</td>
</tr>
</tbody>
</table>

Click on a keyword to view the command details.

**see Also**

- Learn about Trace Hold
- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

**CALCulate<cnum>:HOLD:TYPE <char>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

**(Read-Write)** Sets the type of trace hold to perform.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>` Trace Hold type. Choose from:
  - **OFF** - Disables the Trace Hold feature.
  - **MINimum** - Sets Trace Hold to store the lowest measured data points.
**MAXimum** - Sets Trace Hold to store the highest measured data points.

**Examples**

```
CALC:HOLD:TYPE MAX
```

```
calculate2:hold:type minimum
```

**Query Syntax**  
CALCulate<ch>:HOLD:TYPE?

**Return Type**  
Character

**Default**  
OFF

---

CALCulate<ch>:HOLD:CLEar

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Write-only)* Resets the currently-stored data points to the live data trace and restarts the currently-selected Trace Hold type.

**See Critical Note**

**Parameters**

<chnum>  
Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

**Examples**

```
CALC:HOLD:CLE
```

```
calculate2:hold:clear
```

**Query Syntax**  
Not Applicable

**Default**  
Not Applicable
**Calculate:Transform Commands**

Specifies the settings for time domain transform.

These commands are **Superseded** by the `CALCulate:MEASure:TRANsform` commands.

<table>
<thead>
<tr>
<th>CALCulate:TRANsform</th>
</tr>
</thead>
<tbody>
<tr>
<td>COUple:</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
CALCulate<cnum>:TRANsform:COUPle:PARameters <num>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Specifies the time domain transform parameters to be coupled. The settings for those parameters will be copied from the selected measurement to all other measurements on the channel.

- To turn coupling ON and OFF, use SENS:COUP:PAR
- To specify Gating parameters to couple, use CALC:FILT:COUP:PAR

Learn more about **Time Domain Trace Coupling**

See Critical Note

**Parameters**

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<num>** (Numeric) Parameters to couple. To specify more than one parameter, add the numbers.

  1 - Transform Stimulus (Start, Stop, Center, and Span TIME settings.)
  
  2 - Transform State (ON / OFF)
  
  4 - Transform Window (Kaiser Beta / Impulse Width)
  
  8 - Transform Mode (Low Pass Impulse, Low Pass Step, Band Pass)
  
  16 - Transform Distance Marker Units
Examples

'To couple all parameters:
CALC:TRAN:COUP:PAR 31

'To couple Stimulus and Mode:
calculate2:transform:couple:parameters 9

Query Syntax
CALCulate<cnum>:TRANSform:COUPle:PARameters?

Return Type
Numeric

Default
29 (All parameters except 2 - Transform State)

CALCulate<cnum>:TRANSform:TIME:CENTer <num>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the center time for time domain measurements.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
<num> Center time in seconds; any number between:
± (number of points-1) / frequency span

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples
CALC:TRAN:TIME:CENT 1e-8
calculate2:transform:time:cen 15 ps

Query Syntax
CALCulate<cnum>:TRANSform:TIME:CENTer?

Return Type
Numeric

Default
0

CALCulate<cnum>:TRANSform:TIME:IMPulse:WIDTh <num>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the impulse width for the transform window.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Impulse width in seconds; Choose any number between: 
  0.6 / frequency span and 1.39 / frequency span

**Examples**

```
CALC:TRAN:TIME:IMP:WIDTh 10  
calculate2:transform:time:impulse:width 13
```

**Query Syntax**

CALCulate<cnum>:TRANsform:TIME:IMPulse:WIDTh?

**Return Type** Numeric

**Default** 0.98 / Default Span

---

CALCulate<cnum>:TRANsform:TIME:KBESsel <num>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the parametric window for the Kaiser Bessel window.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Window width for Kaiser Bessel in seconds; Choose any number between: 
  0.0 and 13.0

**Examples**

```
CALC:TRAN:TIME:KBES 10  
calculate2:transform:time:kbessel 13
```

**Query Syntax**

CALCulate<cnum>:TRANsform:TIME:KBESsel?

**Return Type** Numeric

**Default** 6

---

CALCulate<cnum>:TRANsform:TIME:LPFREQuency
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Write-only) Sets the start frequencies in LowPass Mode.

See Critical Note

Parameters

<cnun> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnun> is set to 1.

Examples

CALC:TRAN:TIME:LPFR
calculate2:transform:time:lpfrequency

Query Syntax

Not applicable

Default

Not applicable

CALCulate<cnun>:TRANsform:TIME:MARKer:MODE <char>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Specifies the measurement type in order to determine the correct marker distance.

- Select Auto for S-Parameter measurements.
- Select Reflection or Transmission for arbitrary ratio or unratioed measurements.

This setting affects the display of ALL markers for only the ACTIVE measurement.

Learn more about Distance Markers.

See Critical Note

Parameters

<cnun> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnun> is set to 1.

<char> Choose from:

AUTO If the active measurement is an S-Parameter, automatically chooses reflection or transmission. If non S-Parameter measurements, reflection is chosen.

REFLection Displays the distance from the source to the receiver divided by two (to compensate for the return trip.)

TRANsmission Displays the distance from the source to the receiver.
CALCulate<cnum>:TRANSform:TIME:MARKer:UNIT <char>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Specifies the unit of measure for the display of marker distance values. This settings affects the display of ALL markers for only the ACTIVE measurement (unless Distance Maker Units are coupled using `CALC:TRAN:COUP:PAR`.

Learn more about Distance Markers.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>` Choose from:
  - METRs
  - FEET
  - INCHes

**Examples**

- `CALC:TRAN:TIME:MARK:UNIT INCH`
- `calculate2:transform:time:marker:unit feet`

**Query Syntax**

CALCulate<cnum>:TRANSform:TIME:MARKer:UNIT?

**Return Type** Character

**Default** METRs

---

CALCulate<cnum>:TRANSform:TIME:SPAN <num>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the span time for time domain measurements.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

- `<num>` Span time in seconds; any number between: 0 and `2* [(number of points-1) / frequency span]`

**Note:** This command will accept **MIN** or **MAX** instead of a numeric parameter. See **SCPI Syntax** for more information.

**Examples**

```
CALC:TRAN:TIME:SPAN 1e-8
CALC:TRAN:TIME:SPAN maximum
```

**Query Syntax**

CALCulate<cnum>:TRANsform:TIME:SPAN?

**Return Type** Numeric

**Default** 20 ns

---

**CALCulate<cnum>:TRANsform:TIME:STARt <num>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the start time for time domain measurements.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

- `<num>` Start time in seconds; any number between: 
  `± (number of points-1) / frequency span`

**Note:** This command will accept **MIN** or **MAX** instead of a numeric parameter. See **SCPI Syntax** for more information.

**Examples**

```
CALC:TRAN:TIME:STAR 1e-8
CALC:TRAN:TIME:STAR minimum
```

**Query Syntax**

CALCulate<cnum>:TRANsform:TIME:STARt?

**Return Type** Numeric
CALCulate<cnum>:TRANsform:TIME:STATe <ON | OFF>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Turns the time domain transform capability ON or OFF.

See Critical Note

Note: Sweep type must be set to Linear Frequency in order to use Time Domain Transform.

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <ON|OFF> ON (or 1) - turns time domain ON.
  OFF (or 0) - turns time domain OFF.

Examples

- CALC:TRAN:TIME:STAT ON
- calculate2:transform:time:state off

Query Syntax

- CALCulate<cnum>:TRANsform:TIME:STATe?

Return Type

- Boolean (1 = ON, 0 = OFF)
- Default OFF

CALCulate<cnum>:TRANsform:TIME:STOP <num>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets the stop time for time domain measurements.

See Critical Note

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Stop time in seconds; any number between:

\[ \pm \frac{(number \ of \ points - 1)}{frequency \ span} \]

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

- CALC:TRAN:TIME:STOP 1e-8
- calculate2:transform:time:stop maximum
CALCulate<cnum>:TRANsform:TIME:STEP:RTIMe <num>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the step rise time for the transform window.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Rise time in seconds; Choose any number between: `.45 / frequency span` and `1.48 / frequency span`

**Examples**

```
CALC:TRAN:TIME:STEP:RTIM 1e-8
calculate2:transform:time:step:rtime 15 ps
```

**Query Syntax**

CALCulate<cnum>:TRANsform:TIME:STEP:RTIMe?

**Return Type** Numeric

**Default** `.99 / Default Span`

---

CALCulate<cnum>:TRANsform:TIME:STIMulus <char>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the type of simulated stimulus that will be incident on the DUT.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>` Choose from:
  - **STEP** - simulates a step DUT stimulus
  - **IMPulse** - simulates a pulse DUT stimulus

**Examples**

```
STEP can ONLY be used when CALC:TRAN:TIME:TYPE is set to LPASs (Lowpass). (STEP cannot be used with TYPE = BPASs.)
```
:STIM STEP will set :TYPE to LPASs

:TYPE BPASs will set :STIM to IMPulse

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

Query Syntax

CALCulate<cnum>:TRANsform:TIME:STIMulus?

Return Type

Character

Default

IMPulse

CALCulate<cnum>:TRANsform:TIME:ALIGnment <enum>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Selects the way the PNA computes the DC value of the frequency-domain measurement. The correct DC value is required for inverse-FFT accuracy, and if not estimated properly, can cause distortions in the time-domain measurement in the form of an undesired slope in the waveform.

See Critical Note

Parameters

- **<cnum>**
  - Channel number of the measurements to be listed. If unspecified, <cnum> is set to 1.

- **<enum>**
  - Choose from:

    - **LEGacy** - The DC value is extrapolated using three data points. The transform offset is calculated using the delay of the first frequency point. This is the same algorithm used in the HP 8510 network analyzer.

    - **NORMalize** - The DC value is extrapolated using three data points. The transform offset is set to zero at t=0 minus six rise-times. This mode requires that a good S-parameter calibration has been performed, which can be verified by observing a flat time-domain response at t=0 when measuring a load located at the physical point corresponding to t=0. Setting the time domain trace to zero at a time before t=0 stabilizes the trace for determining impedances after time t=0, resulting in improved behavior compared to Legacy mode. This method is similar to that used with PLTS, and is very useful in determining the time-domain-transform response of transmission lines and printed-circuit-board characteristics.

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>
CALCulate<cnum>:TRANSform:TIME[:TYPE] <char>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the type of time domain measurement.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>` Type of measurement. Choose from:
  - **LPASs** - Lowpass; Must also send `CALC:TRAN:TIME:LPFRequency` before calibrating.
  - **BPASs** - Bandpass;

BPASs can only be used when `CALC:TRAN:TIME:STIM` is set to IMPulse. (BPASs cannot be used with :STIM = STEP)

**:STIM STEP** will set :TYPE to **LPASs**

**:TYPE BPASs** will set :STIM to **IMPulse**

**Examples**

```
CALC:TRAN:TIME LPAS
calculate2:transform:time:type bpas
```

**Query Syntax**

CALCulate<cnum>:TRANSform:TIME[:TYPE]?

**Return Type**

- **Enumeration**
- **Default** LEGacy

**Default**

- **Character** BPAS
CALCulate:Uncertainty Commands

Controls the trace settings for Dynamic Uncertainty for S-Parameters.

These commands are Superseded by the CALCulate:MEASure:UNCertainty commands.

### Trace Properties

<table>
<thead>
<tr>
<th>Trace Property</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trace Type</td>
<td>CALC:UNC:DISP:TYPE</td>
</tr>
<tr>
<td>Coverage factor (95%)</td>
<td>CALC:UNC:DISP:CFAC</td>
</tr>
<tr>
<td>Noise</td>
<td>CALC:UNC:MOD:NOIS</td>
</tr>
<tr>
<td>Fluctuability</td>
<td>CALC:UNC:MODE:CABL:REP</td>
</tr>
<tr>
<td>Calibration</td>
<td>CALC:UNC:MODE:ETER</td>
</tr>
</tbody>
</table>

- **Apply to all traces**: None
- **Add Trace**: None
- **Save Data**: CALC:UNCertainty:SAVE

**See Also**

- See other Dynamic Uncertainty commands
- Learn more about Dynamic Uncertainty

**Critical Note**: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

```
CALCulate<ch>:UNCertainty:DISPlay:TYPE <char>
```
Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the display type for uncertainties for the selected measurement trace.

See Critical Note

Parameters

<ch> Channel number of the measurement to display uncertainties. If unspecified, value is set to 1.

<char> Display type. Choose from:

**NONE** - Display the trace without uncertainties.

**MAXimum** – Display the trace as the uncertainty maximum (measured or memory data + upper limit uncertainty values). Not supported with Smith Chart or Polar display format.

**MINimum** - Display the trace as the uncertainty minimum (measured or memory data - lower limit uncertainty values). Not supported with Smith Chart or Polar display format.

**BAR** – Display the uncertainties as “error bars” around the trace. Not supported with Smith Chart or Polar display format.

**SHADe** – Display the uncertainties as a shaded region around the trace. Not supported with Smith Chart or Polar display format.

**ELLipse** – Display the uncertainties in ellipse form. Supported only in Smith Chart or Polar display format.

Examples

```
CALC1:UNC:DISP:TYPE BAR
```

Query Syntax

CALCulate<ch>:UNCertainty:DISPlay:TYPE?

Return Type

Character

Default

NONE

CALCulate<ch>:UNCertainty:DISPlay:CFACtor <value>
Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the coverage factor (sigma) value to apply to the displayed uncertainty for the selected measurement trace. Coverage Factor corresponds to the level of confidence used in computing the specified measurement uncertainties.

See Critical Note

Parameters

<ch> Channel number of the measurement to display uncertainties. If unspecified, value is set to 1.

)value> Integer. Choose from:

<table>
<thead>
<tr>
<th>Coverage Factor</th>
<th>Approximate confidence level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>67%</td>
</tr>
<tr>
<td>2</td>
<td>95%</td>
</tr>
<tr>
<td>3</td>
<td>99%</td>
</tr>
<tr>
<td>4</td>
<td>&gt;99%</td>
</tr>
</tbody>
</table>

Examples CALC1:UNC:DISP:CFAC 2

Query Syntax CALCulate<ch>:UNCertainty:DISPlay:CFACtor?

Return Type Numeric

Default 2

CALCulate<ch>:UNCertainty:MODE:CABLe:REPeat <bool>

Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns whether the cable/connection repeatability contribution is currently included in the uncertainty values for the selected measurement trace.

See Critical Note

Parameters

<ch> Channel number of the measurement to display uncertainties. If unspecified, value is set to 1.

<bool> Choose from:

OFF (or 0) - Cable repeatability is NOT included.

ON (or 1) - Cable repeatability IS included.
CALCulate<ch>:UNCertainty:MODE:CABLe:REPeat?

Return Type: Boolean
Default: ON

CALCulate<ch>:UNCertainty:MODE:NOISe <bool>

Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns whether the noise contribution is currently included in the uncertainty values for the selected measurement trace.

See Critical Note

Parameters:

<ch> Channel number of the measurement to display uncertainties. If unspecified, value is set to 1.

<bool> Choose from:

OFF (or 0) - Noise contribution is NOT included.

ON (or 1) - Noise contribution IS included.

Examples: CALC1:UNC:MODE:NOIS ON

Query Syntax: CALCulate<ch>:UNCertainty:MODE:NOISe?

Return Type: Boolean
Default: ON

CALCulate<ch>:UNCertainty:MODE:ETERm <bool>
Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns whether the uncertainties associated with the correction error terms are being included in the uncertainty values for the selected measurement trace.

See Critical Note

Parameters

<ch> Channel number of the measurement to display uncertainties. If unspecified, value is set to 1.

<bool> Choose from:

OFF (or 0) - Error term uncertainty contribution is NOT included.

ON (or 1) - Error term uncertainty contribution IS included.

Examples

CALC1:UNC:MODE:ETER ON

Query Syntax

CALCulate<ch>:UNCertainty:MODE:ETERm?

Return Type

Boolean

Default

ON

CALCulate<ch>:UNCertainty:SAVE '"x,y,z'','<filename>

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Saves all three uncertainty data formats from the active measurement for the specified ports.

- This command is valid ONLY with Dynamic Uncertainty Calibration applied to the active measurement.
- Data that is not available is zero-filled.
- For sweeps with a large number of data points, always follow this command with *OPC? Learn more.

See Critical Note

Parameters

<ch> Channel number of the uncertainty measurement. If unspecified, value is set to 1.

"x,y,z" String Comma or space delimited port numbers for which data is requested, enclosed in quotes.

<filename> String Path, filename, and suffix of location to store the uncertainty data,
enclosed in quotes. The suffix is not checked for accuracy.

- **(".up") S-parameter Uncertainty File** - If saving 2 ports, specify "filename.u2p"; If saving 4 ports, specify "filename.u4p.", and so forth.
- **(".dsd") S-parameter Data Standard Definition file** - Data for Only one or two ports is allowed.
- **(".sdatcv") METAS S-parameter Covariance File.**

**Examples**

All three examples save the active uncertainty measurement on channel 1

'Saves ports 2 and 3 to .u2p file

CALC1:UNC:SAVE "2,3","C:\myData.u2p"

'Saves ports 1 and 2 to .dsd file

CALC1:UNC:SAVE "1,2","C:\myData.dsd"

'Saves ports 1 through 4 to .sdatcv file

CALC1:UNC:SAVE "1,2,3,4","C:\myData.sdatcv"

**Query Syntax**

Not Applicable

**Default**

Not Applicable
CALCulate:X (Axis) Commands

Controls the display of X-axis for various measurements.

These commands are **Superseded** by the CALCulate:MEASure:X commands.

```text
CALCulate:X:
  AXIS
    | :DOMain
    [:VALues]
```

Click on a keyword to view the command details.

**See Also**

- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using `Calc:Par:MNUM` or `Calc:Par:Select`. Learn more.

CALCulate<ch>:X:AXiS <string>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

**(Write-Read)** Sets the X-axis of the selected measurement to a DC Source. This command does not change the default setting for new traces.

**Parameters**

- `<ch>` Channel number of the selected measurement. If unspecified, value is set to 1.
- `<string>` String - (Not case-sensitive) For all channels EXCEPT DIQ, choose from the following:

  - **"Default"** - The default X-axis setting for the selected measurement. For Application measurements, the X-Axis domain is set with specific commands.
  - **"AO1"** - Internal DC source #1
"AO2" - Internal DC source #2

**Note:** For DIQ channels, see `CALC:X:AXIS:DOMain`

**Examples**

- `CALC:X:AXIS 'Default'`
- `calculate:x:axis "AO1"

**Query Syntax**

`CALCulate<ch>:X:AXIS?`

**Return Type**

String

**Default**

"Default"

---

`CALCulate<ch>:X:AXIS:DOMain <string>`

**Applicable Models:** N522xB, N523xB, N524xB

*(Write-Read)* Sets and returns the X-Axis domain of the selected DIQ measurement.

**Parameters**

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.

**Choose one of these:**

<table>
<thead>
<tr>
<th>Choose one of these:</th>
<th>Then set X-Axis Source <em>(CALC:X:AXIS)</em> using one of these as the argument.</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Frequency&quot;</td>
<td>&quot;F1&quot;, &quot;F2&quot;, etc.</td>
</tr>
<tr>
<td>&quot;Power&quot;</td>
<td>Source port: &quot;Port 1&quot;, &quot;Port 2&quot;, etc.</td>
</tr>
<tr>
<td>&quot;Phase&quot;</td>
<td>Source port: &quot;Port 1&quot;, &quot;Port 2&quot;, etc.</td>
</tr>
<tr>
<td>&quot;DC&quot;</td>
<td>DC Source: &quot;AO1&quot;, &quot;AO2&quot;</td>
</tr>
<tr>
<td>&quot;Points&quot;</td>
<td>&quot;Points&quot;</td>
</tr>
</tbody>
</table>

**Example**

1. `CALC:X:AXIS:DOM "Power"
2. `CALC:X:AXIS "Port 1"

**Query Syntax**

`CALCulate<ch>:X:AXIS:DOMain?`

**Return Type**

String
**CALCulate\(<\text{nnum}>\):X[:\text{VALues}]\)?**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-only)* Returns the stimulus values for the selected measurement in the current units. You can select one measurement for each channel using `Calc:Par:MNUM` or `Calc:Par:Select`. Learn more.

This command can be used for all Measurement Classes.

**Note:** To avoid frequency rounding errors, specify `FORM:DATA <\text{Real},64>` or `\text{<ASCii, 0>}`

### Parameters

- **<\text{nnum}>**
  Any existing channel number; if unspecified, value is set to 1.

### Examples

1. `Calc:Par:Sel "MyGCATrace"
2. `CALC:X?`

### Return Type

- **Default** Not applicable
### CalPod Commands

The following commands are sent as a string argument from:

\[ \text{CONTrol:CALPod:COMMan} \text{d <string>} \]

<table>
<thead>
<tr>
<th>CALPod</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>AVERage</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DISable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENABle</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HIDE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INITialize</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACTive</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAUNch</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POWER</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECorrect</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACTive</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHOW</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEMP?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Click on a blue keyword to view the command details.

In addition to the above Calpod commands, the following IEE 488 Common Commands can also be sent as a string argument:

- **CLS** - Clears all errors and event data from the error/event queue.
- **IDN?** - Returns the instrument identification information.

For 4 state CALPod support the following is required.
- **CCT (configurable command table) version 1.4**
- **Controller FPGA version 8.4**

- **OPC?** - Operation complete query. This query immediately returns a value, independent of whether or not the operation is complete. A return value of 0 indicates the operation is not complete. A value of +1 indicates the operation is complete. Typically this command is used in a loop with a 0.25 second delay when waiting for an operation to complete.

- **TST?** - Performs a communication test on all the currently enabled Calpods. 0 = Test failed on one or more enabled Calpods. 1 = All enabled Calpods working.

- **SYSTem:ERRor?** - Queries the Event/Error queue and returns the most recent error element.

### Important Notes

- ALL commands on this page are sent as a string argument from: `CONTrol:CALPod:COMMa nd <string>`
- Use single quotes ONLY (NOT double quotes) for the CONT:CALP:COMM string arguments.
- Sending queries requires TWO question marks. See following note as example.
- To read errors with the commands on this page, use the Calpod query: 
  ```
  CONT:CALP:COMM? 'SYSTem:ERRor'
  ```

- ALL queries return strings.

### See Also

- **Example Programs**
- **Synchronizing the Analyzer and Controller**
- **SCPI Command Tree**

```
CALPod:AVERage < num >
```
Applicable Models: All

(Read-Write) Sets the number of OSL averages used during initialization and recorrect.

See important notes.

Parameters

<num> Number of OSL averages to apply. Use an integer between 1 and 100.

Examples

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONT:CALP:COMM 'CALP:AVER 8'</td>
<td>Sets the number of OSL averages to 8.</td>
</tr>
</tbody>
</table>

Query Syntax

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONT:CALP:COMM? 'CALP:AVER?'</td>
<td>Query the number of OSL averages.</td>
</tr>
</tbody>
</table>

Return Type

Numeric

Default

4

CALPod:DISable <port>

Applicable Models: All

(Write-only) Unassign Calpod serial number from the specified VNA port.

See important notes.

Parameters

<port> VNA port number to un-assign.

Examples

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONT:CALP:COMM 'CALP:DIS 2'</td>
<td>Unassign the Calpod from port 2.</td>
</tr>
</tbody>
</table>

Query Syntax

Not Applicable

Default

Not Applicable

CALPod:ENABle <port>,<sn>
(Write-read) Assign or return the Calpod serial number for the specified VNA port. If a Calpod module is already assigned to the specified VNA port, this assignment will replace the existing assignment.

See important notes.

**Parameters**

- **<port>** VNA port number to be assigned the Calpod serial number.
- **<sn>** Calpod serial number.

**Examples**

```
CONT:CALP:COMM 'CALP:ENAB 2, 0001234' 'WRITE
CONT:CALP:COMM 'CALP:ENAB? 2' 'READ
```

**Query Syntax**

CONTrol:CALPod:COMMand? 'CALPod:ENABle? <port>'

**Return Type** String

**Default** Not Applicable

---

**CALPod:HIDE**

Applicable Models: All

(Write-only) Hides the Calpod setup dialog.

See important notes.

**Parameters** None

**Examples**

```
CONT:CALP:COMM 'CALP:HIDE'
```

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**CALPod:INITialize:ACTive**

3825
**Applicable Models:** All

*Write-only* Performs the initialize process for the active (selected) channel. Select a channel using `CALCulate:MEASure:PARameter`.

See important notes.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td><code>CONT:CALP:COMM 'CALP:INIT:ACT'</code></td>
</tr>
<tr>
<td>Query Syntax</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**CALPod:INITialize:ALL**

**Applicable Models:** All

*Write-only* Performs the initialize ALL channels process.

See important notes.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td><code>CONT:CALP:COMM 'CALP:INIT:ALL'</code></td>
</tr>
<tr>
<td>Query Syntax</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**CALPod:LAUNch**

**Applicable Models:** All

*Write-only* Starts the Calpod software. The Calpod software can be started using this (Launch) command or by activating the Calpod user interface. Once the Calpod software is started it remains active until the VNA application is terminated.

Send this command first in your program, then wait a couple seconds while the software starts before sending the next command.

See important notes.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
</table>
| Examples   | `CONT:CALP:COMM 'CALP:LAUN'`
|            | `wait 3` |
**CALPod:POWer < 0 | 1 >**

**Applicable Models**: All

(Read-Write) Enables or disables the ‘Correct Power’ setting in the Calpod setup dialog.

See important notes.

**Parameters**
- 0 – disable power correction
- 1 – enable power correction

Note that ON and OFF cannot be used as arguments to this command.

**Examples**
- `CONT:CALP:COMM 'CALP:POW 1'`
- `CONT:CALP:COMM? 'CALP:POW?'`

**Query Syntax**
- `CONT:CALP:COMM 'CALP:POW 1'`
- `CONT:CALP:COMM? 'CALP:POW?'`

**Return Type**
- Numeric

**Default**
- 0 (disabled)

---

**CALPod:RECorrect:ACTive**

**Applicable Models**: All

(Write-only) Performs the recorrect process for the active (selected) channel. Select a channel using CALCulate:MEASure:PARameter.

See important notes.

**Parameters**
- None

**Examples**
- `CONT:CALP:COMM 'CALP:REC:ACT'`

**Query Syntax**
- Not Applicable

**Default**
- Not Applicable

---

**CALPod:RECorrect:ALL**
Applicable Models: All

(Write-only) Performs the recorrect process for ALL channels.

See important notes.

**Parameters** None

**Examples** CONT:CALP:COMM 'CALP:REC:ALL'

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**CALPod:SHOW**

Applicable Models: All

(Write-only) Shows the Calpod setup dialog.

See important notes.

**Parameters** None

**Examples** CONT:CALP:COMM 'CALP:SHOW'

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**CALPod:STATe <sn>,<state>**

Applicable Models: All

(Write-only) Sets the specified Calpod module to specified impedance state.

See important notes.

**Parameters**

<sn> Serial number of the Calpod module. When set to 1, all modules are set to the specified state.

<state> Impedance state. Not case sensitive. Choose from:

*Short*, *Open*, *Load*, or *Thru*

**Examples** CONT:CALP:COMM 'CALPod:STATE 0001234,thru'

**Query Syntax** Not Applicable
CALPod:TEMP? <sn>

**Applicable Models:** All

*(Read-only)* Returns the temperature of the specified Calpod module in degrees Celsius.

See important notes.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;sn&gt;</td>
<td>Serial number of the Calpod module.</td>
</tr>
</tbody>
</table>

**Examples**

```
CONT:CALP:COMM? 'CALPod:TEMP? 0001234'
```

**Query Syntax**

Not Applicable

**Return Type**

String

**Default**

Not Applicable
Control Commands

Specifies the settings to remotely control the rear panel connectors, an external test set, Calpod modules, and ECAL Module state.

```plaintext
CONTrol
  AUXiliary - More Commands
  CALPod:COMMand
  CHANnel:INTerface:CONTrol:
    | CONFig:RECall
    | [STATe]
  ECAL:MODule:
    | PATH:
      | COUNt?
      | STATe
    | STATe
  EXTernal:TESTset - More Commands
  HANDler - More Commands
  NOISe:SOURce[:STATe]
  PHASE
    | COUNt?
    | MODule
      | DATA
        | STORe
        | MODel?
        | SERial?
```
<table>
<thead>
<tr>
<th>SETup</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SIGNal:</strong></td>
</tr>
<tr>
<td>TRIGger</td>
</tr>
<tr>
<td>ATBA</td>
</tr>
<tr>
<td>OUTP</td>
</tr>
<tr>
<td>AIO</td>
</tr>
<tr>
<td>PIN</td>
</tr>
<tr>
<td>COUNt?</td>
</tr>
<tr>
<td><strong>FUNCtion</strong></td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>INPut:LEVel?</td>
</tr>
<tr>
<td>CHANnel</td>
</tr>
<tr>
<td><strong>FUNCtion</strong></td>
</tr>
<tr>
<td>CATalog</td>
</tr>
<tr>
<td><strong>KDMI</strong></td>
</tr>
<tr>
<td>MAIN</td>
</tr>
<tr>
<td>COUNt?</td>
</tr>
<tr>
<td>CHANnel:FUNCtion</td>
</tr>
<tr>
<td><strong>FUNCtion</strong></td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>INPut:LEVel?</td>
</tr>
<tr>
<td>LOGic</td>
</tr>
<tr>
<td><strong>SUB</strong></td>
</tr>
<tr>
<td><strong>COUNt?</strong></td>
</tr>
</tbody>
</table>
Click on a keyword to view the command details.

Blue command is superseded.

See Also

- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree
- See a pinout and detailed description of the rear panel connectors:
  - External Test Set IO connector
  - Material Handler IO connector

**CONTrol:CALPod:COMMand <string>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, E5080A/B, P937xA

**(Write-Read)** Sends commands that control a Calpod module. Reads query versions Calpod commands.

See ALL Calpod commands.

Learn more about Calpod.

**Parameters**

<string> Calpod command. See ALL Calpod commands that can be used in this string.

**Write Example**

```plaintext
'Enclose all strings in SINGLE quotes (NOT double quotes)
```
CONTroL:CALPod:COMMaNd? <string>

Relevant only for query strings.

Read Example

CONT:CALP:COMM? '*OPC?'

'returns 0 if the calpod software is currently processing an operation
'returns 1 if operations are complete

Return Type
String
Default Not Applicable

CONTroL:CHANnel:INTerface:CONTroL:CONFig:RECall[:STATe] <string>

Applicable Models: N522xB, N523xB, N524xB, E5080A

(Write-only) Recalls an Interface Control configuration file. Learn more about Interface Control.

Parameters
<string> File name and extension (.xml) of the configuration file to recall. Files are typically stored in the default folder "D:\". To recall from a different folder, specify the full path name.

Examples
CONT:CHAN:INT:CONT:CONF:REC 'MyConfigFile.xml'

Query Syntax
Not Applicable
Default Not Applicable

CONTroL:CHANnel:INTerface:CONTroL[:STATe] <bool>
Applicable Models: N522xB, N523xB, N524xB, E5080A/B

(Read-Write) Enables and disables ALL Interface Control settings. To send data, the individual interfaces must also be enabled. Learn more about Interface Control.

Parameters

<bool> Boolean

OFF (0) - Interface Control is disabled; NO control data is sent.

ON (1) - Interface Control is enabled.

Examples

```plaintext
CONT:CHAN:INT:CONT 1
control:channel:interface:control:state 0
```

Query Syntax

CONTrol:CHANnel:INTerface:CONTrol[:STATe]?

Return Type

Boolean

Default

OFF (0)

---

CONTrol:ECAL:MODule<num>:PATH:COUNt? <name>

Applicable Models: All

(Read-only) Returns the number of unique states that exist for the specified path name on the selected ECal module.

This command performs exactly the same function as SENS:CORR:CKIT:ECAL:PATH:COUNt?

Use the CONT:ECAL:MOD:PATH:STAT command to set the module into one of those states.

Use SENS:CORR:CKIT:ECAL:PATH:DATA? to read the data for a state.

Parameters

[num] Optional argument. USB number of the ECal module. If unspecified (only one ECal module is connected to the USB), <num> is set to 1. If two or more modules are connected, use SENS:CORR:CKIT:ECAL:LIST? to determine how many, and SENS:CORR:CKIT:ECAL:INF? to verify their identities.

<name> Name of the path for which to read number of states. Choose from:

Reflection paths

- A
- B
- C (4-port modules)
- D (4-port modules)

Transmission paths

- AB
- AC (4-port modules)
- AD (4-port modules)
- BC (4-port modules)
- BD (4-port modules)
- CD (4-port modules)

Note: For each transmission path, the first of the available states is the through state, the second is the confidence (attenuator) state.

Examples

```
CONT:ECAL:MOD:PATH:COUN? A
control:ecal:module2:path:count? cd
```

See example program

Return Type

- Integer

Default

- Not Applicable

**CONTrol:ECAL:MODule<num>:PATH:STATe <path>, <stateNum>**

Applicable Models: All

(Write-only) Sets the internal state of the selected ECAL module. This command supersedes CONT:ECAL:MOD:STAT .

- Use CONT:ECAL:MOD:PATH:COUN? to read the number of unique states that exist for the specified path name on the module.
- Use SENS:CORR:CKIT:ECAL:PATH:DATA? to read the data for a state (from the module memory) corresponding to the stimulus values of a channel.

Parameters
[num] Optional argument. USB number of the ECal module. If unspecified (only one ECal module is connected to the USB), <num> is set to 1. If two or more modules are connected, use SENS:CORR:CKIT:ECAL:LIST? to determine how many, and SENS:CORR:CKIT:ECAL:INF? to verify their identities.

<path> Path name for which to set a state.

**Note:** The impedance paths are not independent. For example, changing the impedance presented on path A will cause a change to the impedance on path B.

Choose from:

Reflection paths

- A
- B
- C (4-port modules)
- D (4-port modules)

Transmission paths

- AB
- AC (4-port modules)
- AD (4-port modules)
- BC (4-port modules)
- BD (4-port modules)
- CD (4-port modules)

<stateNum> Number of the state to set. Refer to the following table to associate the <stateNum> with a state in your ECal module.

In addition, CONT:ECAL:MOD:PATH:COUNt? returns the number of states in the specified ECal module.
<table>
<thead>
<tr>
<th>&lt;stateNum&gt;</th>
<th>N4432A and N4433A States</th>
<th>N4431A States</th>
<th>N469x and N755x States**</th>
<th>8509x States</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-Port Reflection States</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Open</td>
<td>Open</td>
<td>Impedance 1</td>
<td>Open</td>
</tr>
<tr>
<td>2</td>
<td>Short</td>
<td>Short</td>
<td>Impedance 2</td>
<td>Short</td>
</tr>
<tr>
<td>3</td>
<td>Impedance 1</td>
<td>Impedance 1</td>
<td>Impedance 3</td>
<td>Impedance 1</td>
</tr>
<tr>
<td>4</td>
<td>Impedance 2</td>
<td>Impedance 2</td>
<td>Impedance 4</td>
<td>Impedance 2</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td>Impedance 5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td>Impedance 6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td>Impedance 7</td>
<td></td>
</tr>
<tr>
<td>Two-Port Transmission States</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Thru</td>
<td>Thru</td>
<td>Thru</td>
<td>Thru</td>
</tr>
<tr>
<td>2</td>
<td>Confidence</td>
<td>Confidence</td>
<td>Confidence</td>
<td>Confidence</td>
</tr>
</tbody>
</table>

** The following modules have only FOUR Impedance states (1, 2, 3, 4): N4690B, N4691B, N4692A, N4696B, N7550A - N7556A.

Examples

CONT:ECAL:MOD:PATH:STATe A, 5
control:ecal:module2:state BC, 1

See example program

Query Syntax Not Applicable

Default Not Applicable

CONTrol:ECAL:MODule<num>:STATe <value> Superseded
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

This command is replaced with CONT:ECAL:MOD:PATH:STATe.

(Write-only) Sets the internal state of the selected ECAL module.

Parameters

[num] Optional argument. USB number of the ECAL module. If unspecified (only one ECAL module is connected to the USB), <num> is set to 1. If two or more modules are connected, use SENS:CORR:COLL:CKIT:INF? to verify their identity.

[value] Integer code for switching the module. The following are codes for Keysight ECAL modules.

<table>
<thead>
<tr>
<th>8509x Modules</th>
<th>Port A</th>
<th>Port B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Short</td>
<td>43</td>
<td>43</td>
</tr>
<tr>
<td>Load</td>
<td>33</td>
<td>33</td>
</tr>
<tr>
<td>Mismatch</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>Thru</td>
<td></td>
<td>84</td>
</tr>
<tr>
<td>Confidence</td>
<td></td>
<td>20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>N469x and N755x Modules</th>
<th>Port A</th>
<th>Port B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>36</td>
<td>33</td>
</tr>
<tr>
<td>Short</td>
<td>39</td>
<td>45</td>
</tr>
<tr>
<td>Load</td>
<td>37</td>
<td>37</td>
</tr>
<tr>
<td>Mismatch (Offset short)</td>
<td>53</td>
<td>53</td>
</tr>
<tr>
<td>Impedance 5 (Offset open)</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Impedance 6 (Offset short)</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>Impedance 7 (Offset short)</td>
<td>38</td>
<td>41</td>
</tr>
<tr>
<td>Thru</td>
<td></td>
<td>42</td>
</tr>
<tr>
<td>Confidence</td>
<td></td>
<td>40</td>
</tr>
</tbody>
</table>
### N4431A Modules

<table>
<thead>
<tr>
<th>State</th>
<th>Port A</th>
<th>Port B</th>
<th>Port C</th>
<th>Port D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>-1398</td>
<td>-1384</td>
<td>-2774</td>
<td>-2654</td>
</tr>
<tr>
<td>Short</td>
<td>-1350</td>
<td>-1381</td>
<td>-2582</td>
<td>-2642</td>
</tr>
<tr>
<td>Load</td>
<td>26985</td>
<td>-26986</td>
<td>-26986</td>
<td>26985</td>
</tr>
<tr>
<td>Mismatch</td>
<td>-26986</td>
<td>26985</td>
<td>26985</td>
<td>-26986</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Path</th>
<th>Thru</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB Path</td>
<td>-2590</td>
<td>-598</td>
</tr>
<tr>
<td>AC Path</td>
<td>-4011</td>
<td>85</td>
</tr>
<tr>
<td>AD Path</td>
<td>-2517</td>
<td>16042</td>
</tr>
<tr>
<td>BC Path</td>
<td>-1650</td>
<td>-598</td>
</tr>
<tr>
<td>BD Path</td>
<td>-4011</td>
<td>85</td>
</tr>
<tr>
<td>CD Path</td>
<td>-1352</td>
<td>16042</td>
</tr>
</tbody>
</table>

### N4432A and N4433A Modules

<table>
<thead>
<tr>
<th>State</th>
<th>Port A</th>
<th>Port B</th>
<th>Port C</th>
<th>Port D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>-6971</td>
<td>-11835</td>
<td>-14895</td>
<td>-14876</td>
</tr>
<tr>
<td>Short</td>
<td>-14395</td>
<td>-12859</td>
<td>-14899</td>
<td>-14905</td>
</tr>
<tr>
<td>Load</td>
<td>-14907</td>
<td>-14907</td>
<td>-14907</td>
<td>-14907</td>
</tr>
<tr>
<td>Offset Short</td>
<td>-9787</td>
<td>-6459</td>
<td>-14874</td>
<td>-14887</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Path</th>
<th>Thru</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB Path</td>
<td>13765</td>
<td>30069</td>
</tr>
<tr>
<td>AC Path</td>
<td>-10519</td>
<td>-2327</td>
</tr>
<tr>
<td>AD Path</td>
<td>-10538</td>
<td>-2346</td>
</tr>
<tr>
<td>BC Path</td>
<td>-5655</td>
<td>-1559</td>
</tr>
<tr>
<td>BD Path</td>
<td>-5674</td>
<td>-1578</td>
</tr>
<tr>
<td>CD Path</td>
<td>-15051</td>
<td>30069</td>
</tr>
</tbody>
</table>

### Examples

```
CONT:ECAL:MOD:STAT 36
control:ecal:module2:state 38
```

### Query Syntax

Not Applicable

### Default

Not Applicable
CONTrol:MULTiplexer<id>:OUTPut:<grp>[:DATA] <num>

**Applicable Models:** E5080A/B

*(Read-Write)* Sets or returns the output port data for specified group with id of the E5092A multiport test set.

**Notes:** This command is available only for E5092A multiport test set.

**Parameters**

- `<id>`  Id of the multiport test set either 1 or 2. If unspecified, Id is assumed to be 1.
- `<grp>` A | B | C | D
- `<num>` An integer specifying the decimal value of the control line. Values are obtained by adding weights from the following table that correspond to individual lines.

The output port data range is between 0 to 255 (0=All line are turns OFF and 255 all lines are turn ON).

<table>
<thead>
<tr>
<th>Line</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td>8</td>
<td>128</td>
</tr>
</tbody>
</table>

**Examples**

CONT:MULT1:OUTP:B 8

**Query Syntax**

CONTrol:MULTiplexer<id>:OUTPut:<grp>[:DATa]?

**Return Type**

Numeric

**Default** 0

CONTrol:MULTiplexer<id>:OUTPut:<grp>VOLTage[:DATA] <volt>

**Applicable Models:** E5080A/B
(Read-Write) Sets or returns the output voltage for specified group with id of the E5092A multiport test set.

Notes: This command is available only for E5092A multiport test set.

Parameters

<id>  Id of the multiport test set either 1 or 2. If unspecified, Id is assumed to be 1.
<grp> A | B | C | D
<volt> Output voltage range for <grp> is between 0 to 5.2V and resolution is 10mV.

Examples

CONT:MULT1:OUTP:B:VOLT 4.2

Query Syntax

CONTrol:MULTiplexer<id>:OUTPut:<grp>:VOLtage[:DATa]?

Return Type

Numeric

Default

0 V

CONTROL:NOISe:SOURce[:STATe] <bool>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Set and read the noise source (28V) ON and OFF.

Parameters

<bool> Boolean

OFF (0) - Noise Source OFF

ON (1) - Noise Source ON

Examples

CONT:NOIS:SOUR 1
control:noise:source:state 0

Query Syntax

CONTrol:NOISe:SOURce[:STATe]?

Return Type

Boolean

Default

For VNA models with a Noise Figure option (028/029/H29), the 28V line is ON at application start and after a preset. The ON/OFF state is also available from a VNA softkey menu.

For VNA models WITHOUT a Noise Figure option (028/029/H29), the 28V line is OFF at application start and its state is not affected by a preset. The ON/OFF state is NOT available from a VNA softkey menu.
**CONTrol:PHASe:COUNt?**

**Applicable Models:** All

*(Read only)* Reads the number of phase reference modules connected.

**Parameters** None

**Examples**

```
CONT:PHAS:COUN?
control:phase:count?
```

**Return Type** Numeric value

**Default** N/A

---

**CONTrol:PHASe:MODule[x]:DATA:STORe <path>**

**Applicable Models:** All

*(Write-only)* Reads the calibration data from the phase reference of the specified module (x), and writes the data to the Path.

**Parameters**

- `<path>` (String) Directory path to stored data.

**Examples**

```
CONT:PHAS:MOD1:DATA:STOR "D:MyPhaseRefCalData.csv"
control:phase:module1:data:store "D:MyPhaseRefCalData.csv"
```

**Return Type** N/A

**Default** N/A

---

**CONTrol:PHASe:MODule[x]:MODel?**

**Applicable Models:** All

*(Read only)* Reads the model number of the specified module (x).

**Parameters** None

**Examples**

```
CONT:PHAS:MOD1:MOD?
control:phase:module1:model?
```

**Return Type** String

**Default** N/A
CONTrol:PHASe:MODule[x]:SERial?

Applicable Models: All

(Read only) Reads the serial number of the specified module (x).

Parameters

Examples

```
CONT:PHAS:MOD1:SER?
control:phase:module1:serial?
```

Return Type Numeric

Default N/A

CONTrol:PHASe:MODule[x]:SETup <divideNumber>

Applicable Models: All

(Write-only) Turns on the phase reference, and sets the divide number.

Parameters

<divideNumber> Divide number can be 1, 2, 4, 8, 16.

Examples

```
CONT:PHAS:MOD1:SET 2
control:phase:module1:setup 2
```

Return Type Numeric

Default N/A

CONTrol:SIGNal <conn>,<char>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Configures external triggering in the VNA.

Note: To configure external triggering in the current VNA models, use the Trigger commands.

- To control BNC1 and BNC2 with this command, then you **MUST** have TRIG:PREF:AIGlobal = **ON**. Learn more
- Trigger:Sequence:Source is automatically set to External when **CONTrol:SIGNal** is sent.
- Edge triggering is only available on some Microwave VNA models.
- For more information, see External Triggering in the VNA.

Parameters
Rear Panel connector to send or receive trigger signals. Choose from:

**BNC1** Trigger IN from rear-panel Trigger IN BNC connector

*Note:* Only one of the input connectors is active at a time. When a command is sent to one, the VNA automatically makes the other INACTIVE.

**BNC2** Trigger OUT to rear-panel Trigger OUT BNC connector.

**MATHtrigger** - Trigger IN from rear-panel Material Handler connector Pin 18

**RDY** - Ready for trigger OUT.

- PNA-X: Meas Trig RDY
- PNA-L: Handler I/O p21 (Some models)

INACTIVE - Disables the specified connector **<conn>**.

Choose from ONLY the following when **<conn>** is set to BNC1 or AUXT or MATHtrigger:

- **TIENEGATIVE** - (Trigger In Edge Negative) - Triggers the VNA when receiving a negative going signal

- **TIEPOSITIVE** - (Trigger In Edge Positive) - Triggers the VNA when receiving a positive going signal

- **TILLOW** - (Trigger In Level Low) - Triggers the VNA when receiving a low level signal

- **TILHIGH** - (Trigger In Level High) - Triggers the VNA when receiving a High-level signal

Choose from ONLY the following when **<conn>** is set to BNC2:

Use CONTrol:SIGNal:TRIGger:OUTP to enable the BNC2 output.

The following selections send a positive or negative pulse before or after each trigger acquisition. This normally occurs each sweep unless a channel is in point trigger mode.

- **TOPPAFTER** - (Trigger Out Pulse Positive After) - Sends a POSITIVE going TTL pulse at the END of each trigger acquisition.

- **TOPPBEFORE** - (Trigger Out Pulse Positive Before) - Sends a POSITIVE going TTL pulse at the START of each trigger acquisition.
- **TOPNAFTER** - (Trigger Out Pulse Negative After) - Sends a NEGATIVE going TTL pulse at the END of each trigger acquisition.

- **TOPNBEFORE** - (Trigger Out Pulse Negative Before) - Sends a NEGATIVE going TTL pulse at the START of each trigger acquisition.

Choose from **ONLY the following** when <conn> is set to **RDY** :

- **LOW**  Outputs a TTL low when the VNA is ready for trigger. (Default setting)
- **HIGH** Outputs a TTL high when the VNA is ready for trigger.

### Examples

**CONT:SIGN BNC1,TIENEGATIVE**

**control:signal bnc2,toppbefore**

**CONT:SIGN RDY,LOW**

### Query Syntax

**CONTrol:SIGNal? <conn>**

In addition to the arguments listed above, the following is also a possible returned value:

**NAVAILABLE** - This feature is not available on this VNA

### Return Type

Character

**Default**

At Preset:

BNC1 = INACTIVE  
BNC2 = INACTIVE  
AUXT = TILHIGH

When Output is enabled :

BNC1 = INACTIVE  
BNC2 = TOPPAFTER  
AUXT = TILHIGH

**CONTrol:SIGNal:TRIGger:ATBA <bool>**
**Applicable Models:** N522xB, N523xB, N524xB, M980xA, P50xxA/B, P93xxB, E5080B

**Accept Trigger Before Armed** Determines what happens to an EDGE trigger signal if it occurs before the VNA is ready to be triggered. (LEVEL trigger signals are always ignored.) For more information, see External triggering.

**Parameters**

- **<bool>**
  - **OFF (0)** - A trigger signal is ignored if it occurs before the VNA is ready to be triggered.
  - **ON (1)** - A trigger signal is remembered and then used when the VNA becomes armed (ready to be triggered). The VNA remembers only one trigger signal.

**Examples**

```
CONT:SIGN:TRIG:ATBA 0
control:signal:trigger:atba ON
```

**Query Syntax**

`CONTrol:SIGNal:TRIGger:ATBA?`

**Return Type**

Boolean

**Default**

OFF

---

**CONTrol:SIGNal:TRIGger:OUTP <bool>**

**Applicable Models:** N522xB, N523xB, N524xB

**Output Enabled** The VNA can be enabled to send trigger signals out the rear-panel TRIGGER OUT BNC connector. Use CONTrol:SIGNal to configure for output triggers.

**Note:** To configure external triggering in the current VNA models, use the Trigger commands.

For more information, see External triggering.

**Parameters**

- **<bool>**
  - **OFF (0)** - VNA does NOT output trigger signals.
  - **ON (1)** - VNA DOES output trigger signals.

**Examples**

```
CONT:SIGN:TRIG:OUTP 1
control:signal:trigger:outp OFF
```

**Query Syntax**

`CONTrol:SIGNal:TRIGger:OUTP?`

**Return Type**

Boolean
CONTrol:SIGNal:AIO:PIN:COUNt?

Applicable Models: E5080B

(Read only) Read the total number of pin on the application I/O. In case for the E5080B, always 15. For more information, see application I/O.

Parameters
- None

Examples
- CONT:SIGN:AIO:PIN:COUN?
- control:signal:aio:pin:count?

Query Syntax
- CONTrol:SIGNal:AIO:PIN:COUNt?

Return Type
- Numeric value

Default
- 15

CONTrol:SIGNal:AIO:PIN<pin>:FUNCtion <func>

Applicable Models: E5080B

(Write-Read) Set and Read the function for the specified port in the application I/O interface. The assigned settings are not saved in a state file, but the setting is stored when changing function and recall them at the next firmware start up.

Parameters
- <pin> Pin Number.
- <func> function name

<table>
<thead>
<tr>
<th>Function name</th>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>PULSE_OUT1</td>
<td>1st pulse output</td>
<td>Out</td>
</tr>
<tr>
<td>PULSE_OUT2</td>
<td>2nd pulse output</td>
<td>Out</td>
</tr>
<tr>
<td>PULSE_OUT3</td>
<td>3rd pulse output</td>
<td>Out</td>
</tr>
<tr>
<td>PULSE_OUT4</td>
<td>4th pulse output</td>
<td>Out</td>
</tr>
<tr>
<td>RF_PULSE_MOD_IN</td>
<td>RF pulse modulation</td>
<td>Inp</td>
</tr>
<tr>
<td>PULSE_SYNC_IN</td>
<td>Pulse generator synchronization trigger input</td>
<td>INI</td>
</tr>
<tr>
<td>INPUT</td>
<td>Arbitrary input (CONT:SIGN:APPL:INP:LEV? to query the level on the assigned port)</td>
<td>Inp</td>
</tr>
<tr>
<td>LOW</td>
<td>Set the Low (Output Low level) at the assigned port.</td>
<td>Out</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------</td>
<td>-----</td>
</tr>
<tr>
<td>HIGH</td>
<td>Set the High (Output High level) at the assigned port</td>
<td>Out</td>
</tr>
<tr>
<td>NF_SOURCEx</td>
<td>Noise Figure source switch control for port x</td>
<td>Out</td>
</tr>
<tr>
<td>(x=1,2,...)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NF_RECEIVERx</td>
<td>Noise Figure receiver switch control for port x</td>
<td>Out</td>
</tr>
<tr>
<td>(x=1,2,...)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCV_ON</td>
<td>+12V power output enable</td>
<td>Out</td>
</tr>
<tr>
<td>DCV_OFF</td>
<td>+12V power output disable</td>
<td>Out</td>
</tr>
<tr>
<td>CHANNEL_CTRL</td>
<td>Function is controlled per channel by</td>
<td></td>
</tr>
</tbody>
</table>

*1) For Noise Figure switch control, the signal is asserted no matter which pins are used in index(x) is not specified.

Examples

```
CONT:SIGN:AIO:PIN:FUNC "LOW"
control:signal:aio5:pin:function "NF_SOURCE15"
```

Query Syntax

```
CONTrol:SIGNal:AIO:PIN:FUNC?
```

Return Type

<sting>

Default

See the table in <func>

---


**Applicable Models:** E5080B

*(Read Only)* Read the list of the available function for the specified pin in the application I/O.

Example, "RF_PULSE_MOD_IN" is returned for the pin 8.

**Parameters**

- `<pin>` Pin Number.

**Examples**

```
CONT:SIGN:AIO:PIN3:FUNC:CAT?
control:signal:aio:pin3:function:catalog?
```

**Query Syntax**

```
```

**Return Type**

<sting>, available function list with Comma separated chars

**Default**

See the table in <func> of CONTrol:SIGNal:APPLication:FUNC?

Applicable Models: E5080B

(Read-Write) Set and read the function for the selected pin on the Application I/O connector. It is set on channel basis and reflected only when “CHANNEL_CTRL” is selected in :CONTroll:SIGNal:AIO:PIN<pin>:FUNCtion <func> for the specified pin. The setting is saved in the state file and subject to preset. Function can be selected from Output signals

Parameters

<pin> Pin Number 1-5, 10-13
<ch> Channel number

Examples

:CONT:SIGN:AIO:PIN3:CHAN1:FUNC?
:CONT:SIGN:AIO:PIN3:CHAN1:FUNC "HIGH"

Query Syntax

CONTroll:SIGNal:AIO:PIN:CHAN:FUNC?

Return Type <char>

Default HIGH


Applicable Models: E5080B

(Read Only) Read the catalog of the functions assignable to the pin on the Application I/O connector.

Parameters

<pin> Pin Number 1-5, 10-13

Examples

control:signal:aio:pin5:channel:function:catalog?

Query Syntax


Return Type Comma separated chars

Default NA
Applicable Models: E5080B

(Read only) Read the level of the specified INPUT pin of the application I/O. This command reads the level immediately after its execution. It is not necessary to assign the pin as INPUT by command. When the specified pin number is 6 to 9 or 14, "Specified Application IO port is not input port." error is returned.

Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;pin&gt;</td>
<td>Pin Number. (E5080B: 1 to 5, 10 to 13)</td>
<td>CONT:SIGN:AIO:PIN:INP2:LEV?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>control:signal:aio:pin:input1:level?</td>
</tr>
</tbody>
</table>

Query Syntax

CONTrol:SIGNal:AIO:PIN:INPut:LEVel?

Return Type

Char ("HIGH" or "LOW")

Default

N/A

CONTrol:SIGNal:KDMI:MAIN:COUNt?

Applicable Models: M980xA,P50xxA/B, P93xB

(Read only) Read the number of ports in the main control port of the I/O connector interface. In case for the M980xA/P50xxA, always 6.

For more information, see I/O Connector Interface (M980xA/P50xxA)

Parameters

None

Example

CONT:SIGN:KDMI:MAIN:COUN?

control:signal:kDMI:main:count?

Query Syntax

CONTrol:SIGNal:KDMI:MAIN:COUNt?

Return Type

Numeric value

Default

6.

CONTrol:SIGNal:KDMI:MAIN<port>:CHANnel<ch>:FUNCtion <func>

Applicable Models: M980xA,P50xxA/B, P93xB

(Write-Read) Set and Read the function for the specified port and channel in the main of the I/O adapter interface. To use this, set CONTrol:SIGNal:KDMI:MAIN:FUNCion at CHANNEL_CTRL. The setting is saved in the state file and subject to preset. Function can be selected from Output signal.
Parameters

<port> Port Number. The number of total ports is returned by CONTrol:SIGNal:KDMI:MAIN:COUNt?.

<func> function name

<table>
<thead>
<tr>
<th>Function name</th>
<th>Description</th>
<th>Type</th>
<th>Assignable port number</th>
</tr>
</thead>
<tbody>
<tr>
<td>PULSE_OUT1</td>
<td>1st pulse output</td>
<td>Output</td>
<td>1</td>
</tr>
<tr>
<td>PULSE_OUT2</td>
<td>2nd pulse output</td>
<td>Output</td>
<td>2</td>
</tr>
<tr>
<td>PULSE_OUT3</td>
<td>3rd pulse output</td>
<td>Output</td>
<td>3</td>
</tr>
<tr>
<td>PULSE_OUT4</td>
<td>4th pulse output</td>
<td>Output</td>
<td>4</td>
</tr>
<tr>
<td>LOW</td>
<td>Set the Low (Output Low level) at the assigned port</td>
<td>Output</td>
<td>1, 2, 3 or 4</td>
</tr>
<tr>
<td>HIGH</td>
<td>Set the High (Output High level) at the assigned port</td>
<td>Output</td>
<td>1, 2, 3 or 4</td>
</tr>
<tr>
<td>NF_SOURCEx (x=1,2,…) *1</td>
<td>Noise Figure source switch control for port x</td>
<td>Output</td>
<td>1, 2, 3 or 4</td>
</tr>
<tr>
<td>NF_RECEIVERx (x=1,2,…) *1</td>
<td>Noise Figure receiver switch control for port x</td>
<td>Output</td>
<td>1, 2, 3 or 4</td>
</tr>
<tr>
<td>NF_LOy (y=1,2,…)</td>
<td>Noise Figure LO switch control for module y</td>
<td>Output</td>
<td>1, 2, 3 or 4</td>
</tr>
</tbody>
</table>

*1) For Noise Figure switch control, the signal is asserted no matter which ports are used in the NF channel if the port/module index(x) is not specified.

Examples

```
CONTrol:SIGNal:KDMI:MAIN1:FUNCion "CHANNEL_CTRL"
CONTrol:SIGNal:KDMI:MAIN1:CHAN1:FUNC "LOW"
CONTrol:SIGNal:KDMI:main2:FUNCion "CHANNEL_CTRL"
control:signal:kDMI:main2:channel2:function "NF_LO1"
```

Query Syntax

CONTrol:SIGNal:KDMI:MAIN:CHAN:FUNC?
**Return Type**  <string>

**Default**  HIGH

**CONTrol:SIGNal:KDMI:MAIN<port>:FUNCtion <func>**

**Applicable Models:**  M980xA, P50xxA/B, P93xxB

*(Write-Read)* Set and Read the function for the specified port in the main c of the I/O adapter interface. The assigned settings are preserved for each system configuration even after preset, firmware restart and power on/off. The CONTrol:SIGNal:KDMI:MAIN:CHANnel:FUNCtion allows you to set the function independently for each channel.

For more information, see I/O Connector Interface

**Parameters**

<port>  Port Number. The number of total ports is returned by CONTrol:SIGNal:KDMI:MAIN:CO

<func>  function name

<table>
<thead>
<tr>
<th>Function name</th>
<th>Description</th>
<th>Type</th>
<th>Assignable</th>
</tr>
</thead>
<tbody>
<tr>
<td>PULSE_OUT1</td>
<td>1st pulse output</td>
<td>Output</td>
<td>1</td>
</tr>
<tr>
<td>PULSE_OUT2</td>
<td>2nd pulse output</td>
<td>Output</td>
<td>2</td>
</tr>
<tr>
<td>PULSE_OUT3</td>
<td>3rd pulse output</td>
<td>Output</td>
<td>3</td>
</tr>
<tr>
<td>PULSE_OUT4</td>
<td>4th pulse output</td>
<td>Output</td>
<td>4</td>
</tr>
<tr>
<td>RF_PULSE_MOD_IN</td>
<td>RF pulse modulation</td>
<td>Input</td>
<td>5 (Factory default)</td>
</tr>
<tr>
<td>INPUT</td>
<td>Arbitrary input (CONTl:SIGN:KDMI:MAIN:INP:LEV? to query the level on the assigned port)</td>
<td>Input</td>
<td>3 (Factory default)</td>
</tr>
<tr>
<td>LOW</td>
<td>Set the Low (Output Low level) at the assigned port.</td>
<td>Output</td>
<td>1, 2, 1 (Factory default)</td>
</tr>
<tr>
<td>HIGH</td>
<td>Set the High (Output High level) at the assigned port</td>
<td>Output</td>
<td>1 or 4 (Factory default)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHANNEL_CTRL</th>
<th>NF_SOURCE_x (x=1,2,...) *1</th>
<th>NF_RECEIVER_x (x=1,2,...) *1</th>
<th>NF_LO_y (y=1,2,...)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Noise Figure source switch control for port x</td>
<td>Noise Figure receiver switch control for port x</td>
<td>Noise Figure LO switch control for module y</td>
</tr>
<tr>
<td></td>
<td>Output 1, 2, 3 or 4</td>
<td>Output 1, 2, 3 or 4</td>
<td>Output 1, 2, 3 or 4</td>
</tr>
</tbody>
</table>

*1) For Noise Figure switch control, the signal is asserted no matter which ports are used in channel if the port/module index(x) is not specified.

Examples

```plaintext
CONT:SIGN:KDMI:MAIN1:FUNC "LOW"
control:signal:kdmimain2:function "NF_LO1"
```

Query Syntax

`CONTrol:SIGNal:KDMI:MAIN:FUNC?`

Return Type

<sting>

Default

See the table in <func>

---

`CONTrol:SIGNal:KDMI:MAIN<port>:FUNCtion:CATalog?`

**Applicable Models:** M980xA, P50xA/B, P93xB

*(Read Only)* Read the list of the available function for the specified port in the main control of the I/O adapter interface. Example, "RF_PULSE_MOD_IN, INPUT1" is returned for the port 5.

For more information, see I/O Connector Interface

**Parameters**

- `<port>` Port Number. The number of total ports is returned by `CONTrol:SIGNal:KDMI:MAIN:COUNt?`.

**Examples**

```plaintext
CONT:SIGN:KDMI:MAIN:FUNC:CAT?
control:signal:kdmimain:func:catalog?
```

**Query Syntax**


**Return Type**

<sting>, available function list with Comma separated chars

**Default**

See the table in <func> of `CONTrol:SIGNal:KDMI:MAIN:FUNC?`

**Applicable Models:** M980xA, P50xxA

*(Read only)* Read the level of the specified INPUT port of the I/O adapter main side. This command reads the level immediately after its execution. It is not necessary to assign the port as INPUT by CONTrol:SIGNal:KDMI:MAIN:FUNCtion command. When the specified `<num>` is not 3 to 6, "Requested input is not assigned to any KDMI ports" error is returned.

For more information, see I/O Connector Interface

**Parameters**

- `<num>` Input Number (M980xA, P50xxA: 3 to 6)

<table>
<thead>
<tr>
<th>Examples</th>
<th>CONT:SIGN:KDMI:MAIN:INP2:LEV?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>control:signal:kdi:main:input1:level?</td>
</tr>
</tbody>
</table>

**Query Syntax** CONTrol:SIGNal:KDMI:MAIN:INPut:LEVEL?

**Return Type** Char ("HIGH" or "LOW")

**Default** N/A

CONTrol:SIGNal:KDMI:SUB:COUNt?

**Applicable Models:** M980xA,P50xxA/B, P93xxB

*(Read only)* Read the number of ports in the sub control of the I/O adapter interface. In case for the M980xA/P50xxA, always 6.

For more information, see I/O Connector Interface

**Parameters** None

<table>
<thead>
<tr>
<th>Examples</th>
<th>CONT:SIGN:KDMI:SUB:COUN?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>control:signal:kdi:sub:count?</td>
</tr>
</tbody>
</table>

**Query Syntax** CONTrol:SIGNal:KDMI:SUB:COUNt?

**Return Type** Numeric value

**Default** 6


**Applicable Models:** M980xA,P50xxA/B, P93xxB

*(Write-Read)* Set and Read the function for the specified port and specified channel in the sub control.
port of the I/O adapter interface. To use this, set CONTrol:SIGNal:KDMI:SUB:FUNCion at CHANNEL_CTRL. The setting is saved in the state file and subject to preset. Function can be selected from Output signal.

**Parameters**

*<port>* Port Number. The number of total ports is returned by CONTrol:SIGNal:KDMI:SUB:COUNt?.

*<ch>* Channel Number.

*<func>* function name

<table>
<thead>
<tr>
<th>Function name</th>
<th>Description</th>
<th>Type</th>
<th>Assignable pot number</th>
</tr>
</thead>
<tbody>
<tr>
<td>READY_FOR_TRIGGER</td>
<td>Ready for trigger</td>
<td>Output</td>
<td>3</td>
</tr>
<tr>
<td>TRIGGER_OUT</td>
<td>AUX trigger</td>
<td>Output</td>
<td>4</td>
</tr>
<tr>
<td>INDEX</td>
<td>index</td>
<td>Output</td>
<td>5</td>
</tr>
<tr>
<td>SWEEP_END</td>
<td>Sweep end</td>
<td>Output</td>
<td>6</td>
</tr>
<tr>
<td>LOW</td>
<td>Set the Low at the assigned port</td>
<td>Output</td>
<td>3, 4, 5 or 6</td>
</tr>
<tr>
<td>HIGH</td>
<td>Set the High at the assigned port</td>
<td>Output</td>
<td>3, 4, 5 or 6</td>
</tr>
<tr>
<td>NF_RECEIVERx (x=1,2,...) *1</td>
<td>Noise Figure receiver switch control for port x</td>
<td>Output</td>
<td>3, 4, 5 or 6</td>
</tr>
<tr>
<td>NF_LOy (y=1,2,...)</td>
<td>Noise Figure LO switch control for module y</td>
<td>Output</td>
<td>3, 4, 5 or 6</td>
</tr>
</tbody>
</table>

*1) For Noise Figure switch control, the signal is asserted no matter which ports are used in the NF channel if the port/module index(x) is not specified.

**Examples**

- CONT:SIGN:KDMI:SUB3:FUNC "CHANNEL_CTRL"
- CONTrol:SIGNal:KDMI:SUB2:FUNCion "CHANNEL_CTRL"
**Query** CONTrol:SIGNal:KDMI:SUB:FUNCtion?

**Return Type** <string>

**Default** HIGH

### CONTrol:SIGNal:KDMI:SUB<port>:FUNCtion <func>

**Applicable Models:** M980xA, P50xxA/B, P93xxB

**(Write-Read)** Set and Read the function for the specified port in the sub control port of the I/O adapter interface. The assigned settings are preserved for each system configuration even after preset, firmware restart and on/off. The CONTrol:SIGNal:KDMI:SUB:CHANnel:FUNCtion allows you to set the function independently for each channel.

For more information, see I/O Connector Interface

**Parameters**

- `<port>` Port Number. The number of total ports is returned by CONTrol:SIGNal:KDMI:SUB:COU
- `<func>` function name

<table>
<thead>
<tr>
<th>Function name</th>
<th>Description</th>
<th>Type</th>
<th>AsphoneNumber</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRIGGER_IN</td>
<td>External trigger</td>
<td>Input</td>
<td>1</td>
</tr>
<tr>
<td>PULSE_SYNC_IN</td>
<td>Pulse synchronization</td>
<td>Input</td>
<td>2</td>
</tr>
<tr>
<td>READY_FOR_TRIGGER</td>
<td>Ready for trigger</td>
<td>Output</td>
<td>3</td>
</tr>
<tr>
<td>TRIGGER_OUT</td>
<td>AUX trigger</td>
<td>Output</td>
<td>4</td>
</tr>
<tr>
<td>INDEX</td>
<td>index</td>
<td>Output</td>
<td>5</td>
</tr>
<tr>
<td>SWEEP_END</td>
<td>Sweep end</td>
<td>Output</td>
<td>6</td>
</tr>
<tr>
<td>INPUT</td>
<td>Arbitrary input</td>
<td>Input</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>(CONTl:SIGN:KDMI:SUB:INP:LEV? to query the level on the assigned port)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOW</td>
<td>Set the Low at the assigned port</td>
<td>Output</td>
<td>3</td>
</tr>
<tr>
<td><strong>HIGH</strong></td>
<td>Set the High at the assigned port</td>
<td><strong>Output</strong> 3, (Factory default) 6</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td><strong>CHANNEL_CTRL</strong></td>
<td>Function is controlled per channel. Define the function by :CONTrol:SIGNal:KDMI:SUB:CHANnel:FUNCtion</td>
<td><strong>N/A</strong> 3,</td>
<td></td>
</tr>
<tr>
<td><strong>NF_RECEIVERx</strong> (x=1,2,....) *1</td>
<td>Noise Figure receiver switch control for port x</td>
<td><strong>Output</strong> 3,</td>
<td></td>
</tr>
<tr>
<td><strong>NF_LOy</strong> (y=1,2,...)</td>
<td>Noise Figure LO switch control for module y</td>
<td><strong>Output</strong> 3,</td>
<td></td>
</tr>
</tbody>
</table>

*1) For Noise Figure switch control, the signal is asserted no matter which ports are used in channel if the port/module index(x) is not specified.

**Examples**

```
CONT:SIGN:KDMI:SUB3:FUNC "LOW"
control:signal:kmi:sub2:func "INPUT4"
```

**Query Syntax**

```
```

**Return Type**

<sting>, available function list with Comma separated chars

**Controllable Models**: M980xA, P50xxA/B, P93xxB

*(Read Only)* Read the list of the available function for the specified port in the sub control port of the I/O adapter interface. Example, "TRIGGER_IN, INPUT3" is returned for the port 1.

For more information, see I/O Connector Interface

**Parameters**

- `<port>` Port Number. The number of total ports is returned by CONTrol:SIGNal:KDMI:SUB:COUNt?.

**Examples**

```
CONT:SIGN:KDMI:SUB:FUNC:CAT?
control:signal:kdm:sub:function:catalog?
```

**Query Syntax**

```
```

**Return Type**

<sting>, available function list with Comma separated chars
**Default**  See the table in <func> of  CONTrol:SIGNal:KDMI:SUB:FUNC?

CONTrol:SIGNal:KDMI:SUB:INPut<num>:LEVel?

**Applicable Models:** M980xA, P50xxA

*(Read only)* Read the level of the specified INPUT port of the I/O adapter sub side. This command reads the level immediately after its execution. It is not necessary to assign the port as INPUT by CONTrol:SIGNal:KDMI:SUB:FUNCtion command. The When the specified input number is not 1 to 4, "Requested input is not assigned to any KDMI port" error is returned.

For more information, see I/O Connector Interface (M980xA/P50xxA)

**Parameters**

- `<num>`  Input Number (M980xA, P50xxA: 1 to 4)

**Examples**

- `CONT:SIGN:KDMI:SUB:INP4:LEV?`
- `control:signal:kmi:sub:input5:level?`

**Query Syntax**  CONTrol:SIGNal:KDMI:MAIN:INPut:LEVel?

**Return Type**  Char ("HIGH" or "LOW")

**Default**  N/A

CONTrol:SIGNal:PXI:RTRigger[:STATe] <bool>

**Applicable Models:** M980xA

*(Write-Read)* Turns ON / OFF Ready for Trigger output from Backplane Trigger Lines.

**Parameters**

- `<bool>`  Choose from:
  - **ON** (or 1) - VNA DOES output ready for trigger signal from Backplane Trigger line
  - **OFF** (or 0) – VNA does NOT output ready for trigger signal from Backplane Trigger line.

**Examples**

- `CONT:SIGN:PXI:RTR 1`
- `control:signal:pxi:rtrigger:stat off`

**Query Syntax**  CONTrol:SIGNal:PXI:RTRigger[:STATe]?

**Return Type**  Boolean

3858
CONTrol:SIGNal:PXI:RTTrigger:ROUTe <char>

Applicable Models: M980xA

(Write-Read) Specifies the Backplane Trigger Line to use for the trigger OUT ready line.

Parameters

<char> Choose from:

- **TRIG0** – Backplane Trigger Lines (PXI TRIG0)
- **TRIG1** – Backplane Trigger Lines (PXI TRIG1)
- **TRIG2** – Backplane Trigger Lines (PXI TRIG2)
- **TRIG3** – Backplane Trigger Lines (PXI TRIG3)
- **TRIG4** – Backplane Trigger Lines (PXI TRIG4)
- **TRIG5** – Backplane Trigger Lines (PXI TRIG5)
- **TRIG6** – Backplane Trigger Lines (PXI TRIG6)
- **TRIG7** – Backplane Trigger Lines (PXI TRIG7)

Examples

CONT:SIGN:PXI:RTR:ROUT TRIG0
control:signal:pxi:rtrigger:route trig0

Query Syntax

CONTrol:SIGNal:PXI:RTTrigger:ROUTe?

Return Type

Character

Default

TRIG1

**Applicable Models:** M980xA

(*Write-Read*) Turns ON / OFF the trigger output from PXI Backplane line when “CONTrol:SIGNal:TRIGger:OUTP[:STATe]” is “ON”.

**Parameters**

<bool> **ON** (or 1) - VNA DOES output trigger signals from PXI Backplane when “CONTrol:SIGNal:TRIGger:OUTP[:STATe]” is “ON”.

**OFF** (or 0) – VNA does NOT output trigger signals from PXI Backplane.

**Examples**

```
CONT:SIGN:PXI:TRIG:OUTP 1
control:signal:pxi:trigger:output:stat off
```

**Query Syntax**

CONTrol:SIGNal:PXI:TRIGger:OUTPut[:STATe]?

**Return Type** Boolean

**Default** OFF

**CONTrol:SIGNal:PXI:TRIGger:OUTPut:ROUTe <char>**

**Applicable Models:** M980xA

(*Write-Read*) Specifies the Backplane Trigger Line to use for the trigger output line.

**Parameters**

<char> Choose from:

- **TRIG0** – Backplane Trigger Lines (PXI TRIG0)
- **TRIG1** – Backplane Trigger Lines (PXI TRIG1)
- **TRIG2** – Backplane Trigger Lines (PXI TRIG2)
- **TRIG3** – Backplane Trigger Lines (PXI TRIG3)
- **TRIG4** – Backplane Trigger Lines (PXI TRIG4)
- **TRIG5** – Backplane Trigger Lines (PXI TRIG5)
- **TRIG6** – Backplane Trigger Lines (PXI TRIG6)
- **TRIG7** – Backplane Trigger Lines (PXI TRIG7)

**Examples**

```
CONT:SIGN:PXI:TRIG:OUTP:ROUT TRIG0
control:signal:pxi:trigger:output:route trig0
```
**Query Syntax**  

**Return Type**  
Character

**Default**  
TRIG2

CONTrol:SIGNal:STReamline:RTRigger[:STATe] <bool>

**Applicable Models:** P50xxA/B, P93xxB

*(Write-Read)* Turns ON / OFF Ready for Trigger output from Streamline Rear SMB.

**Parameters**

<bool> Choose from:

- **ON** (or 1) - VNA DOES output ready for trigger signal from Streamline Rear SMB
- **OFF** (or 0) – VNA does NOT output ready for trigger signal from Streamline Rear SMB

**Examples**

```
CONT:SIGN:STR:RTR 1
control:signal:streamline:rtrigger:stat off
```

**Query Syntax**  
CONTrol:SIGNal:STReamline:RTRigger[:STATe]?

**Return Type**  
Boolean

**Default**  
0 (Preset will not reset the setting.)

CONTrol:SIGNal:STReamline:RTRigger:ROUTe <char>

**Applicable Models:** P50xxA/B, P93xxB

*(Write-Read)* Specifies the Backplane Trigger Line to use for the trigger OUT ready line.

**Parameters**

<char> Choose from:

- **NONE** – Streamline Rear SMB Not connected (NONE)
- **REAR1** – Streamline Rear SMB (TRIG1)
- **REAR2** – Streamline Rear SMB (TRIG2)

**Examples**

```
CONT:SIGN:STR:RTR:ROUT TRIG1
control:signal:streamline:rtrigger:route trig1
```

**Query Syntax**  
CONTrol:SIGNal:STReamline:RTRigger:ROUTe?
**Return Type**  Character  
**Default**  N/A

**CONTrol:SIGNal:STReamline:TRIGger:OUTPut[:STATe] <bool>**

**Applicable Models:**  P50xxA/B, P93xxB  

*(Write-Read)* Turns ON / OFF the trigger output from Streamline Rear SMB when “CONTrol:SIGNal:TRIGger:OUTP[:STATe]” is “ON”.

**Parameters**

<bool>  ON (or 1) - VNA DOES output trigger signals from Streamline Rear SMB when “CONTrol:SIGNal:TRIGger:OUTP[:STATe]” is “ON”.  

OFF (or 0) – VNA does NOT output trigger signals from Streamline Rear SMB.

**Examples**

```
CONT:SIGN:STR:TRIG:OUTP 1
control:signal:streamline:trigger:output:stat off
```

**Query Syntax**  CONTrol:SIGNal:STReamline:TRIGger:OUTPut[:STATe]?

**Return Type**  Boolean  
**Default**  OFF

**CONTrol:SIGNal:STReamline:TRIGger:OUTPut:ROUTe <char>**

**Applicable Models:**  P50xxA/B, P93xxB  

*(Write-Read)* Specifies the Streamline Rear SMB to use for the trigger output line.

**Parameters**

<char>  Choose from:

**NONE** – Streamline Rear SMB Not connected (NONE)  

**REAR1** – Streamline Rear SMB (REAR TRIG1)  

**REAR2** – Streamline Rear SMB (REAR TRIG2)

**Examples**

```
CONT:SIGN:STR:TRIG:OUTP:ROUT TRIG0
control:signal:streamline:trigger:output:route trig0
```

**Query Syntax**  CONTrol:SIGNal:STReamline:TRIGger:OUTPut:ROUTe?

**Return Type**  Character
Default TRIG2
## Control:Multiplexer Commands

Controls the E5092A Configurable Multiport Test Set.

### CONTroMULTiplexer:

#### OUTPut

| A|B|C|D|DATA |
| A|B|C|D:VOLTage[DATA] |

#### PORT

| [:SELect] |

Click on a keyword to view the command details.

### See Also

- Learn about External Test Set Control
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

### CONTroMULTiplexer<id>:OUTPut:<grp>[:DATA] <num>
Applicable Models: N522xB, N523xB, N524xB, E5080B, M980xA

(Read-Write) Sets or returns the output port data for specified group with id of the E5092A multiport test set.

Notes:
This command performs an **immediate** setting of the specified data on the indicated test set, as opposed to the `SENS<e<cn><num>:MULTiplexer<id>:OUTPut:<grp>[[:DATA]] <num>` command which sets the data only at the beginning of each sweep of the specified channel number ‘cn’. To use this command with the M980xA PXI VNA, the firmware version must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required.

**Parameters**

- `<id>`  
  Id of the external test set either 1 or 2. If unspecified, Id is assumed to be 1.

- `<grp>`  
  A | B | C | D

- `<num>`  
  An integer specifying the decimal value of the control line. Values are obtained by adding weights from the following table that correspond to individual lines.

The output port data range is between 0 to 255 (0=All lines are turned OFF and 255 all lines are turned ON).

<table>
<thead>
<tr>
<th>Line</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td>8</td>
<td>128</td>
</tr>
</tbody>
</table>

**Examples**

```
CONT:MULT1:STAT OFF
CONT:MULT1:OUTP:B 8
' immediately output 8 from control line B.
```

**Query Syntax**

`CONTrol:MULTiplexer<id>:OUTPut:<grp>[[:DATa]]?`

**Return Type**  
Numeric

**Default**  
0

`CONTrol:MULTiplexer<id>:OUTPut:<grp>:VOLTage[:DATA] <volt>`
**Applicable Models:** N522xB, N523xB, N524xB, E5080B, M980xA

(Read-Write) Sets or returns the output voltage for specified group with id of the E5092A multiport test set.

**Notes:**
This command performs an immediate setting of the specified voltage on the indicated test set, as opposed to the `SENSe<cnum>:MULTiplexer<id>:OUTPut:<grp>:VOLTage[:DATA] <volt>` command which sets the voltage only at the beginning of each sweep of the specified channel number `cnum`. To use this command with the M980xA PXI VNA, the firmware version must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required.

**Parameters**
- `<id>`  Id of the external test set either 1 or 2. If unspecified, Id is assumed to be 1.
- `<grp>`  A | B | C | D
- `<volt>`  Output voltage range for `<grp>` is between 0 to 5.2V and resolution is 10mV.

**Examples**

```
CONT:MULT1:STAT OFF
CONT:MULT1:OUTP:B:VOLT 4.2
' immediately set control line B voltage to 4.2 volt.
```

**Query Syntax**  `CONTro:MULTiplexer<id>:OUTPut:<grp>:VOLtage[:DATa]`?

**Return Type**  Numeric  
**Default**  0 V

```
CONTrol:MULTiplexer<id>:PORT<pnum>[[:SELect] <string>

Applicable Models: N522xB, N523xB, N524xB, E5080B, M980xA

(Write-Only) Sets the multiport test set port. If this command creates a conflict with an existing port, the VNA will resolve the conflict.

**Notes:**
This command performs an immediate setting of the specified port on the indicated test set, as opposed to the `SENSe<cnum>:MULTiplexer<id>:PORT<pnum>:SELect <string>` command which sets the port only at the beginning of each sweep of the specified channel number `cnum`. To use this command with the M980xA PXI VNA, the firmware version must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required.

**Parameters**
- `<id>`  Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the `SENSe:MULT:TYPE` command.
<pnum> The VNA connection port number (1 to 4).

If the E5092A’s configuration is E5092_28, this parameter specifies the individual switch number (1 to 4: SP4T switches, 5 to 10: SPDT switches).

<string> The label of the E5092A’s measurement port.

If the E5092A’s configuration is E5092_28, this parameter specifies the connection of the individual switch. Refer to the table of SENS:MULT:PORT:SEL.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONT:MULT1:STAT OFF</td>
<td></td>
</tr>
<tr>
<td>CONT:MULT1:TYP 'E5092_22'</td>
<td></td>
</tr>
<tr>
<td>CONT:MULT1:PORT1:SEL 'A2'</td>
<td>' set E5092A to 22-port config and immediately connect port1 to 'A2'.</td>
</tr>
</tbody>
</table>

**Return Type** String

**Default** Not Applicable
CSET Commands

Manages several aspects of Cal Sets.

<table>
<thead>
<tr>
<th>CSET:</th>
</tr>
</thead>
<tbody>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>COPY</td>
</tr>
<tr>
<td>DALL</td>
</tr>
<tr>
<td>DATE?</td>
</tr>
<tr>
<td>DELeete</td>
</tr>
<tr>
<td>EXISts?</td>
</tr>
<tr>
<td>ETERm:</td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>[:DATA]</td>
</tr>
<tr>
<td>FORMat?</td>
</tr>
<tr>
<td>X:VALues</td>
</tr>
<tr>
<td>FIXTure:</td>
</tr>
<tr>
<td>CASCade</td>
</tr>
<tr>
<td>CHARacterize</td>
</tr>
<tr>
<td>DEEMbed</td>
</tr>
<tr>
<td>EMBed</td>
</tr>
<tr>
<td>ENR:EMBed</td>
</tr>
<tr>
<td>ZERO</td>
</tr>
<tr>
<td>ITEM:</td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>DATA?</td>
</tr>
</tbody>
</table>
CSET:CATalog?

**Applicable Models:** All

This command replaces SENS:CORR:CSET:CAT?

*(Read-only)* Returns the names of Cal Sets stored on the VNA.

**Parameters** None

**Examples**

CSET:CAT?

**Returns:**

"CalSet_0913,CalSet_1,CalSet_2,CalSet_3,CalSet_4,CH1_CALREG,CH31_CALREG,
CH1_CALREG,CH31_CALREG,MyCalAll_SMC_002,MyCalAll_STD_001"

**Return Type** Comma-separated string of names

**Default** Not Applicable

CSET:COPY <string>,<string>
Applicable Models: All

(Write-only) Creates a new Cal Set and copies the current Cal Set data into it. Use this command to manipulate data on a Cal Set without corrupting the original cal data.

Parameters

<string>,<string> The first string is the name of the current Cal Set. The second string is the name of the new Cal Set copy.

Examples

CSET: COPY 'My2Port','My2PortCopy'

Query Syntax Not Applicable

Default Not Applicable

CSET:DALL

Applicable Models: All

(Write-only) Deletes ALL Cal Sets from the VNA, including phase reference and Global Delta Match Cal Sets.

Parameters None

Examples CSET: DALL

Query Syntax Not Applicable

Default Not Applicable

CSET:DATE? <string>

Applicable Models: All

(Read-only) Returns the (year, month, day) that the specified Cal Set was last saved.

See Also

MMEM:DATE?

MMEM:TIME?

CSET:TIME?

Parameters

<string> Cal Set name.
Examples

CSET:DATE? "CalSet_11"

Returns:

+2013,+5,+1

Return Type: Comma-separated integers.

Default: Not Applicable

---

CSET:DELETE <string>

Applicable Models: All

This command replaces SENS:CORR:CSET:DELETE

(Write-only) Deletes the specified Cal Set from the VNA.

- If the Cal Set is currently being used by a channel, the Cal Set is deleted and correction for the channel is turned off.
- If the Cal Set is not found, no error is returned.

Parameters

<string> Name of the Cal Set to delete. Not case-sensitive.

Examples

CSET:DEL "MyCalSet"

Query Syntax: Not Applicable

Default: Not Applicable

---

CSET:EXISTS? <string>
Applicable Models: All

(Read-only) Returns whether or not the specified Cal Set exists on the VNA.

Parameters

<string> Name or GUID of the Cal Set enclosed in quotes.

The GUID must also be enclosed in curly brackets.

Examples

dim check
check = CSET:EXISts? "MyCalSet"
check = CSET:EXISts? "{7C4EEA5E-40D2-4D70-A048-33BFFE704163}"

Return Type Boolean

ON or 1 - Cal Set exists.
OFF or 0 - Cal Set does NOT exist.

Default Not Applicable

CSET:ETERm:CATalog? <CSET Name>[,<errorTermFilter>]

Applicable Models: All

(Read-only) Returns a list of error term names for the given Cal Set.

Parameters

<CSET Name> (String) Name of Cal Set to query.
<errorTermFilter> (Optional argument) CSET:ETER:CAT? <CSETName>, "<errorTermFilter>" will return only the error term names with the filter string in them. For example, if it is a full 2-port cal, then CSET:ETER:CAT? <CSETName>, “cross” would return all “Crosstalk(n,n)” error terms. (Note that the filter is not case sensitive.)

Entering CSET:ETER:CAT? <CSETName> "" or CSET:ETER:CAT? <CSETName> will return all error terms for the given Cal Set.

Examples

CSET:ETER:CAT? "CalSet_1"
CSET:ETER:CAT? "CalSet_1", "trans"

Return Type Variant

Default Not Applicable
CSET:ETERm[:DATA] <CSET Name>,<ETerm Name>,<data>

**Applicable Models:** All

*(Read-Write)* Sets and returns the error term data (real, imaginary pairs) for the given Cal Set and error term name.

**Parameters**
- `<CSET Name>` *(String)* Name of Cal Set to manipulate.
- `<ETerm Name>` *(String)* Name used to identify an error term in the Cal Set.
- `<data>` *(Block)* Error term data - a real/imaginary data pair for each data point.

**Examples**
```
CSET:ETER "CalSet_1","Directivity(1,1)", 0.237,-1.422, 0.513, 0.895
CSET:ETER? "CalSet_1","Directivity(1,1)"  'read
```

**Query Syntax**
CSET:ETERm:DATA? <CSET Name>,<ETerm Name>

**Return Type**
Block data

**Default**
Not Applicable

---

CSET:ETERm:FORMat? <term name>,<rcv port>,<src port>[,<rcvPort gainstate>][,<srcPort gainstate>]

**Applicable Models:** All

*(Read-only)* Returns switched gain error terms. Switched gain error terms have a suffix on the term name to identify the receiver gain states that apply to the term. For example:

- TransmissionTracking(2,1)
- TransmissionTracking(2,1)[0,1]
- TransmissionTracking(2,1)[1,0]
- TransmissionTracking(2,1)[1,1]

**Parameters**
- `<term name>` *(String)* Name used to identify an error term.
- `<rcv port>` Receiver port number.
- `<src port>` Source port number.
- `<rcvPort gainstate>` Receiver port gain state.
- `<srcPort gainstate>` Source port gain state.
CSET:ETERm:X:VALues <CSET Name>,<ETerm Name>,<freqlist>

Applicable Models: All

(Read-Write) Sets and returns the x-axis frequencies for the given Cal Set and error term name. This command requires that the error term already be in existence either from a calibration session or having been created with CSET:ETER:DATA.

This command requires that the frequency array length match the existing size of the error term. For example, if the error term is 3 buckets long (3 complex numbers), then the frequency list must be 3 values long.

Parameters

- `<CSET Name>` (String) Name of Cal Set to manipulate.
- `<ETerm Name>` (String) Name used to identify an error term in the Cal Set.
- `<freqlist>` (Block) X-axis frequencies associated with the error term.

Examples

'CQuery error term data from calset named "Calset1"
CSET:ETER:DATA? "Calset7","Directivity(1,1)"

'If needed, change or upload the data for the "Directivity" error term (example of a three point eterm)

'Query what the frequency values are for the error term
CSET:ETER:X:VAL? "Calset7","Directivity(1,1)"

'If needed, change the frequency values for the error term
CSET:ETER:X:VAL "Calset7","Directivity(1,1)" 1.00000000000E+007,1.42450000000E+008,2.74900000000E+008

Query Syntax

CSET:ETERm:X:VALues? <CSET Name>,<ETerm Name>

Return Type

Block data
CSET:FIXTure:CASCade <s2p1>,<s2p2>,<s2pResult>,<char>

**Applicable Models:** All

*(Write-only)* Combines the losses and phase shift of two S2P files into a single S2P file. Learn more.

**Parameters**

- `<s2p1>` *(String)* Path and filename of one of the S2P files to be combined.
- `<s2p2>` *(String)* Path and filename of the other S2P file to be combined.
- `<s2pResult>` *(String)* Path and filename of the combined S2P file.
- `<char>` *(Character)* Format. Choose from:
  - REIM - Real, imaginary data pairs
  - LOG - Log magnitude, phase
  - LINear - Linear magnitude, phase

**Examples**

```
CSET:FIXT:CASC "D:\a.s2p","D:\b.s2p","D:\c.s2p",LOG
```

**Query Syntax** Not Applicable

**Default** Not Applicable

---

CSET:FIXTure:CHARacterize <cs1>,<cs2>,<port>,<s2p>,<char>[,<pivot>]

**Applicable Models:** All

*(Write-only)* Characterizes a fixture based on two Cal Sets. The stimulus settings of the two Cal Sets do NOT have to be identical, but they MUST have a common frequency range for interpolation. A new S2P file is created. Learn more about Cal Plane Manager.

**Parameters**

- `<cs1>` *(String)* Name of an existing Cal Set 1 which describes the cal closest to the VNA. The Cal Set must reside on the VNA.
- `<cs2>` *(String)* Name of an existing Cal Set 2 which describes the cal closest to the DUT. The Cal Set must reside on the VNA.
- `<port>` *(Numeric)* Port number described in the Cal Sets.
- `<s2p>` *(String)* Name of the S2P file containing the adapter/fixture characterization.
(Character) Format. Choose from:

- REIM - Real, imaginary data pairs
- LOG - Log magnitude, phase
- LINear - Linear magnitude, phase


Examples

CSET:FIXT:CHAR "CalSet1","CalSet2",1,"Fixture.s2p"

cset:fixture:characterize "CalSet1","CalSet2",2,"Fixture.s2p",90

Query Syntax

Not Applicable

Default

Not Applicable

CSET:FIXTure:DEEMbed <cs1>,<cs2>,<s2p>,<port>, <compPwr>[,extrap]

Applicable Models: All

(Write-only) De-embeds a fixture from an existing Cal Set based on an S2P file. A new Cal Set is created with the effects of the fixture removed.

When the new Cal Set is applied to a channel, the effects of fixturing are removed from the measurement data. Do NOT enable fixturing. The effects of the fixture are removed when the new Cal Set is selected and correction is turned ON.

Parameters

<cs1> (String) Name of an existing Cal Set which resides on the VNA.

<cs2> (String) Name of new Cal Set which contains updated error terms with fixture de-embedded.

<s2p> (String) Name of the S2P file which characterizes the adapter/fixture.

<port> (Numeric) Port number from which fixture will be de-embedded.

<compPwr> (Boolean)

ON (1) - When the Cal Set contains a power correction array for the fixture port, that array will be compensated for the fixture loss.

Warning: enabling power compensation can result in an increase in test port power and consequently, increased power to the DUT. Use with caution.

OFF (0) - Do not compensate for loss in source power through the fixture.

[extrap] (Boolean) Optional argument.
ON (1) - Applies a simple extrapolation when the S2P file has a narrower frequency range than the Cal Set. The values for the first and last data points are extended in either direction to cover the frequency range of the Cal Set.

OFF (0) - Extrapolation is NOT performed (default setting).

Examples

CSET:FIXT:DEEM "MyCalSet","MyNewCalSet","Fixture.s2p",1,1,1

cset:fixture:deembed
"MyCalSet","MyNewCalSet","Fixture.s2p",1,1,1 'extrapolation is performed if the s2p frequency range is narrower than that of the Cal Set.

Query Syntax Not Applicable

Default Not Applicable

CSET:FIXTURE:EMBED <cs1>,<cs2>,<s2p>,<port>,<compPwr>[,extrap]

Applicable Models: All

(Write-only) Embeds a fixture (usually a matching network) into an existing Cal Set based on an S2P file. A new Cal Set is created with the effects of the matching network included in the correction data.

When the new Cal Set is applied to a channel, the effects of the fixture are included in the measurement data. Do NOT enable fixturing. The effects of the matching network are included when the new Cal Set is selected and correction is turned ON.

Parameters

<cs1> (String) Name of an existing Cal Set which resides on the VNA.
<cs2> (String) Name of new Cal Set which contains updated error terms with fixture embedded.
<s2p> (String) Name of the S2P file which characterizes the fixture / matching network.
<port> (Numeric) Port number to which fixture will be added.
<compPwr> (Boolean)

ON (1) - Increase the source power to compensate for the loss through the fixture. The result is that the specified power level will be correct at the DUT input.

Warning: enabling power compensation can result in an increase in test port power and consequently, increased power to the DUT. Use with caution.
OFF (0) - Do not compensate for loss in source power through the matching network.

[extrap] (Boolean) Optional argument.

ON (1) - Applies a simple extrapolation when the S2P file has a narrower frequency range than the Cal Set. The values for the first and last data points are extended in either direction to cover the frequency range of the Cal Set.

OFF (0) - Extrapolation is NOT performed (default setting).

Examples

CSET:FIXT:EMB "MyCalSet","MyNewCalSet","Fixture.s2p",1,1

cset:fixture:embed "MyCalSet","MyNewCalSet","Fixture.s2p",1,1,1

'extrapolation is performed if the s2p frequency range is narrower than that of the Cal Set.'

Query Syntax
Not Applicable

Default
Not Applicable

CSET:FIXTure:ENR:EMBed <inEnr>,<s2p>,<outEnr>

Applicable Models: All

(Write-only) Generate a new ENR file by embedding an adapter to an existing ENR file.

Parameters

<inEnr> (String) Path and filename of original ENR file.

<s2p> (String) Path and filename of the S2P file which characterizes the adapter/fixture network.

<outEnr> (String) Path and filename of new ENR file to output

Examples

CSET:FIXT:EMB "D:\Original.enr","D:\adapter.s2p","D:\new.enr"

Query Syntax
Not Applicable

Default
Not Applicable

CSET:FIXTure:ZERO<original_s4p>,<result_s4p>,<sparameters>,<complexformat>
Applicable Models: All

(Write-only) Creates a new S4P file.

Parameters

- `<original_s4p>` (String) Path and filename of the original S4P file.
- `<result_s4p>` (String) Path and filename of the new S4P file.
- `<sparameters>` (String) Comma-separated terms to zero-out.
- `<complexformat>` (Character) Format. Choose from:
  - REIM - Real, imaginary data pairs
  - LOG - Log magnitude, phase
  - LIN - Linear magnitude, phase

Examples

```
CSET:FIXT:ZERO
"D:\originalFile.s4p","D:\newFile.s4p","S11,S21,S33",LOG
```

Query Syntax

Not Applicable

Default

Not Applicable

---

CSET:ITEM:CATalog? <calset>

Applicable Models: All

(Read-only) The Cal Set mainly contains error term data for measurement correction purposes. But Cal Sets can also contain auxiliary information used to describe how the Cal Set was constructed. This information is stored as name value pairs and can be accessed by the item name. The catalog query returns a list of item names contained in the specific Cal Set being queried.

Parameters

- `<calset>` (String) Name of the Cal Set item.

Examples

```
CSET:ITEM:CAT? "mycalset"
```

Return Type

String

Default

Not Applicable

---

CSET:ITEM:DATA? <calset>,<itemName>
Applicable Models: All

(Read-Write) Returns the VNA Measurement Class or Channel that created the specific Cal Set item.

About Cal Set Items

A Cal Set item is a named value. You can list the named values using CSET:ITEM:CATalog? or SENS:CORR:CSET:ITEM:CATalog?

You can query the value of a specific item by asking for its data: CSET:ITEM:DATA?

For example, one of the items added by the VNA firmware to every Cal Set is named 'Created By'. The value attached to this item is the name of the VNA Measurement Class or Channel that created the Cal Set. When an SMC cal is performed, you can query the Cal Set for the 'Create By' item, and it will return 'Scalar Mixer/Converter'. The same query on an NFx channel returns 'Noise Figure Converters'.

CSET:ITEM:DATA? "mycalset","Created By"

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<calset>` (String) Name of the Cal Set item.
- `<itemName>` (String) VNA Measurement Class or Channel that created the Cal Set.

Examples

```
SENS:CORR:CSET:ITEM? "mycalset","Created By"
```

Return Type

String

Default

Not Applicable

CSET:TIME? <string>

Applicable Models: All

(Read-only) Returns the (hour, minute, second) that the specified Cal Set was last saved. The time is returned in local time as setup in the VNA operating system.

See Also

CSET:DATE?

MMEM:DATE?

MMEM:TIME?

Parameters

- `<string>` Cal Set name.
Examples

CSET:TIME? "CalSet_11"

'Returns:
+13,+6,+1

Return Type
Comma-separated integers.

Default
Not Applicable

CSET:VALIDATE? <calset>,<channel>

Applicable Models: All

(Read-only) This query compares the stimulus of "chanNumber" (<channel>) with the stimulus stored in the specified calset.

Parameters

<calset>  (String) Name of the Cal Set item.
<channel> (Integer) Any existing channel number.

Examples

CSET:VALIDATE? "CalSet_11",2

Return Type
Enumeration:

EQUal - Calset stimulus matches the channel stimulus.

DELTa - Calset can be used, but differs in some property that is not related to the xAxis.

XAXis - Calset can be used but will be "interpolated": interpolated data maybe computed or just decimated based on the alignment.

REStore - Calset could be used but requires restoring the calibration stimulus stored in the calset.

REJect - Calset is rejected (calset not recognized or was created by an incompatible channel).

Default
Not Applicable
## Display Commands

Controls the settings of the front panel screen.

**Display:**
- **ANnotation**
  - **FREQuency[:STATE]**
  - **MESSage:STATe**
  - **[:STATus]**
- **ARRange**
- **CATalog?**
- **COLOr More**
- **Commands**
- **ENABLE**
- **FSIGn**
- **GUI**
  - **POWer**
  - **SPIN**
  - **RESolution**
- **MEASure**
  - **DELeTe**
  - **FEED**
  - **MEMory**
    - **[:STATe]**
  - **MOVE**
  - **SELect**
    - **[:STATe]**
  - **TITLe**
    - **DATA**
    - **[:STATe]**
    - **Y[:SCALe]**
    - **AUTO**
    - **PDIVision**
    - **RLEVel**
    - **RPOsition**
- **SHEET**
  - **ARRange**
  - **CATalog**
  - **STATe**
  - **TITLe:DATA**
- **SPLit**
Click on a keyword to view the command details.

Blue keywords are superseded.

See Also

- Referring to Traces Channels Windows and Meas Using SCPI
- See an example using some of these commands
- Synchronizing the Analyzer and Controller
- Learn about Screen Setup
- SCPI Command Tree

**DISPlay:ANNotation:FREQuency[:STATe] <ON | OFF>**
Applicable Models: All

(Read-Write) Turns frequency information on the display title bar ON or OFF for all windows.

**Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ON</td>
<td>OFF&gt;</td>
</tr>
</tbody>
</table>

**Examples**

```
DISP:ANN:FREQ ON
display:annotation:frequency:state off
```

**Query Syntax**

DISPlay:ANNotation:FREQuency[:STATe]?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON (1)

---

**DISPlay:ANNotation:MESSage:STATe <ON | OFF>**

Applicable Models: All

(Read-Write) Enables and disables error pop-up messages on the display.

**Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ON</td>
<td>OFF&gt;</td>
</tr>
</tbody>
</table>

**Examples**

```
DISP:ANN:MESS:STAT ON
display:annotation:message:state off
```

**Query Syntax**

DISPlay:ANNotation:MESSage:STATe?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON (1)

---

**DISPlay:ANNotation[:STATus] <ON | OFF>**
Applicable Models: All

(Read-Write) Turns the status bar at the bottom of the screen ON or OFF. The status bar displays information for the active window.

Parameters

<ON | OFF>
  ON (or 1) - turns status bar ON.
  OFF (or 0) - turns status bar OFF.

Examples

  DISP:ANN ON
  display:annotation:status off

Query Syntax

  DISPlay:ANNotation[:STATus]?

Return Type

  Boolean (1 = ON, 0 = OFF)

Default

  Last state that was set

---

DISPlay:ARRange <char>

Applicable Models: All

(Write-only) Places EXISTING measurements into pre-configured window arrangements. Overlay, Stack(2), Split(3), and Quad(4) creates new windows. To learn more, see Window Layout.

Parameters

<char>
  Window arrangement. Choose from:

  - TILE - tiles existing windows
  - CASCade - overlaps existing windows
  - OVERlay - all traces placed in 1 window
  - STACk - 2 windows
  - SPLit - 3 windows
  - QUAD - 4 windows
  - MEASure - 1 measurement per window
  - CHANnel - 1 channel per window
  - LTOR - Arrange existing windows as a single row of side-by-side windows.

Examples

  DISP:ARR CASC
display:arrange cascade

Query Syntax

  Not Applicable
DISPlay:CATalog?

Applicable Models: All

(Read-only) Returns the existing Window numbers.

**Note:** If there are no traces in the window, this query returns the "EMPTY" string.

To read the window number of the selected trace, use `Calc:Par:WNUM`.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>String of Character values, separated by commas</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example</td>
<td>Two windows with numbers 1 and 2 returns: &quot;1,2&quot;</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

DISPlay:ENABle <ON | OFF>

Applicable Models: All

(Read-Write) Specifies whether to disable or enable all analyzer display information in all windows in the analyzer application. Marker data is not updated. More CPU time is spent making measurements instead of updating the display.

**Parameters**

<ON | OFF>  
ON (or 1) - turns the display ON.  
OFF (or 0) - turns the display OFF.

**Examples**

`DISP:ENAB ON`  
`display:enable off`

**Query Syntax**  
`DISP:ENABle?`

**Return Type**  
Boolean (1 = ON, 0 = OFF)

**Default**  
ON

DISPlay:FSIGn <ON | OFF>
Applicable Models: All

(Read-Write) Shows or hides the window which displays global pass/fail results.

Parameters

<ON | OFF>  
ON (or 1) - displays the pass/fail dialog
OFF (or 0) - hides the pass/fail dialog

Examples

DISP:FSIG ON  
display:fsign off

Query Syntax  
DISPlay:FSIGn?

Return Type  
Boolean (1 = ON, 0 = OFF)

Default  
OFF

DISPlay:GUI:POWer:SPIN:RESolution <real>

Applicable Models: All

(Read-Write) Sets and returns the resolution of the front panel knob when it is used to adjust Source Power manually.

Parameters

<real>  
Spin resolution value (Real value). The range of acceptable values is 0.01 to 100.

Examples

DISP:GUI:POW:SPIN:RES 0.01  
'dWrite - Every tick of the front panel knob will change the Power Level by 0.01 dBm.'

display:gui:power:spin:resolution 0.01

Query Syntax  
DISPlay:GUI:POWer:SPIN:RESolution?

Return Type  
Real

Default  
0.1

DISPlay:MEASure<mnum>:DELete
### Applicable Models: All

(Write-only) Deletes the trace associated with the specified measurement number.

**Note:** The measurement is not deleted. This command does the reverse of `DISP:MEAS:FEED`.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;mnum&gt;</code></td>
<td>Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, <code>&lt;mnum&gt;</code> is set to 1.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:MEAS:DEL</td>
<td></td>
</tr>
<tr>
<td>display:measure2:delete</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>DISP:MEASure&lt;mnum&gt;:DELETE?</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Not Applicable</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Default</th>
<th>Not Applicable</th>
</tr>
</thead>
</table>

### DISPlay:MEASure<mnum>:FEED <wnum>

**Applicable Models:** All

(Write-only) This command creates a new trace in the specified window and connects the trace to measurement which results in the trace displaying the data from measurement.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;mnum&gt;</code></td>
<td>Measurement number for the measurement. If unspecified, <code>&lt;mnum&gt;</code> is set to 1.</td>
</tr>
<tr>
<td><code>&lt;wnum&gt;</code></td>
<td>Display the measurement in a specified Window number. The window must be turned on. In addition, a window number must be specified. The range is 1 to 500.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:MEAS2:FEED 10</td>
<td></td>
</tr>
<tr>
<td>display:measure:feed 90</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Not Applicable</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Default</th>
<th>Not Applicable</th>
</tr>
</thead>
</table>

### DISPlay:MEASure<mnum>:MEMory[:STATe] <bool>
Applicable Models: All

(Read-Write) Turns the memory trace ON or OFF for the specified measurement.

Note: DISP:MEAS:FEED must first be done to feed the measurement to a trace. This command behaves the same as DISP:WIND:TRAC:MEM[:STAT] except that it only requires the measurement number.

Parameters
<mnum>
Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, <mnum> is set to 1.

<bool>
ON or 1 - Turns the memory trace ON.
OFF or 0 - Turns the memory trace OFF.

Examples
DISP:MEAS:MEM ON
display:measure:memory:state off

Query
Syntax
DISPlay:MEASure<mnum>:MEMory[:STATe]?

Return Type
Boolean
Default
OFF

DISPlay:MEASure<mnum>:MOVE <toWin>

Applicable Models: All

(Write-only) Moves a trace associated with measurement number to the specified window. If the window is OFF, it will be turn ON.

Parameters
<mnum>
Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, <mnum> is set to 1.

<toWin>
Number of the window to which the specified measurement is moved. If the window does not exist, it will be created.

Examples
DISP:MEAS:MOVE 2
display:measure:move 1

Query
Syntax
Not Applicable

Default
Not Applicable

DISPlay:MEASure<mnum>:SELect
**Applicable Models:** All

*(Write-only)* Activates the specified measurement to be selected.

**Parameters**

- `<mnum>` Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, `<mnum>` is set to 1.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:MEAS:SEL</td>
</tr>
<tr>
<td>display:measure:select</td>
</tr>
</tbody>
</table>

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**DISPlay:MEASure<mnum>[:STATe] <bool>**

**Applicable Models:** All

*(Read-Write)* Turns ON or OFF the display of a trace associated with the specified measurement. When OFF, the measurement behind the trace is still active.

Note: A trace must first be created (via FEED), then the visibility of the trace can be affected with this command. If the trace has not been created, an error is generated: 107, Requested trace not found.

**Parameters**

- `<mnum>` Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, `<mnum>` is set to 1.
- `<bool>` ON or 1 - Turns the trace ON.
  OFF or 0 - Turns the trace OFF.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:MEAS:STAT ON</td>
</tr>
<tr>
<td>display:measure off</td>
</tr>
</tbody>
</table>

**Query Syntax** `DISPlay:MEASure<mnum>[:STATe]?`

**Return Type** Boolean

**Default** ON or 1

---

**DISPlay:MEASure<mnum>:TITle:DATA <string>**
Applicable Models: All

(Read-Write) Sets or gets the title for the specified measurement. The trace title is embedded in the trace status field. Learn more about Trace Titles.

Newer entries replace (not append) older entries. The title is turned ON and OFF with \texttt{DISP:WIND:TRAC:TITL:STAT}.

**Parameters**

- \texttt{<mnum>} Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, \texttt{<mnum>} is set to 1.
- \texttt{<string>} Used as the title to be displayed for the measurement. Any characters (not spaces) enclosed with quotes.

**Examples**

\begin{verbatim}
DISP:MEAS:TITL:DATA 'MyNewMeas'
display:measure:title: data 'hello'
\end{verbatim}

**Query Syntax**

\texttt{DISPlay:MEASure<mnum>:TITLe:DATA?}

**Return Type**

String

**Default**

Not Applicable

\begin{verbatim}
DISPlay:MEASure<mnum>:TITLe[:STATe] <bool>
\end{verbatim}

Applicable Models: All

(Read-Write) Turns ON or OFF the measurement title.

Note: The measurement and trace need to exist. When turned OFF, the previous trace title returns. Set a new trace title using \texttt{DISP:WIND:TRAC:TITL:DATA}

Learn more about Trace Titles

**Parameters**

- \texttt{<mnum>} Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, \texttt{<mnum>} is set to 1.
- \texttt{<bool>} ON or 1 - turns the title ON.

**Examples**

\begin{verbatim}
DISP:MEAS:TITL ON
Display:measure:title:state off
\end{verbatim}

**Query Syntax**

\texttt{DISPlay:MEAS<mnum>:TITLe[:STATe]?
}

**Return Type**

Boolean

**Default**

OFF or 0
DISPlay:MEASure<mnum>:Y[:SCALe]:AUTO

**Applicable Models:** All

*(Write-only)* Performs an Autoscale on the specified trace in the specified measurement, providing the best fit display.

Autoscale is performed only when the command is sent; it does NOT keep the trace autoscaled indefinitely.

Autoscale behaves differently when *scale coupling* is enabled. How it behaves depends on the scale coupling method. [Learn more.]

See Also, **DISPlay:WINDow:Y:AUTO** which performs an Autoscale All.

**Parameters**
- `<mnum>` Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, `<mnum>` is set to 1.

**Examples**
- `DISP:MEAS:Y:AUTO`
- `display:measure:y:scale:auto`

**Query Syntax**
Not Applicable

**Default**
Not Applicable

---

**DISPlay:MEASure<mnum>:Y[:SCALe]:PDIVision <num>**

**Applicable Models:** All

*(Read-Write)* Sets the Y axis Scale Per Division value of the specified trace associated with the specified measurement.

**Note:** The measurement and trace need to exist.

**Parameters**
- `<mnum>` Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, `<mnum>` is set to 1.
- `<num>` Units / division value (Real value). The range of acceptable values is dependent on format and domain.

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See [SCPI Syntax](#) for more information.

**Examples**
- `DISP:MEAS:Y:PDIV 1`
- `display:measure:y:scale:pdivision maximum`

**Query Syntax**
- `DISP:MEASure<mnum>:Y[:SCALe]:PDIVision?`
**DISPlay:MEASure<mnum>:Y[:SCALe]:RLEVel <num>**

**Applicable Models:** All

(Read-Write) Sets the Y axis Reference Level of the specified trace associated with the specified measurement.

**Note:** The measurement and trace need to exist.

**Parameters**

- `<mnum>`: Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, `<mnum>` is set to 1.
- `<num>`: Reference level value (Real value). The range of acceptable values is dependent on format and domain.

**Examples**

```plaintext
DISP:MEAS:Y:RLEV 0
display:measure:y:scale:rlevel minimum
```

**Query Syntax**

DISPlay:MEASure<mnum>:Y[:SCALe]:RLEVel?

**Return Type**

Numeric

**Default**

0

---

**DISPlay:MEASure<mnum>:Y[:SCALe]:RPOSition <num>**
Applicable Models: All

(Read-Write) Sets the Reference Position of the specified trace associated with the specified measurement.

Note: The measurement and trace need to exist.

Parameters

- `<mnum>`: Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, `<mnum>` is set to 1.
- `<num>`: Reference position on the screen measured in horizontal graticules from the bottom (Real value). The range of acceptable values is dependent on format and domain.

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

- `DISP:MEAS:Y:RPOS 0`
- `display:measure:y:rposition maximum`

Query Syntax

`DISPlay:MEASure<mnum>:Y[:SCALe]:RPOsition?`

Return Type

Numeric

Default

5

Display:SHEet<num>:ARRange <char>

Applicable Models: All

(Write-only) This command arranges existing windows to sheets.

Parameters

- `<num>`: Sheet number
- `<char>`: Sheet arrangement. Choose from:
  - WINDow: one sheet per window
  - CHANnel: one sheet per channel
  - TRACe: one channel per sheet
  - ONE: merge all windows into one sheet

Examples

- `DISP:SHE:ARR CHAN`
- `display:sheet:arrange channel`
**Query Syntax**

**Return Type**

| Default | One sheet per window |

**Applicable Models:** All

**(Read-only)** This command reads and displays comma separated list of window numbers which the sheet contains.

**Parameters**

- `<num>`: Sheet number

**Examples**

- `DISP:SHE:CAT?`
- `display:sheet:catalog?`

**Return Type**: Character

| Default | 1 |

---

**Applicable Models:** All

**(Write-only)** Sets the sheet visible and invisible:

- **ON**: If OFF, sets the sheet visible with a new window.
- **OFF**: If ON, sets the sheet invisible with all the containing window state OFF (DISPlay:WINDow:STATe OFF)

**Parameters**

- `<num>`: Sheet number

**Examples**

- `DISP:SHE:STAT ON`
- `display:sheet:state off`

**Query Syntax**: DISPlay:SHEet:STATe?

**Return Type**: Bool

| Default | OFF except for Sheet 1 |
**Display:Sheet<num>:Title:Data <char>**

**Applicable Models:** All

*(Read-Write)* This command sets or gets the sheet label.

**Parameters**

- `<num>` Sheet number
- `<char>` The label of the sheets. Default and present value is "Sheet 1"

**Examples**

```
DISP:SHE:TITL:DATA "Sheet 1"
display: sheet: title: data "Sheet 1"
```

**Query Syntax**

DISPlay:SHEet:TITLe:DATA?

**Return Type** Character

**Default** "Sheet 1"

---

**Display:Split <num>**

**Applicable Models:** All

*(Write-only)* Destroys all existing traces, channels and windows, then creates N windows. No channels are created.

**Parameters**

- `<num>` N is 1 or greater.

**Examples**

```
DISP:SPL
display:split
```

**Query Syntax**

DISPlay:SPLit?

**Return Type** Numeric

**Default** Not Applicable

---

**Display:Status:Log:Clear**

**Applicable Models:** All

*(Write-only)* Clears the message region in the status bar.

**Parameters**

**Examples**

```
DISP:STAT:LOG:CLE
display:status:log:clear
```

**Query Syntax**

Not Applicable

**Default** Not Applicable
**DISPlay:TMAX <bool>**

**Applicable Models:** All

*(Read-Write)* Maximizes (isolates) or restores the active trace in the active window. When turned ON, the active trace is the ONLY trace on the display. All other traces are hidden. Learn more.

**Parameters**

<bool>  
ON (or 1) - Maximize / isolates the active trace.
OFF (or 0) - Restores other traces to the normal window setting.

**Examples**

```
DISP:TMAX ON
display:tmax 0
```

**Query Syntax**  
DISPlay:TMAX?

**Return Type**  
Boolean (1 = ON, 0 = OFF)

**Default**  
OFF

---

**DISPlay[:TILE] - Superseded**

This command is replaced by **DISP:ARRange**

*(Write-only)* Tiles the windows on the screen.

**Examples**

```
DISP
display:tile
```

**Default**  
Not Applicable

---

**DISPlay:TOOLbar:CSET[:STATe] <bool>**
Applicable Models: All

(Read-Write) Show or hide the calset toolbar.

Parameters

<bool>

ON (or 1) - Toolbar ON.

OFF (or 0) - Toolbar OFF.

Examples

DISP:TOOL:CSET ON
display:toolbar:cset:state off

Query Syntax

DISPlay:TOOLbar:CSET[:STATe]?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON

---

DISPlay:TOOLbar:ENTRy[:STATe] <bool>

Applicable Models: All

(Read-Write) Specifies whether to show or hide the active entry toolbar. See this toolbar.

Parameters

<bool>

ON (or 1) - Toolbar ON.

OFF (or 0) - Toolbar OFF.

Examples

DISP:TOOL:ENTR ON
display:toolbar:entry:state off

Query Syntax

DISPlay:TOOLbar:ENTRy[:STATe]?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON

---

DISPlay:TOOLbar:EXTensions[:STATe] <bool>
Applicable Models:  All

(Read-Write) Specifies whether to show or hide the port extensions toolbar. See this toolbar.

**Parameters**

- `<bool>`  
  - **ON** (1) - Toolbar ON.
  - **OFF** (0) - Toolbar OFF.

**Examples**

```plaintext
DISP:TOOL:EXT ON
display:toolbar:extensions:state off
```

**Query Syntax**  
DISPlay:TOOLbar:EXTensions[:STATe]?

**Return Type**  
Boolean (1 = ON, 0 = OFF)

**Default**  
ON

**DISPlay:TOOLbar:KEYS[:STATe] <bool>**

Applicable Models: All

(Read-Write) Specifies whether to show or hide the virtual hardkeys on the VNA display. These are primarily used when the VNA is accessed remotely using VNC or Windows Remote Desktop.

**Parameters**

- `<bool>`  
  - **ON** (1) - Keys ON.
  - **OFF** (0) - Keys OFF.

**Examples**

```plaintext
DISP:TOOL:KEYS ON
display:toolbar:keys:state off
```

**Query Syntax**  
DISPlay:TOOLbar:KEYS [:STATe]?

**Return Type**  
Boolean (1 = ON, 0 = OFF)

**Default**  
ON

**DISPlay:TOOLbar:MARKer[:STATe] <bool>**
Applicable Models: All

(Read-Write) Specifies whether to show or hide the marker toolbar. See this toolbar.

**Parameters**

<bool> ON (or 1) - Toolbar ON.

OFF (or 0) - Toolbar OFF.

**Examples**

| Display:TOOL:MARK ON | display:toolbar:marker:state off |

**Query Syntax**

DISPlay:TOOLbar:MARKer[:STATe]?

**Return Type**

Boolean (1 = ON, 0 = OFF)

Default ON

---

**DISPlay:TOOLbar:MEASurement[:STATe] <bool> OBSOLETE**

This toolbar was eliminated with A.10.00

(Read-Write) Specifies whether to show or hide the measurement toolbar.

**Parameters**

<bool> ON (or 1) - Toolbar ON.

OFF (or 0) - Toolbar OFF.

**Examples**

| DISP:TOOL:MEAS ON | display:toolbar:measurement:state off |

**Query Syntax**

DISPlay:TOOLbar:MEASurement[:STATe]?

**Return Type**

Boolean (1 = ON, 0 = OFF)

Default ON

---

**DISPlay:TOOLbar:STIMulus[:STATe] <bool> OBSOLETE**
This toolbar was eliminated with A.10.00

(Read-Write) Specifies whether to show or hide the stimulus toolbar.

Parameters

<bool>  
ON (or 1) - Toolbar ON.
OFF (or 0) - Toolbar OFF.

Examples

DISP:TOOL:STIM ON
display:toolbar:stimulus:state off

Query Syntax

DISPlay:TOOLbar:STIMulus[:STATe]?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON

DISPlay:TOOLbar:SWEep[:STATe] <bool> OBSOLETE

This toolbar was eliminated with A.10.00

(Read-Write) Specifies whether to show or hide the sweep control toolbar.

Parameters

<bool>  
ON (or 1) - Toolbar ON.
OFF (or 0) - Toolbar OFF.

Examples

DISP:TOOL:SWE ON
display:toolbar:sweep:state off

Query Syntax

DISPlay:TOOLbar:SWEep[:STATe]?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON
Applicable Models: All

(Read-Write) Specifies whether to show or hide the Time Domain toolbar. See this toolbar.

Parameters

<bool>  
ON (or 1) - Toolbar ON.
OFF (or 0) - Toolbar OFF.

Examples  
DISP:TOOL:TRAN ON
display:toolbar:transform:state off

Query Syntax  
DISPlay:TOOLbar:TRANsform[:STATe]?

Return Type  
Boolean (1 = ON, 0 = OFF)

Default  
ON

DISPlay:UPDATE[:STATe] <bool>

Applicable Models: All

(Read-Write) Enables or disables display updates. Disabling display updates improves measurement performance. When disabled, the display windows (traces, markers, etc.) are frozen.

Parameters

<bool>  
ON (or 1) - Toolbar ON.
OFF (or 0) - Toolbar OFF.

Examples  
DISP:UPD ON
display:update:state off

Query Syntax  
DISPlay:UPDate[:STATe]?

Return Type  
Boolean (1 = ON, 0 = OFF)

Default  
ON

DISPlay:UPDATE:IMMediate
Applicable Models: All

(Write-only) Executes the display update once when the display update of the LCD screen is set to OFF (specifying False with the DISPlay:ENABle object).

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DISP:UPD:IMM</td>
</tr>
<tr>
<td></td>
<td>display: update: immediate</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Not Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

DISPlay:VISible <ON | OFF>

Applicable Models: All

(Read-Write) Makes the VNA application visible or not visible. In the Not Visible state, the analyzer cycle time for making measurements, and especially data transfer, can be significantly faster because the display does not process data.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>&lt;ON</th>
<th>OFF&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ON (or 1) - VNA app is visible</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OFF (or 0) - VNA app is NOT visible</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
<th>DISP:VIS ON</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>display:visible off</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>DISPlay:VISible?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Boolean (1 = ON, 0 = OFF)</td>
</tr>
<tr>
<td>Default</td>
<td>ON</td>
</tr>
</tbody>
</table>

DISPlay:WINDow<wnum>:ANNotation:LIMit:XPOSition <num>
Applicable Models: All

(Read-Write) Sets and returns the X-axis position of the Limit Line Pass/Fail indicator on the VNA screen. The lower-left corner of the Pass/Fail indicator is the point of reference for positioning.

Parameters

| <wnum> | Any existing window number. If unspecified, value is set to 1. |
| <num> | X-axis position. Choose a value between 0 (far left) and 10 (far right). |

Examples

```
DISP:WIND:ANN:LIM:XPOS 1.5
```
```
display:window:annotation:limit:xposition 5
```

Query Syntax

DISPlay:WINDow:ANNotation:LIMit:XPOSition?

Return Type

Numeric

Default

7

DISPlay:WINDow<wnum>:ANNotation:LIMit:YPOSition <num>

Applicable Models: All

(Read-Write) Sets and returns the Y-axis position of the Limit Line Pass/Fail indicator on the VNA screen. The lower-left corner of the Pass/Fail indicator is the point of reference for positioning.

Parameters

| <wnum> | Any existing window number. If unspecified, value is set to 1. |
| <num> | Y-axis position. The maximum position is limited to the current Y-axis division value. Choose a value between 2 (bottom) and 30 (top). |

Examples

```
DISP:WIND:ANN:LIM:YPOS 1.5
```
```
display:window:annotation:limit:yposition 5
```

Query Syntax

DISPlay:WINDow:ANNotation:LIMit:YPOSition?

Return Type

Numeric

Default

0

DISPlay:WINDow<wnum>:ANNotation:MARKer:COUPle[:STATe] <bool>

Applicable Models: All
(Read-Write) Sets the marker readouts to coupled (one combination annotation) or not coupled (one annotation per trace). This setting is per Window scope.

See other SCPI Marker commands. Learn more about Marker readout.

Parameters

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<bool>` ON (or 1) - Marker readouts are coupled
  OFF (or 0) - Marker readouts are not coupled

Examples

```markdown
DISP:WIND:ANN:MARK:COUP ON
display:window:annotation:marker:couple on
```

Query Syntax

`DISPlay:WINDow:ANNotation:MARKer:COUPle?`

Return Type

Boolean

Default

ON

`DISPlay:WINDow<wnum>:ANNotation:MARKer:NUMBer <num>`

Applicable Models: All

This command replaces `DISP:WIND:ANN:MARK:SINGle`

(Read-Write) Sets the number of marker readouts to display per trace. Display up to 20 marker readouts per window.

See other SCPI Marker commands. Learn more about Marker readout.

Parameters

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<num>` Number of marker readouts to display. Choose a value between 1 and 16.

Examples

```markdown
DISP:WIND:ANN:MARK:NUMB 7
display:window:annotation:marker:number 2
```

Query Syntax

`DISPlay:WINDow:ANNotation:MARKer:NUMBer?`

Return Type

Numeric

Default

5

`DISPlay:WINDow:ANNotation:MARKer:MEASure<mnum>:XPOSition <num>`

Applicable Models: All
(Read-Write) Sets the X-axis position of marker readouts. Readouts are right-justified at the specified position. This function is used when :DISP:WIND:ANN:MARK:COUP:STAT is off. Use :DISP:WIND:ANN:MARK:XPOS is used when :DISP:WIND:ANN:MARK:COUP:STAT is on.

See other SCPI Marker commands. Learn more about Marker readout.

**Parameters**

- `<mnum>` Measurement. If unspecified, value is set to 1.
- `<num>` X-axis position. Choose a value between 1 (far left) and 10 (far right).

**Examples**

```
display:window:annotation:marker:measure:xposition 5
```

**Query Syntax**

DISPlay:WINDow:ANNotation:MARKer:MEASure<mnum>:YPOSitio<wnum>

**Return Type**

Numeric

**Default**

10

DISPlay:WINDow:ANNotation:MARKer:MEASure<mnum>:YPOSitio<wnum>

Applicable Models: All

(Read-Write) Sets the Y-axis position of marker readouts. Readouts are top-justified at the specified position. This function is used when :DISP:WIND:ANN:MARK:COUP:STAT is off. Use :DISP:WIND:ANN:MARK:YPOS is used when :DISP:WIND:ANN:MARK:COUP:STAT is on.

See other SCPI Marker commands. Learn more about Marker readout.

**Parameters**

- `<mnum>` Measurement. If unspecified, value is set to 1.
- `<num>` Y-axis position. Choose a value between 1 (bottom) and 10 (top).

**Examples**

```
display:window:annotation:marker:measure:yposition 5
```

**Query Syntax**

DISPlay:WINDow:ANNotation:MARKer:MEASure<mnum>:YPOSitio<wnum>

**Return Type**

Numeric

**Default**

10

DISPlay:WINDow<wnum>:ANNotation:MARKer:RESolution:STIMulus <num>
Applicable Models: All

(Read-Write) For the X-axis (stimulus), sets the number digits to display after the decimal point in marker readouts.

See other SCPI Marker commands. Learn more about Marker readout.

Parameters

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<num>` Number of digits to display. Choose a value between 2 and 6.

Examples

```
DISP:WIND:ANN:MARK:RES:STIM 2
display:window:annotation:marker:resolution:stimulus 4
```

Query Syntax

```
DISPlay:WINDow:ANNotation:MARKer:RESolution:STIMulus?
```

Return Type

Numeric

Default

3

---

DISPlay:WINDow<wnum>:ANNotation:MARKer:RESolution:RESPonse <num>

Applicable Models: All

(Read-Write) For the Y-axis (response), sets the number digits to display after the decimal point in marker readouts.

See other SCPI Marker commands. Learn more about Marker readout.

Parameters

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<num>` Number of digits to display. Choose a value between 1 and 4.

Examples

```
DISP:WIND:ANN:MARK:RES:RESP 1
display:window:annotation:marker:resolution:stimulus 2
```

Query Syntax

```
DISPlay:WINDow:ANNotation:MARKer:RESolution:RESPonse?
```

Return Type

Numeric

Default

2

---

DISPlay:WINDow<wnum>:ANNotation:MARKer:SINGle[:STATe] <bool> - Superseded
Applicable Models: All

**Note:** This command is replaced by `DISP:WIND:ANN:MARK:NUMB` (Read-Write) Either shows marker readout of only the active trace or other traces simultaneously.

See other SCPI Marker commands. Learn more about Marker readout.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<bool>`
  - **ON** (or 1) - Shows the readout of only the active marker for each trace.
  - **OFF** (or 0) - Shows up to 5 marker readouts per trace, up to 20 total readouts.

**Examples**

```
DISP:WIND:ANN:MARK:SING ON
display:window:annotation:marker:single off
```

**Query Syntax**

`DISPlay:WINDow:ANNotation:MARKer:SINGle?`

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

---

**DISPlay:WINDow<wnum>:ANNotation:MARKer:SIZE <char>**

Applicable Models: All

(Read-Write) Specifies the size of the marker readout text. See other SCPI Marker commands. Learn more about Marker readout.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<char>` Readout text size. Choose from: NORMAL | LARGE

**Examples**

```
DISP:WIND:ANN:MARK:SIZE LARG
display:window:annotation:marker:size normal
```

**Query Syntax**

`DISPlay:WINDow:ANNotation:MARKer:SIZE?`

**Return Type**

Character

**Default**

NORMAL

---

DISPlay:WINDow<wnum>:ANNotation:MARKer[:STATe] <ON | OFF>
**Applicable Models:** All

(Read-Write) Specifies whether to show or hide the Marker readout (when markers are ON) on the selected window. See other SCPI Marker commands. Learn more about Marker readout.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<ON | OFF>` **ON** (or 1) - turns marker readout ON.
  **OFF** (or 0) - turns marker readout OFF.

**Examples**

- `DISP:WIND:ANN:MARK ON`  
  `display:window:annotation:marker:state on`
- `DISP:WIND:ANN:MARK:OFF`  
  `display:window:annotation:marker:state off`

**Query Syntax**

`DISP:WIND:ANN:MARK[:STAT]e?`

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON

**Applicable Models:** All

(Read-Write) Sets the symbol to display for marker position.

See other SCPI Marker commands.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<char>` Marker symbol. Choose from:
  
  - TRIangle
  - FLAG
  - LINE

**Examples**

- `DISP:WIND:ANN:MARK:SYMB TRI`  
  `display:window:annotation:marker:symbol triangle`
- `DISP:WIND:ANN:MARK:SYMB LINE`  
  `display:window:annotation:marker:symbol line`

**Query Syntax**

`DISP:WIND:ANN:MARK:SYMB?`

**Return Type**

Character

**Default**

TRIangle
DISPlay:WINDow<wnum>:ANNotation:MARKer:VISible <char>

**Applicable Models:** All

*(Read-Write)* Shows the marker readouts only for active trace or for all traces. This setting is per Window scope. See this toolbar.

**Parameters**

- **<wnum>** Any existing window number. If unspecified, value is set to 1.
- **<char>** ACTive - readout is turned on for active trace only.
- **ALL** - readout is turned on for all traces.

**Examples**

<table>
<thead>
<tr>
<th>DISPlay:WIND:ANN:MARK:VIS ACT</th>
</tr>
</thead>
<tbody>
<tr>
<td>display:windows:annotation:marker:visible:all</td>
</tr>
</tbody>
</table>

**Query Syntax**

DISPlay:WINDow:ANNotation:MARKer:VISible?

**Return Type** Character

**Default** ALL

---

DISPlay:WINDow<wnum>:ANNotation:MARKer:SYMBol:ABOVe[:STATe] <ON | OFF>

**Applicable Models:** E5080, M9485A

*(Read-Write)* Specifies whether or not to force marker symbols to be displayed above the trace. When ON, all marker symbols will be displayed above the trace and the active marker will be filled solid. See other SCPI Marker commands.

**Parameters**

- **<wnum>** Any existing window number. If unspecified, value is set to 1.
- **<ON | OFF>**
  - **ON (or 1)** - ALL marker symbols are displayed above the trace. Only the active marker is filled solid.
  - **OFF (or 0)** - ONLY the active marker is displayed above the trace. The active marker is not filled solid.

**Examples**

<table>
<thead>
<tr>
<th>DISPlay:WIND:ANN:MARK:SYMB:ABOV ON</th>
</tr>
</thead>
<tbody>
<tr>
<td>display:window:annotation:marker:symbol:above:state off</td>
</tr>
</tbody>
</table>

**Query Syntax**

DISPlay:WINDow:ANNotation:MARKer:SYMBol:ABOVe[:STATe]?

**Return Type** Boolean (1 = ON, 0 = OFF)
**DISPlay:WINDow<wnum>:ANNotation:MARKer:XPOSition <num>**

**Applicable Models:** All

*(Read-Write)* Sets the X-axis position of marker readouts. Readouts are right-justified at the specified position.

See other SCPI Marker commands. Learn more about Marker readout.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<num>` X-axis position. Choose a value between 1 (far left) and 10 (far right).

**Examples**

```
DISP:WIND:ANN:MARK:XPOS 1.5
display:window:annotation:marker:xposition 5
```

**Query Syntax** DISPlay:WINDow:ANNotation:MARKer:XPOSition?

**Return Type** Numeric

**Default** 10

---

**DISPlay:WINDow<wnum>:ANNotation:MARKer:YPOSition <num>**

**Applicable Models:** All

*(Read-Write)* Sets the Y-axis position of marker readouts. Readouts are top-justified at the specified position.

See other SCPI Marker commands. Learn more about Marker readout.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<num>` Y-axis position. Choose a value between 1 (bottom) and 10 (top).

**Examples**

```
DISP:WIND:ANN:MARK:YPOS 1.5
display:window:annotation:marker:yposition 5
```

**Query Syntax** DISPlay:WINDow:ANNotation:MARKer:YPOSition?

**Return Type** Numeric

**Default** 10
DISPlay:WINDow<wnum>:ANNotation[:TRACe]:SCOPe <enum>

Applicable Models: All

(Read-Write) Set and read the scope for trace annotation. Either display trace annotation for all traces or just the active trace.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.
<enum> Choose from:

ALL
ACTive

Examples DISP:WIND:ANN:SCOP ACT

Query Syntax DISPlay:WINDow:ANNotation[:TRACe]:SCOPe?
Return Type Enumeration

Default ALL

DISPlay:WINDow<wnum>:ANNotation[:TRACe][:STATe] <ON | OFF>

Applicable Models: All

(Read-Write) Specifies whether to show or hide the Trace Status buttons on the left of the display.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.
<ON | OFF> ON (or 1) - turns the buttons ON.
OFF (or 0) - turns the buttons OFF.

Examples DISP:WIND:ANN ON
display:window:annotation:trace:state off

Query Syntax DISPlay:WINDow:ANNotation[:TRACe][:STATe]?
Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:WINDow<wnum>:ANNotation: X[:STATe] <bool>
Applicable Models: All

(Read-Write) Turns ON or OFF the X-axis scale label in display window.

Parameters

<wnum>  Any existing window number. If unspecified, value is set to 1.

<bool>  ON or 1 - Turns ON the X-axis scale.
        OFF or 0 - Turns OFF the X-axis scale.

Examples

DISP:WIND:ANN:X ON
display:window:annotation:x off

Query Syntax

DISPlay:WINDow:ANNotation:X?

Return Type  Boolean

Default  ON or 1

DISPlay:WINDow<wnum>:ANNotation: Y[:STATe] <bool>

Applicable Models: All

(Read-Write) Turns ON or OFF the Y-axis scale label in display window.

Parameters

<wnum>  Any existing window number. If unspecified, value is set to 1.

<bool>  ON or 1 - Turns ON the Y-axis scale.
        OFF or 0 - Turns OFF the Y-axis scale.

Examples

DISP:WIND:ANN:Y ON
display:window:annotation:y off

Query Syntax

DISPlay:WINDow:ANNotation:Y?

Return Type  Boolean

Default  ON or 1

DISPlay:WINDow<wnum>:CATalog?
Applicable Models: All

(Read-only) Returns the trace numbers for the specified window.

**Note:** If there are no traces in the window, this query returns the "EMPTY" string.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.

**Example**

Window 1 with four traces:

```
DISPlay:WINDow1:CATalog?
Returns:
"1,2,3,4"
```

**Return Type**

- String of Character values separated by commas

**Default**

- Not applicable

**DISPlay:WINDow<wnum>:ENABle <ON | OFF>**

Applicable Models: All

(Read-Write) Specifies whether to disable or enable all analyzer display information in the specified window. Marker data is not updated. More CPU time is spent making measurements instead of updating the display.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<ON | OFF>`
  - ON (or 1) - turns the display ON.
  - OFF (or 0) - turns the display OFF.

**Examples**

```
DISP:WIND:ENABLE ON
display:window1:enable off
```

**Query Syntax**

```
DISPlay:WINDow<wnum>:ENABle?
```

**Return Type**

- Boolean (1 = ON, 0 = OFF)

**Default**

- ON

**DISPlay:WINDow<wnum>:FEED <snum>**
(Write-only) This command feeds a specified window to the sheet. If there is a window in the sheet, the sheet is visible. If there is no window in the sheet, the sheet is not visible. If no windows exists in the system, one empty sheet is visible.

**Parameters**

- `<wnum>`: Any existing window number. If unspecified, value is set to 1.
- `<snum>`: Sheet number

**Examples**

- `DISP:WIND:FEED 5`
- `display:window:feed 5`

**Return Type**: Not Applicable

**Default**: Not Applicable

---

**DISPlay:WINDow<wnum>:NEXT[:NUMBer]?**

(Read-only) Returns the lowest window number which has less than the maximum number of traces. Basically, returns the first window which has room for another trace. Note that the window may need to be turned on first (i.e. disp:wind:stat ON may be needed).

**Parameters**

- `<wnum>`: Any existing window number. If unspecified, value is set to 1.

**Examples**

- `DISP:WIND:NEXT`
- `display:window1:NEXT`

**Query Syntax**: DISPlay:WINDow<wnum>:NEXT?

**Return Type**: Not Applicable

**Default**: Not Applicable

---

**DISPlay:WINDow<wnum>:SIZE <char>**
**Applicable Models:** All

*(Read-Write)* Sets or returns the window setting of Maximized or Normal. To arrange all of the windows, use `DISP:ARR`.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1
- `<char>` Window size. Choose from:

<table>
<thead>
<tr>
<th>MAX</th>
<th>NORM</th>
</tr>
</thead>
</table>

**Examples**

```markdown
DISP:WIND:SIZE MAX
display:window:size norm
```

**Query Syntax**

`DISPlay:WINDow:SIZE?`

**Default**

Not Applicable

---

**DISPlay:WINDow<wnum>[::STATe] <ON | OFF>**

**Applicable Models:** All

*(Read-Write)* Write to create or delete a window on the screen or Read whether a window is present.

**Parameters**

- `<wnum>` Window number to create; choose any integer between 1 and the maximum number of windows allowed in the VNA.
- `<ON | OFF>`
  - **ON** (or 1) - The window `<wnum>` is created.
  - **OFF** (or 0) - The window `<wnum>` is deleted.

**Examples**

```markdown
DISP:WIND ON
display:window2:state off
```

**Query Syntax**

`DISPlay:WINDow<wnum>[::STATe]?`

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

Window number "1" ON

---

**DISPlay:WINDow<wnum>:TABLe <char>**
Applicable Models: All

(Read-Write) Write to show the specified table at the bottom of the analyzer screen or Read to determine what table is visible.

Parameters

- `<wnum>` Any existing window number. If unspecified, value is set to 1
- `<char>` Table to show. Choose from:
  - OFF | MARKer | LIMit | SEGment | RLIMit | DISTortion

Examples

- `DISP:WIND:TABLE SEGM`
- `display:window:table off`

Query Syntax

- `DISPlay:WINDow<wnum>:TABLe?`
  - **Default**: OFF

**DISPlay:WIND<wnum>:TABLe:INOise:ENABle <ON | OFF>**

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Enable or disable displaying the integrated noise table.

Parameters

- `<wnum>` Window number to create; choose any integer between 1 and the maximum number of windows allowed in the VNA.
- `<ON | OFF>` ON (or 1) - Display the integrated noise table.
  - OFF (or 0) - Do not display the integrated noise table.

Examples

- `DISP:WIND:TABLE:INO:ENAB  ON`
- `display:window2:table:inoise:enable off`

Query Syntax

- `DISPlay:WINDow<wnum>:TABLe:INOise:ENABle?`
  - **Return Type**: Boolean (1 = ON, 0 = OFF)
  - **Default**: OFF

**DISPlay:WIND<wnum>:TABLe:SNOise:ENABle <ON | OFF>**
**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

*(Read-Write)* Enable or disable displaying the spot noise table.

**Parameters**

- `<wnum>` Window number to create; choose any integer between 1 and the maximum number of windows allowed in the VNA.
- `<ON | OFF>` ON (or 1) - Display the spot noise table. OFF (or 0) - Do not display the spot noise table.

**Examples**

```
DISP:WIND:TABL:SNO:ENAB ON
display:window2:table:snoise:enable off
```

**Query Syntax** DISPLAY:WIND<wnum>:TABLe:SNOise:ENABle?

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** OFF

**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

*(Read-Write)* Enable or disable displaying the spurious table.

**Parameters**

- `<wnum>` Window number to create; choose any integer between 1 and the maximum number of windows allowed in the VNA.
- `<ON | OFF>` ON (or 1) - Display the spurious table. OFF (or 0) - Do not display the spurious table.

**Examples**

```
DISP:WIND:TABL:SPUR:ENAB ON
display:window2:table:spurious:enable off
```

**Query Syntax** DISPLAY:WIND<wnum>:TABLe:SPURious:ENABle?

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** OFF

**DISPlay:WIND<wnum>:TABLe:DATA <string>**
Applicable Models: All

(Read-Write) Sets data in the window title area. The title is turned ON and OFF with DISP:WIND:TITL:STAT OFF.

Parameters

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<string>` Title to be displayed. Any characters, enclosed with quotes. If the title string exceeds 50 characters, an error will be generated and the title not accepted. Newer entries replace (not append) older entries.

Examples

```
DISP:WIND:TITL:DATA 'hello'
display:window2:title:data 'hello'
```

Query Syntax

DISPlay:WINDow<wnum>:TITLe:DATA?

Return Type

String

Default

NA

DISPlay:WINDow<wnum>:TITLe[:STATe] <ON | OFF>

Applicable Models: All

(Read-Write) Turns display of the title string ON or OFF. When OFF, the string remains, ready to be redisplayed when turned back ON.

Parameters

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<ON | OFF>`
  - `ON` (or 1) - turns the title string ON.
  - `OFF` (or 0) - turns the title string OFF.

Examples

```
DISP:WIND:TITL ON
display:window1:title:state off
```

Query Syntax

DISPlay:WINDow<wnum>:TITLe[:STATe]?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON
**Applicable Models:** All

(Write-only) Deletes the specified trace from the specified window. The measurement parameter associated with the trace is not deleted.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<tnum>` The number of the trace to be deleted; if unspecified, value is set to 1.

**Note:** This is NOT the trace number of the channel which appears as the Tr annotation on the Trace Status display. This `<tnum>` is the trace number within the specified window, and is used ONLY for remote programs.

**Examples**

```
DISP:WIND:TRAC:DEL
display:window2:trace2:delete
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**DISPlay:WINDow<wnum>:TRACe<tnum>:FEED <name>**

**Applicable Models:** All

(Write-only) Creates a new trace `<tnum>` and associates (feeds) a measurement `<name>` to the specified window `<wnum>`. This command should be executed after creating a new measurement with CALCulate:MEASure:DEFine.

**Note:** Window `<wnum>` must already be turned on before this command can be used. Turn on the window state using DISPPlay:WINDow:STATe.

To feed the same measurement to multiple traces, create another measurement with the same `<parameter>`, but different `<name>`, using the CALC:PAR:DEF command. The analyzer will collect the data only once.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<tnum>` Trace number to be created. Choose any Integer between 1 and the VNA maximum number of traces per window allowed.

**Note:** This is NOT the trace number of the channel which appears as the Tr annotation on the Trace Status display. This `<tnum>` is the trace number within the specified window, and is used ONLY for remote programs.
Name of the measurement that was defined with CALC:PAR:DEF <name>,<parameter>

Example

```
CALCulate1:PARameter:DEFine 'CH1_S11',s11,1
CALCulate1:PARameter:SELect 'CH1_S11'
DISPlay:WINDow1:STATe ON
DISPlay:WINDow1:TRACe1:FEED 'CH1_S11'
```

Query Syntax

Not applicable

Default

"CH1_S11"

DISPlay:WINDow<wnum>:TRACe<tnum>:FEED:MNUMber <int>

Applicable Models: All

(Write-only) Creates a new trace <tnum> for an existing measurement (MNUM) and associates (feeds) the measurement number to the specified window <wnum>. A measurement is created using the CALC:MEAS:DEF command.

Measurements created in the system all have unique numbers. Similarly, every window has a unique number and the numbers are displayed in the lower-left corner of each window. Every window has the capacity to hold a finite number of traces from 1 to N, where N is the maximum number of traces per window. Each window uses the same range of trace numbers. For example, window 1 can have a trace 1 and so can window 2.

Parameters

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<tnum>` Trace number to be created. Choose any Integer between 1 and the VNA maximum number of traces per window allowed.

**Note:** After executing the DISPlay:WIND:TRAC:FEED:MNUM command, a new trace is added to the specified window and the trace number of the channel which appears as the Tr annotation on the Trace Status display is the actual measurement number.

- `<int>` Number of an existing measurement. The range is 1 to 2000.

Examples

```
CALC:MEAS2:DEF "S22"
DISP:WIND:TRAC4:FEED:MNUM 2
```

Query Syntax

Not applicable

Default

Not applicable
**DISPlay:WINDow:TRACe:GRATicule:GRID:LTYPE <value>**

**Applicable Models:** All

*(Read-Write)* Sets and returns the grid line type (solid | dotted) for all open windows. Grid is returned to solid when the VNA is Preset. Learn more.

**Parameters**

- `<value>` Line type. Choose from:
  - **SOLid** - solid lines
  - **DOTTed** - dotted lines

**Examples**

```
DISP:WIND:TRAC:GRAT:GRID:LTYPE SOL
```

**Query Syntax**

```
DISPlay:WINDow:TRACe:GRATicule:GRID:LTYPE?
```

**Return Type** Character

**Default** SOLID

---

**DISPlay:WIND<wnum>:TRACe<tnum>:MEMory[:STATe] <ON | OFF>**

**Applicable Models:** All

*(Read-Write)* Turns the memory trace ON or OFF.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<tnum>` Any existing trace number; if unspecified, value is set to 1.

**Note:** This is **NOT** the trace number of the channel which appears as the Tr annotation on the Trace Status display. This `<tnum>` is the trace number within the specified window, and is used ONLY for remote programs.

- `<ON | OFF>`
  - **ON** (or 1) - turns the memory trace ON.
  - **OFF** (or 0) - turns the memory trace OFF.

**Examples**

```
DISP:WIND:TRAC:MEM ON
```

**Query Syntax**

```
DISPlay:WIND<wnum>:TRACe<tnum>:MEMory[:STATe]?
```

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** OFF

---

3925
DISPlay:WINDow<fromWin>:TRACe<tnum>:MOVE <toWin>

Applicable Models: All

(Write-only) Moves a trace from one window to another window.

Parameters

<fromWin> Window number to move the trace from. If unspecified, value is set to 1.

Use Disp:Cat? to read the existing window numbers.

<tnum> Trace number to be moved. If unspecified, value is set to 1.

Use Disp:Wind:Cat? to read the trace numbers in an existing window.

Note: This is NOT the trace number of the channel which appears as the Tr annotation on the Trace Status display. This <tnum> is the trace number within the specified window, and is used ONLY for remote programs.

[toWin] Number of the window to move the trace to. If the window does not exist, it will be created.

Examples

DISP:WIND:TRAC2:MOVE 2
DISPLAY:WINDOW2:TRACE2:MOVE 1

Query Syntax

Not applicable

Default

Not applicable

DISPlay:WINDow<wnum>:TRACe:NEXT[:NUMBer]?

Applicable Models: All

(Read-only) Returns the next unused trace number. For example, if trace #1, #2, and #3 are being used, then this command will return 4.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

Examples

DISP:WIND:TRAC:NEXT?
DISPLAY:WINDOW1:TRACE:NEXT?

Return Type

Integer

Default

Not Applicable

DISPlay:WINDow<wnum>:TRACe<tnum>:SELect
Applicable Models: All

(Write-only) Activates the specified trace in the specified window for front panel use.

Parameters
<wnum> Any existing window number. If unspecified, value is set to 1.
Use Disp:Cat? to read the existing window numbers.
<tnum> Any existing trace number; if unspecified, value is set to 1.

Note: This is NOT the trace number of the channel which appears as the Tr annotation on the Trace Status display. This <tnum> is the trace number within the specified window, and is used ONLY for remote programs.

Examples
DISP:WIND:TRAC:SEL
display:window2:trace2:select

Query Syntax
Default NA

DISPlay:WINDow<wnum>:TRACe<tnum>[:STATe] <ON | OFF>

Applicable Models: All

(Read-Write) Turns the display of the specified trace in the specified window ON or OFF. When OFF, the measurement behind the trace is still active.

Parameters
<wnum> Any existing window number. If unspecified, value is set to 1.
Use Disp:Cat? to read the existing window numbers.
<tnum> Any existing trace number; if unspecified, value is set to 1.
Use Disp:Wind:Cat? to read the trace numbers in an existing window.

Note: This is NOT the trace number of the channel which appears as the Tr annotation on the Trace Status display. This <tnum> is the trace number within the specified window, and is used ONLY for remote programs.

Examples
DISP:WIND:TRAC ON
display:window2:trace2:state off

Query Syntax
DISPlay:WIND<wnum>:TRACe<tnum>[:STATe]?
**Return Type**  
Boolean (1 = ON, 0 = OFF)  
**Default**  
ON

**DISPlay:WINDow<wnum>:TRACe<tnum>:TITLe:DATA <string>**

**Applicable Models:** All

*(Read-Write)* Writes and read data to the trace title area. The trace title is embedded in the trace status field. [Learn more about Trace Titles](#).

Newer entries replace (not append) older entries. The title is turned ON and OFF with **DISP:WIND:TRAC:TITL:STAT**.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
  
  Use **Disp:Cat?** to read the existing window numbers.

- `<tnum>` Trace number of the specified window. If unspecified, value is set to 1. Use **Display:Cat?** to read the window numbers. Use **Disp:Window:Cat?** to read the trace numbers of the specified window.

  **Note:** This is **NOT** the trace number of the channel which appears as the Tr annotation on the Trace Status display. This `<tnum>` is the trace number within the specified window, and is used ONLY for remote programs.

- `<string>` Title to be displayed. Any characters (not spaces) enclosed with quotes.

**Examples**

- **DISP:WIND:TRAC:TITL:DATA 'MyNewMeas'**
- **display:window2:trace3:title:data 'hello'**

**Query Syntax**  
**DISPlay:WINDow<wnum>:TRACe<tnum>TITLe:DATA?**

**Return Type**  
String  
**Default**  
Not Applicable

**DISPlay:WINDow<wnum>:TRACe<tnum>:TITLe[:STATe] <bool>**
Applicable Models: All

*(Read-Write)* Turns display of the Trace Title ON or OFF. When turned OFF, the previous trace title returns. Set a new trace title using **DISP:WIND:TRAC:TITL:DATA**

Learn more about Trace Titles

**Parameters**

- **<wnum>** Any existing window number. If unspecified, value is set to 1
  
  Use **Disp:Cat?** to read the existing window numbers.
  
- **<tnum>** Trace number of the specified window. If unspecified, value is set to 1. Use **Display:Cat?** to read the window numbers. Use **Disp:Window:Cat?** to read the trace numbers of the specified window.

  **Note:** This is **NOT** the trace number of the channel which appears as the Tr annotation on the Trace Status display. This **<tnum>** is the trace number within the specified window, and is used ONLY for remote programs.

- **<bool>** ON (or 1) - turns the title ON.

  OFF (or 0) - turns the title OFF.

**Examples**

- **DISP:WIND:TRAC:TITL ON**
- **Display:window2:trace3:title:state off**

**Query Syntax**

**DISPlay:WINDow<wnum>:TRACe<tnum>:TITLe[:STATe]?**

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON

**DISPlay:WINDow<wnum>:TRACe<tnum>:X[:SCALe]:RLEVel <num>**
**Applicable Models:** All  

*(Read-Write)* Sets the X axis **Reference Level** of the specified trace in the specified window.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.  
  Use `Disp:Cat?` to read the existing window numbers.
- `<tnum>` Any existing trace number; if unspecified, value is set to 1.  
  Use `Disp:Wind:Cat?` to read the trace numbers in an existing window.
- `<num>` Reference level value.

**Examples**

```
DISP:WIND:TRAC:X:RLEV 0  
display:window2:trace2:x:scale:rlevel 0
```

**Query Syntax**

`DISPlay:WINDow<wnum>:TRACe<tnum>:X[:SCALe]:RLEVel?`

**Return Type** Numeric

**Default** Not Applicable

---

**Applicable Models:** All  

*(Read-Write)* Sets the **Reference Position** of the specified trace in the specified window.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.  
  Use `Disp:Cat?` to read the existing window numbers.
- `<tnum>` Any existing trace number; if unspecified, value is set to 1.  
  Use `Disp:Wind:Cat?` to read the trace numbers in an existing window.

**Note:** This is **NOT** the trace number of the channel which appears as the Tr annotation on the Trace Status display. This `<tnum>` is the trace number within the specified window, and is used ONLY for remote programs.

**Examples**

```
DISPlay:WINDow<wnum>:TRACe<tnum>:X[:SCALe]:RPOSition <num>
```

**Query Syntax**

`DISPlay:WINDow<wnum>:TRACe<tnum>:X[:SCALe]:RPOSition <num>`
Reference position on the screen measured in horizontal graticules from the bottom. Choose a value between 0 and 10.

| Examples | DISP:WIND:TRAC:X:RPOS 0  
display:window2:trace2:x:rposition 10 |
|---|---|

Example Syntax

DISPlay:WINDow<wnum>:TRACe<tnum>:X[:SCALe]:RPOSition?

Return Type

Numeric

Default

5

DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:AUTO

Applicable Models: All

(Write-only) Performs an Autoscale on the specified trace in the specified window, providing the best fit display.

Autoscale is performed only when the command is sent; it does NOT keep the trace autoscaled indefinitely.

Autoscale behaves differently when scale coupling is enabled. How it behaves depends on the scale coupling method. Learn more.

See Also, DISP:WINDow:Y:AUTO which performs an Autoscale All.

Parameters

<table>
<thead>
<tr>
<th>&lt;wnum&gt;</th>
<th>Any existing window number. If unspecified, value is set to 1. Use Disp:Cat? to read the existing window numbers.</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;tnum&gt;</td>
<td>Any existing trace number; if unspecified, value is set to 1. Use Disp:Wind:Cat? to read the trace numbers in an existing window.</td>
</tr>
</tbody>
</table>

Note: This is NOT the trace number of the channel which appears as the Tr annotation on the Trace Status display. This <tnum> is the trace number within the specified window, and is used ONLY for remote programs.

Example Syntax

DISP:WIND:TRAC:Y:AUTO
display:window2:trace2:y:scale:auto

Query Syntax

Not applicable

Default

Not applicable
DISPlay:WINDow:TRACe:Y[:SCALe]:COUPle:METHod <char>

Applicable Models: All

(Read-Write) Sets and returns the method of scale coupling. Learn more about Scale coupling.

Parameters

<char>  OFF - NO scale coupling for any windows.

WINDow - Scale settings are coupled for traces in each window.

ALL - Scale settings are coupled for traces in ALL selected windows.

Enable the selected windows using DISP:WIND:TRAC:Y:COUP ON

Examples

DISP:WIND:TRAC:Y:COUP:METH ALL

Display:window2:trace:y:scale:method window

Query Syntax

DISPlay:WINDow:TRACe:Y[:SCALe]:COUPle:METHod?

Return Type

Character

Default

OFF

DISPlay:WINDow<wnum>:TRACe:Y[:SCALe]:COUPle[:STATe] <bool>

Applicable Models: All

(Read-Write) Enables and disables scale coupling for the specified window. Learn more about Scale coupling.

Parameters

<wnum>  Any existing window number. If unspecified, value is set to 1

Use Disp:Cat? to read the existing window numbers.

<bool>  ON (or 1) - Scale coupling enabled for specified window.

OFF (or 0) - Scale coupling disabled for specified window.

Examples

DISP:WIND:TRAC:Y:COUP ON

Display:window2:trace:y:scale:couple:state off

Query Syntax

DISPlay:WINDow<wnum>:TRACe:Y[:SCALe]:COUPle[:STATe]?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON
DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:PDIVision <num>

Applicable Models: All

(Read-Write) Sets the Y axis Per Division value of the specified trace in the specified window.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

Use Disp:Cat? to read the existing window numbers.

<tnum> Any existing trace number; if unspecified, value is set to 1.

Use Disp:Wind:Cat? to read the trace numbers in an existing window.

Note: This is NOT the trace number of the channel which appears as the Tr annotation on the Trace Status display. This <tnum> is the trace number within the specified window, and is used ONLY for remote programs.

<num> Units / division value. The range of acceptable values is dependent on format and domain.

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

DISP:WIND:TRAC:Y:PDIV 1
display:window2:trace2:y:scale:pdivision maximum

Query Syntax
DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:PDIVision?

Return Type Numeric

Default 10
Applicable Models: All

(Read-Write) Sets the Y axis Reference Level of the specified trace in the specified window.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

Use Disp:Cat? to read the existing window numbers.

<tnum> Any existing trace number; if unspecified, value is set to 1.

Use Disp:Wind:Cat? to read the trace numbers in an existing window.

Note: This is NOT the trace number of the channel which appears as the Tr annotation on the Trace Status display. This <tnum> is the trace number within the specified window, and is used ONLY for remote programs.

<num> Reference level value. The range of acceptable values is dependent on format and domain.

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

DISP:WIND:TRAC:Y:RLEV 0
display:window2:trace2:y:scale:rlevel minimum

Query Syntax

DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:RLEVel?

Return Type

Numeric

Default

Not Applicable

DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:RPOSition <num>
Applicable Models: All

(Read-Write) Sets the **Reference Position** of the specified trace in the specified window.

**Parameters**

- **<wnum>** Any existing window number. If unspecified, value is set to 1.
  
  Use `Disp:Cat?` to read the existing window numbers.

- **<tnum>** Any existing trace number; if unspecified, value is set to 1.
  
  Use `Disp:Wind:Cat?` to read the trace numbers in an existing window.

**Note:** This is NOT the trace number of the channel which appears as the Tr annotation on the Trace Status display. This <tnum> is the trace number within the specified window, and is used ONLY for remote programs.

- **<num>** Reference position on the screen measured in horizontal graticules from the bottom. Choose a value between 0 and 10.

  **Note:** This command will accept MIN or MAX instead of a numeric parameter. See **SCPI Syntax** for more information.

**Examples**

```plaintext
DISP:WIND:TRAC:Y:RPOS 0
display:window2:trace2:y:rposition maximum
```

**Query Syntax**

```plaintext
DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:RPOSition?
```

**Return Type** Numeric

**Default** 5

---

**DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:BOTTom <num>**
**Applicable Models:** All

**(Read-Write)** Sets the minimum scale value for the Log Y-axis.

**Parameters**

<wnum> Any existing window number. If unspecified, value is set to 1.

Use Disp:Cat? to read the existing window numbers.

<tnum> Any existing trace number; if unspecified, value is set to 1.

Use Disp:Wind:Cat? to read the trace numbers in an existing window.

**Note:** This is NOT the trace number of the channel which appears as the Tr annotation on the Trace Status display. This <tnum> is the trace number within the specified window, and is used ONLY for remote programs.

<num> Minimum scale value. Range from 10f to 500P.

**Examples**

```
DISP:WIND:TRAC:Y:BOTT 0
display:window2:trace2:y:bottom 100
```

**Query Syntax**

```
DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:BOTTom?
```

**Return Type** Numeric

**Default** 1

---

**DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:TOP <num>**

**Applicable Models:** All

**(Read-Write)** Sets the maximum scale value for the Log Y-axis.

**Parameters**

<wnum> Any existing window number. If unspecified, value is set to 1.

Use Disp:Cat? to read the existing window numbers.

<tnum> Any existing trace number; if unspecified, value is set to 1.

Use Disp:Wind:Cat? to read the trace numbers in an existing window.

**Note:** This is NOT the trace number of the channel which appears as the Tr annotation on the Trace Status display. This <tnum> is the trace number within the specified window, and is used ONLY for remote programs.

<num> Maximum scale value. Range from 20f to 1E.
Examples

\texttt{DISP:WIND:TRAC:Y:TOP 0}
\texttt{display:window2:trace2:y:top 100}

Query Syntax
\texttt{DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:TOP?}

Return Type
Numeric

Default
1k

\texttt{DISPlay:WINDow<wnum>:TRACe<tnum>:Y:SPACing<char>}

Applicable Models: All

(Read-Write) Sets or returns format type for Y axis graph, either linear or logarithmic.

Parameters

\begin{itemize}
\item \texttt{<wnum>}: Any existing window number. If unspecified, value is set to 1.
\item \texttt{<tnum>}: Any existing trace number; if unspecified, value is set to 1.
\item \texttt{<char>}: choose either \texttt{LINear} or \texttt{LOGarithmic}
\end{itemize}

Examples

\texttt{DISP:WIND:TRACY:SPAC LOG}
\texttt{Display:window2:trace2:y:spacing logarithmic}

Query Syntax
\texttt{DISPlay:WINDow<wnum>:TRACe<tnum>:Y:SPACing?}

Return Type
char

Default
Lin

\texttt{DISPlay:WINDow<wnum>:Y:AUTO}
**Applicable Models:** All

*(Write-only)* Scales **ALL** of the traces to fit in the same window. This is equivalent to "Autoscale All" from the front panel.

Autoscale behaves differently when *scale coupling* is enabled. How it behaves depends on the scale coupling method. [Learn more.](#)

Autoscale is performed only when the command is sent; it does NOT keep the trace autoscaled indefinitely.

See Also, **DISPlay:WINDow:TRACE:y:AUTo** which Autoscales only the specified trace.

**Parameters**

- **<wnum>**  Any existing window number. If unspecified, value is set to 1.

  Use **Disp:Cat?** to read the existing window numbers.

**Examples**

```
DISP:WIND:y:auto
display:window2:y:auto
```

**Query Syntax**

Not applicable

**Default**

Not applicable

---

**DISPlay:WINDow<wnum>:y[:SCALe]:DIFvisions <num>**

**Applicable Models:** All

*(Read-Write)* Sets or returns the number of divisions in all the graphs, for the selected channel

**Parameters**

- **<wnum>**  Any existing window number. If unspecified, value is set to 1

  Use **Disp:Cat?** to read the existing window numbers.

- **<bool>**  ON or 1 - Scale coupling enabled for specified window.

- **<num>**  Number of divisions is between 4 to 30.

  Units / division value. The range of acceptable values is dependent on format and domain.

  **Note:** This command will accept MIN or MAX instead of a numeric parameter. See **SCPI Syntax** for more information.
<table>
<thead>
<tr>
<th>Examples</th>
<th>DISP:WIND:Y:DIV 12</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Display:window2:y:scale:divisions 12</td>
</tr>
<tr>
<td>Query Syntax</td>
<td>DISPlay:WINDow&lt;wnum&gt;:Y[:SCALe]:DIVisions?</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>10</td>
</tr>
</tbody>
</table>
Format Commands

Specifies the way that data will be transferred when moving large amounts of data.

```
FORMat

BORDer [DATA]
```

Click on a keyword to view the command details.

See Also

- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**FORMat:BORDer <char>**

**Applicable Models:** All

(Read-Write) Set the byte order used for GPIB data transfer. Some computers read data from the analyzer in the reverse order. This command is only implemented if FORMAT:DATA is set to :REAL.

If FORMAT:DATA is set to :ASCII, the swapped command is ignored.

**Parameters**

<char> Choose from:

- **NORMal** - Use when your controller is anything other than an IBM compatible computers.
- **SWAPped** - for IBM compatible computers.

**Note:** Use **NORMal** if you are using VEE, LabView, or T&M Tool kit.

**Examples**

```
FORM:BORD SWAP
format:border normal
```
Query Syntax

FORMat:BORDer?

Return Type  Character

Default  Normal

FORMat[:DATA] <char>

Applicable Models: All

(Read-Write) Sets the data format for transferring measurement data and frequency data.

- To transfer measurement data, use CALC:MEAS:DATA.
- To transfer Cal Set data, use SENS:CORR:CSET:DATA
- To transfer Source Power correction data, use:
  - SOURce:POWer:CORRection:COLLect:TABLe:DATA
  - SOURce:POWer:CORRection:COLLect:TABLe:FREQuency
  - SOURce:POWer:CORRection:DATA
- To transfer FIFO buffer data, use SYST:FIFO:DATA?

The following commands transfer frequency data. Use <REAL, 64>

- CALC:MEAS:DATA:SNP?
- CALC:MEAS:X?
- SENS:X?

Use FORMat:BORDer to change the byte order. Use “NORMal” when transferring a binary block from LabView or Vee. For other programming languages, you may need to SWAP the byte order.

Parameters

<char> In the VNA, measurement data is stored as 32 bit and frequencies stored as 64 bit. Therefore, use REAL,32 when getting data and REAL,64 when getting frequencies. That way you are guaranteed to avoid losing any precision as well as getting the maximum speed on the data transfer.

Choose from:
- **REAL,32** - (default value for REAL) Best for transferring large amounts of measurement data. Can cause rounding errors in frequency data.

- **REAL,64** - Slower but has more significant digits than REAL,32. REQUIRED to accurately represent frequency data. See above list for commands which transfer frequency information.

- **ASCii,0** - The easiest to implement, but very slow. Use when you have small amounts of data to transfer.

**Note** The REAL,32 and REAL,64 arguments transfer data in block format as explained in Transferring Measurement Data.

### Examples

```
FORM REAL,64
format:data ascii
```

### Query Syntax

FORMat:DATA?

### Return Type

Character,Character

### Default

ASCii,0

Syst:Preset does NOT reset this command.

However, *RST does reset this command to ASCii,0
Hardcopy Command

Controls printing of the VNA screen and optional data to a printer or a file.

HCOPy:
  DPRinter
  FILE
  [IMMediate]
  ITEM
   | AWINdow
   | CTABLE
   | GPFail
   | LOGO
   | MKRData
   | PNUMber
   | SEGData
   | SWINdow
   | TIME
   | TTABLE
   | WFRaction
   | WINDows
  PAGE
   | DIMension
    | LLEFt
    | URIGht
   | ORlentation
   | SIZE
  SDUMP
   | DATA?
    | FORMat
  PRINters?

Click on a keyword to view the command details.

Blue commands are superseded or obsolete.

See Also

- Learn more about VNA Printing
- Example Programs
HCOPy:DPRinter <string>

Applicable Models: All

(Read-Write) Sets the default printer and selects as the current printer. Use HCOPy:PRINters? to return a list of locally installed printers.

This setting survives instrument preset and VNA application restart.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;string&gt;</td>
<td>Name of the printer to become the default.</td>
</tr>
</tbody>
</table>

Examples

- HCOP:DPR "MyPrinter"
- hcopy:dprinter "YourPrinter"

Query Syntax

HCOPy:DPRinter?

Return Type
String

Default
Not Applicable

HCOPy:FILE <filename>

Applicable Models: All

(Write-only) Saves the screen image to a file. The image does NOT include the optional print data invoked by many HCOPy commands.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;filename&gt;</td>
<td>Name of the file to save the screen to. The file is saved to the current working directory unless a valid full path name is specified. Use one of the following suffixes: .bmp - not recommended due to large file size .jpg - not recommended due to poor quality .png - recommended</td>
</tr>
</tbody>
</table>

Examples

- HCOPY:FILE "myFile.png"
- hcopy:file "c:/data/myfile.png"

Query Syntax

Not Applicable
HCOP[:IMMediate]

Applicable Models: All

(Write-only) Prints the screen to the default printer.

Examples

```
HCOP
hcopy:immediate
```

Query Syntax

Not applicable

Default

Not Applicable

---

HCOPy:ITEM:AWINdow[:STATe] <bool>

Applicable Models: All

(Read-Write) When ON, prints only the Active window. When OFF, prints all windows.

This setting survives instrument preset and VNA application restart.

Parameters

```
<bool> Active window state. Chose from:

    OFF or (0) - Print ALL windows.

    ON or (1) - Print Active window only.
```

Examples

```
HCOP:ITEM:AWIN 1
hcopy:item:awindow:state off
```

Query Syntax

HCOPy:ITEM:AWINdow[:STATe]?

Return Type

Boolean

Default

OFF (0)

---

HCOPy:ITEM:CTABle[:STATe] <bool>
**Applicable Models:** All

(Read-Write) When ON, prints the channel settings table.

This setting survives instrument preset and VNA application restart.

**Parameters**

- **<bool>** Channel table print state. Chose from:
  - **OFF** or (0) - Does NOT print the channel settings table.
  - **ON** or (1) - Prints channel settings table.

**Examples**

- `HCOPy:ITEM:CTAB 1`
- `hcopy:ITEM:ctable:state off`

**Query Syntax**

- `HCOPy:ITEM:CTABle[:STATe]?`

**Return Type**

- Boolean

**Default**

- OFF (0)

---

**HCOPy:ITEM:GPFail[:STATe] <bool>**

**Applicable Models:** All

(Read-Write) When ON, prints the Global Pass/Fail status in the page header.

This setting survives instrument preset and VNA application restart.

**Parameters**

- **<bool>** Pass / Fail print state. Chose from:
  - **OFF** or (0) - Does NOT print Pass / Fail status.
  - **ON** or (1) - Print Pass / Fail status

**Examples**

- `HCOPy:ITEM:GPF 1`
- `hcopy:ITEM:gpfail:state off`

**Query Syntax**

- `HCOPy:ITEM:GPFail[:STATe]?`

**Return Type**

- Boolean

**Default**

- OFF (0)

---

**HCOPy:ITEM:LOGO[:STATe] <bool>**
Applicable Models: All

(Read-Write) When ON, prints the Keysight Technologies logo in the page header.

This setting survives instrument preset and VNA application restart.

Parameters
<bool> Keysight logo print state. Chose from:

OFF or (0) - Prints the Keysight logo.

ON or (1) - Does NOT print the Keysight logo.

Examples
HCOP:ITEM:LOGO 1
hcopy:item:logo:state off

Query Syntax
HCOPy:ITEM:LOGO[:STATe]?

Return Type
Boolean

Default
OFF (0)

---

HCOPy:ITEM:MKRData[:STATe] <bool>

Applicable Models: All

(Read-Write) When ON, includes marker data as part of the trace attributes table.

To print marker data, HCOP:ITEM:TTABLE must also be set to ON.

This setting does not affect the limited marker readout data that can be displayed in the measurement window.

This setting survives instrument preset and VNA application restart.

Parameters
<bool> Marker data print state. Chose from:

OFF or (0) - Does NOT print Marker data.

ON or (1) - Print Marker data.

Examples
HCOP:ITEM:MKRD 1
hcopr:item:mkrd:state off

Query Syntax
HCOPy:ITEM:MKRData[:STATe]?
HCOPy:ITEM:PNUMber[:STATe] <bool>

**Applicable Models:** All

*(Read-Write)* When ON, prints page numbers (1 of n) in the header at the top of each page.

This setting survives instrument preset and VNA application restart.

**Parameters**
- `<bool>` Page number print state. Chose from:
  - **OFF** or (0) - Does NOT print page numbers.
  - **ON** or (1) - Print page numbers.

**Examples**
```
HCOP:ITEM:PNUM 1
hcopy:item:pnumber:state off
```

**Query Syntax**
HCOPy:ITEM:PNUMber[:STATe]?

**Return Type** Boolean
**Default** OFF (0)

---

HCOPy:ITEM:SEGData[:STATe] <bool> - **Obsolete**

**Note:** This command no longer works beginning with A.09.40

*(Read-Write)* When ON, includes ALL segment data as part of the channel settings table.

To print ALL segment data, HCOP:ITEM:CTAB must also be set to ON.

This setting survives instrument preset and VNA application restart.

**Parameters**
- `<bool>` Expanded segment data print state. Chose from:
  - **OFF** or (0) - Does NOT print expanded segment data, but summary data is printed.
  - **ON** or (1) - Print expanded segment data.
**Examples**

HCOP:ITEM:SEGD 1
hcop:item:segdata:state off

**Query Syntax**

HCOPy:ITEM:SEGData[:STATe]?

**Return Type**

Boolean

**Default**

OFF (0)

---

**HCOPy:ITEM:SWINdow[:STATe] <bool>**

**Applicable Models:** All

*(Read-Write)* When ON, prints a single measurement window per page. When OFF, prints up to four measurement windows per page.

This setting survives instrument preset and VNA application restart.

**Parameters**

<bool> Single window print state. Chose from:

- **OFF** or (0) - Print up to four windows per page.
- **ON** or (1) - Print only one window per page.

**Examples**

HCOP:ITEM:SWIN 1
hcop:item:swindow:state off

**Query Syntax**

HCOPy:ITEM:SWINdow[:STATe]?

**Return Type**

Boolean

**Default**

OFF (0)

---

**HCOPy:ITEM:TIME[:STATe] <bool>**
Applicable Models: All

(Read-Write) When ON, prints the VNA computer date and time in the header.

This setting survives instrument preset and VNA application restart.

Parameters

<bool> Time stamp print state. Chose from:

OFF or (0) - Does NOT print time stamp.

ON or (1) - Print time stamp.

Examples

HCOPy:ITEM:TIME 1
hcopy:item:time:state off

Query Syntax

HCOPy:ITEM:TIME:[STATE]?  

Return Type

Boolean

Default

OFF (0)

HCOPy:ITEM:TTABle[:STATE] <bool>

Applicable Models: All

(Read-Write) When ON, prints the trace attributes table.

This setting survives instrument preset and VNA application restart.

Parameters

<bool> Trace attributes table print state. Chose from:

OFF or (0) - Does NOT print the trace attributes table.

ON or (1) - Print the trace attributes table.

Examples

HCOPy:ITEM:TTABle 1
hcopy:item:tttable:state off

Query Syntax

HCOPy:ITEM:TTABle[:STATE]?  

Return Type

Boolean

Default

OFF (0)

HCOPy:ITEM:WFRaction <value>
Applicable Models: All

**Parameters**

<value> Window size as a fraction of the page. Chose a value from .4 (40%) to 1.0 (100%)

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>HCOP:ITEM:WFR .8</td>
</tr>
<tr>
<td>hcopy:ITEM:wfraction .5</td>
</tr>
</tbody>
</table>

**Query Syntax**

HCOP:ITEM:WFRaction?

**Return Type** Numeric

**Default** .4

---

HCOPy:ITEM:WINDows[:STATe] <bool>

Applicable Models: All

(Read-Write) When ON, prints measurement windows.

Use HCOPy:ITEM:AWINDOW to specify all windows or only the active window.

This setting survives instrument preset and VNA application restart.

**Parameters**

<bool> Windows print state. Chose from:

- **OFF** or (0) - Does not print measurement windows.
- **ON** or (1) - Print measurement windows.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>HCOP:ITEM:WIND 1</td>
</tr>
<tr>
<td>hcopy:ITEM:windows:state off</td>
</tr>
</tbody>
</table>

**Query Syntax**

HCOPy:ITEM:WINDows[:STATe]?

**Return Type** Boolean

**Default** OFF (0)

---

HCOPy:PAGE:DIMensions:LLEFt <left, lower>
Applicable Models: All

(Read-Write) Sets the left and lower page margins.

This setting survives instrument preset and VNA application restart.

Parameters

\(<\text{left}>\quad \text{Left page margin as a percentage of entire page width. Value must be between 0 and 1.}\n\)<lower> Lower page margin as a percentage of entire page length. Value must be between 0 and 1.

Examples

\[
\text{HCOP:PAGE:DIM:LLERF} \, 0.10, 0.10
\]
\[
hcopy:page:dimensions:llleft \, 0.5, 0.7
\]

Query Syntax

HCOPy:PAGE:Dimensions:LLFlt?

Return Type

Numeric, Numeric

Default

Depends on selected page size

HCOPy:PAGE:Dimensions:URIGHT <right, upper>

Applicable Models: All

(Read-Write) Sets the right and upper page margins.

This setting survives instrument preset and VNA application restart.

Parameters

\(<\text{right}>\quad \text{Right page margin as a percentage of entire page width. Value must be between 0 and 1.}\n\)<upper> Upper page margin as a percentage of entire page length. Value must be between 0 and 1.

Examples

\[
\text{HCOP:PAGE:DIM:URIG} \, 0.10, 0.10
\]
\[
hcopy:page:dimensions:uright \, 0.5, 0.7
\]

Query Syntax

HCOPy:PAGE:Dimensions:URIGht?

Return Type

Numeric, Numeric

Default

Depends on selected page size

HCOPy:PAGE:ORientation <char>
Applicable Models: All

(Read-Write) Sets the page orientation.

This setting survives instrument preset and VNA application restart.

**Parameters**

- **<char>** Choose from:
  - PORTrait
  - LANDscape

**Examples**

```
HCOP:PAGE:ORI PORT
hcopy:page:orientation landscape
```

**Query Syntax**

HCOPy:PAGE:ORIentation?

**Return Type**

Character

**Default**

PORTrait

---

HCOPy:PAGE:SIZE <int>

Applicable Models:

(Read-Write) Sets the paper type, which implies the page size.

This setting survives instrument preset and VNA application restart.

**Parameters**

- **<int>** Choose from: I
<table>
<thead>
<tr>
<th>Integer</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Letter 8 1/2 x 11 in</td>
</tr>
<tr>
<td>2</td>
<td>Letter Small 8 1/2 x 11 in</td>
</tr>
<tr>
<td>3</td>
<td>Tabloid 11 x 17 in</td>
</tr>
<tr>
<td>4</td>
<td>Ledger 17 x 11 in</td>
</tr>
<tr>
<td>5</td>
<td>Legal 8 1/2 x 14 in</td>
</tr>
<tr>
<td>6</td>
<td>Statement 5 1/2 x 8 1/2 in</td>
</tr>
<tr>
<td>7</td>
<td>Executive 7 1/4 x 10 1/2 in</td>
</tr>
<tr>
<td>8</td>
<td>A3 297 x 420 mm</td>
</tr>
<tr>
<td>9</td>
<td>A4 210 x 297 mm</td>
</tr>
<tr>
<td>10</td>
<td>A4 Small 210 x 297 mm</td>
</tr>
<tr>
<td>11</td>
<td>A5 148 x 210 mm</td>
</tr>
<tr>
<td>12</td>
<td>B4 (JIS) 250 x 354</td>
</tr>
<tr>
<td>13</td>
<td>B5 (JIS) 182 x 257 mm</td>
</tr>
</tbody>
</table>

For more paper type choices, see Microsoft's "wingdi.h" file, which can be downloaded as part of the Platform SDK.

**Examples**

```
HCOP:PAGE:SIZE 2
hcopy:page:size 5
```

**Query Syntax**

HCOPY:PAGE:SIZE?

**Return Type**

Integer

**Default**

1

---

**HCOPY:SDUMP:DATA?**

**Applicable Models:** All

*(Read-only)* Returns the display image in a definite-length arbitrary binary block. The format of the data is PNG by default. Use HCOPY:SDUMP:DATA:FORMat to change the format.

This command is equivalent to saving an image to the VNA (HCOPY:FILE) and then using MMEM:TRAN to transfer the file to the computer.

**Examples**

```
HCOPY:SDUMP?
hcopysdump?
```

**Return Type**

A definite-length arbitrary binary block

**Default**

Not Applicable
HCOPy:SDUMp:DATA:FORMat <char>

**Applicable Models:** All

*(Read-Write)* Sets the graphic format for HCOPy:SDUMp:DATA?

**Parameters**

- **<char>** Choose from: JPG | BMP | PNG

**Examples**

```
HCOP:SDUMp:DATA:FORMat BMP
```

**Query Syntax**

HCOPy:SDUMp:DATA:FORMat?

**Return Type** Character

**Default** PNG

---

HCOPy:PRINters?

**Applicable Models:** All

*(Read-only)* Returns a comma-separated list of printers installed on the VNA. Select a printer using HCOPy:DPRinter.

This setting survives instrument preset and VNA application restart.

**Examples**

```
HCOP:PRIN?
hcopy:printers?
```

**Query Syntax**

HCOPy:PRINters?

**Return Type** String

**Default** Not Applicable
Initiate Commands

Controls triggering signals

```
INITiate
  CONTinuous [IMMediate]
```

Click on a red keyword to view the command details.

See Also

- Example  Triggering the VNA
- Learn about Triggering
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**INITiate:CONTinuous <boolean>**

**Applicable Models:** All

*(Read-Write)* Specifies whether the VNA trigger source is set to Internal (continuous) or Manual.

- For SIMPLE, single-triggering of a single channel, use *Sens:Sweep:Mode SINGle* which sets the number of trigger signals each channel will ACCEPT (Continuous, Groups, *Single*, or HOLD - none.)
- This command is a subset of *TRIG:SEQ:SOURce*, which can also set the trigger source to External.
- See a map of user interface to SCPI triggering commands.
- For more information on triggering, see the VNA Trigger Model.
- See the Example program: Triggering the VNA using SCPI.

**Parameters**

- `<boolean>`
  - **ON** (or 1) - Internal (continuous) trigger.
  - **OFF** (or 0) - Manual sweep. Use *INIT:IMMediate* to send a trigger signal

**Examples**

| INIT:CONT ON |
| initiate:continuous off |

**Query Syntax**

INITiate:CONTinuous?

**Return Type**

Boolean (1 = ON, 0 = OFF)
INITiate<chnum>[::IMMediate]

**Applicable Models:** All

(Write-only) Stops the current sweeps and immediately sends a trigger. (Same as Trigger! on the VNA front panel).

- This command requires Trigger:Source to be set to Manual. This causes ONE trigger signal to be SENT each time INIT:IMM is issued.
- For SIMPLE, single-triggering of a single channel, use Sens:Sweep:Mode SINGLE which sets the number of trigger signals each channel will ACCEPT (Continuous, Groups, Single, or HOLD - none.)

**See the Example program:** Triggering the VNA using SCPI

**Note:** An SMC Fixed Output measurement cannot be triggered using this command. For more information, see the example program.

To trigger ALL channels in turn:

Set ALL channels to Sens<ch>:Sweep:Mode Continuous. The <ch> argument in INIT<ch>:IMM is ignored.

Then…

- TRIG:SCOP ALL triggers ALL channels (in sequence) each time Init:Imm is sent.
- TRIG:SCOP CURRent triggers ONLY the NEXT channel each time Init:Imm is sent.

To trigger ONLY a specified channel:

1. Set ALL channels to Sens<ch>:Sweep:Mode HOLD
2. Send TRIG:SCOP CURRent
3. Send Init<ch>:Imm where <ch> is the channel to be triggered.

**Advanced** Situations that require some channels to be in CONT and others in HOLD are rare. The following describes the behavior of the Init:Imm command in these situations:

**When Trigger:Scope = Global:**

- If the SPECIFIED <chnum> channel is in hold mode, it is put in single trigger (accepts 1 trigger signal) and
goes to the end of the queue of channels to be triggered. The other 'non-hold' channels are triggered. The next Init:Imm triggers the specified channel first.

For example: ch1 is in Hold, ch2 and ch3 are in CONT and we send INIT1:IMM

- On the first INIT:IMM, ch2 and ch3 is triggered.
- next INIT:IMM, ch1, ch2, ch3 is triggered.
- next INIT:IMM, ch2 and ch3 is triggered.
- next INIT:IMM, ch1, ch2, ch3 is triggered, and so forth.

When Trigger:Scope = Channel

- Only ONE channel is triggered for each issued INIT<ch>:IMM command.
- If the specified channel is in hold, it is put in single trigger (accepts 1 trigger signal) and goes the end of the queue of channels to be triggered as in the 'Global' example.

This is one of the VNA overlapped commands. Learn more.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1</td>
</tr>
</tbody>
</table>

**Examples**

- INIT initiate2:immediate

**Query Syntax**

- Not applicable

**Default**

- Not applicable
LXI Command

SCPI Command Tree

LXI:IDENtify[:STATe] <bool>

Applicable Models: N522xB, N523xB, N524xB, E5080A

(Read-Write) Sets and returns the status of the LXI LAN status indicator on the LAN Status dialog.

Parameters

<bool> Choose from:

- **OFF** or **0** - Changes the LXI Status indicator to ‘NORMAL’ and closes the dialog if it was opened by this command.
- **ON** or **1** - Changes the LXI Status indicator to ‘IDENTIFY’ and opens the dialog if it was not already open.

Examples

```
LXI:IDEN 1
lxi:identify:state off
```

Query Syntax

LXI:IDENtify[:STATe]?

Return Type

Boolean

Default

OFF
Memory Commands

The memory commands control saving and loading instrument states and measurement trace data to the hard drive. To read and write trace data in GPIB format, see `CALC:MEAS:DATA`.

<table>
<thead>
<tr>
<th>MME</th>
<th>ory:</th>
</tr>
</thead>
<tbody>
<tr>
<td>CATalog?</td>
<td>CDIrectory</td>
</tr>
<tr>
<td>COPY</td>
<td>DATE?</td>
</tr>
<tr>
<td>DELete</td>
<td>LOAD</td>
</tr>
<tr>
<td></td>
<td>ASCFator</td>
</tr>
<tr>
<td></td>
<td>BSCFator</td>
</tr>
<tr>
<td></td>
<td>CORRection</td>
</tr>
<tr>
<td></td>
<td>CSARchive</td>
</tr>
<tr>
<td></td>
<td>ENR</td>
</tr>
<tr>
<td></td>
<td>[:FILE]</td>
</tr>
<tr>
<td></td>
<td>LIMit</td>
</tr>
<tr>
<td></td>
<td>PLOSs</td>
</tr>
<tr>
<td></td>
<td>PN</td>
</tr>
<tr>
<td></td>
<td>SPURious</td>
</tr>
<tr>
<td></td>
<td>OSSPur</td>
</tr>
<tr>
<td></td>
<td>THReshold</td>
</tr>
<tr>
<td></td>
<td>RLIMit</td>
</tr>
<tr>
<td></td>
<td>SEGMENT</td>
</tr>
<tr>
<td></td>
<td>STATE</td>
</tr>
<tr>
<td>MDIRectory</td>
<td>MOVE</td>
</tr>
<tr>
<td>RDIRectory</td>
<td>STORe</td>
</tr>
<tr>
<td></td>
<td>ASCFator</td>
</tr>
<tr>
<td></td>
<td>BSCFator</td>
</tr>
<tr>
<td></td>
<td>CORRection</td>
</tr>
<tr>
<td></td>
<td>CSARchive</td>
</tr>
<tr>
<td></td>
<td>CSTate</td>
</tr>
<tr>
<td></td>
<td>CITI</td>
</tr>
<tr>
<td></td>
<td>DATA</td>
</tr>
<tr>
<td></td>
<td>FORMat</td>
</tr>
<tr>
<td></td>
<td>CSV:FORMat</td>
</tr>
</tbody>
</table>
Click on a keyword to view the command details.

Blue commands are superseded.

See Also

- Example Programs
- Learn about Save / Recall and File Types
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

Specifying Path Names

The MMEM commands use the following rules to specify path names:

- The default folder is "D:\". Learn more.
- You can change the active directory using MMEMory:CDIRectory.
- Specify only the file name if using the active directory.
- You can also use an absolute path name to specify the folder and file.
**MMEMory:CATalog[:<char>]? [<folder>]**

**Applicable Models:** All

*(Read-only)* Returns a comma-separated string of file names that are in the specified folder. If there are no files of the specified type, "NO CATALOG" is returned. [Learn about File Types.]

**Note:** If Enable Remote Drive Access is unchecked in the Remote Interface dialog, this command will return an error.

**Parameters**

- **<char>** The type of files to list. Choose from:
  - **STATe** - Instrument states (.sta)
  - **CORRection** - Calibration Data (.cal)
  - **CSARchive** - Instrument state and calibration data (.csa)
  - **CSTate** - Instrument state and link to Calibration data (.cst)
  - **[:File]**

  If unspecified then ALL file types (even unknown types) are listed.

- **<folder>** String - Any existing folder name. See [Specifying Path Names]

**Examples**

- `MMEM:CAT?` 'lists all files from the current folder
- `mmemory:catalog:correction? 'D:' ` 'lists .cal files from the specified folder

**Default** Not applicable

---

**MMEMory:CDIRectory <folder>**
Applicable Models: All

(Read-Write) Changes the folder name.

Note: If Enable Remote Drive Access is unchecked in the Remote Interface dialog, this command will return an error.

Parameters

<folder> Any drive and folder name that already exists.

If the same level as the default path, then no punctuation is required.

**MMEM:CDIR Service**

If the new folder is at a different level than the default, use a slash (/) before the folder name and enclose in quotes.

`mmemory:cdirectory '/automation' 'changes default directory up one level.'`

You can use an absolute path to specify the new folder.

`mmemory:cdirectory 'C:/automation/service'`

Query Syntax

MMEMory:CDIRectory? 'Returns the current folder name

Return Type

String

Default See Specifying Path Names

---

MMEMory:COPY <file1>,<file2>

Applicable Models: All

(Write-only) Copies file1 to file2. Extensions must be specified.

Note: If Enable Remote Drive Access is unchecked in the Remote Interface dialog, this command will return an error.

Parameters

<file1> String - Name of the file to be copied. See Specifying Path Names

<file2> String - Name of the file to be created from file1.

Examples

`MMEM:COPY 'MyFile.cst','YourFile.cst'`

Query Syntax

Not applicable

Default Not applicable
MMEMory:DATE? <fileName>

Applicable Models: All

(Read-only) Returns the (year, month, day) that the specified file was last saved.

To query the last date and time a cal set was modified, use CSET DATE? and CSET:TIME?

See Also

MMEM:TIME?

Parameters

<fileName>  String - File name. See Specifying Path Names

Example

MMEM:DATE? "myFile.txt"

>Returns

+2013,+4,+12

mmemory:date? "D:\Calset_18.pcs"

>Returns

+2013,+4,+12

Return Type  Comma-separated integers

Default  Not applicable

---

MMEMory:DELete <file>

Applicable Models: All

(Write-only) Deletes file. Extensions must be specified.

Note: If Enable Remote Drive Access is unchecked in the Remote Interface dialog, this command will return an error.

Parameters

<file>  String - Name of the file to be deleted. See Specifying Path Names

Examples

MMEM:DEL 'MyFile.cst'

Query Syntax  Not applicable

Default  Not applicable
MMEMory:LOAD[:<char>] <file>

Applicable Models: All

(Write-only) Loads the specified file. Learn about File Types

Parameters

<char>  The type of file to load. Choose from:

- ASCFactor
- BSCFactor
- STATe - Instrument states (.sta)
- CORRection - Calibration Data (.cal)
- CSARchive - Instrument state and calibration data (.csa)
- CSTate - Instrument state and link to Calibration data (.cst)
- ENR - Excess Noise Source data (Noise Figure App only)
- SEGment
- [:File]
- PLOSs

When <char> is ENR, then include CAL, - See example below.

*.sNp files CAN be recalled to the VNA although no <char> is used. See example below.

If <char> is unspecified, the extension must be included in the filename.

If an extension is specified in <file> that does not agree with <char> then no action is taken.

=file>  String - Name of the file to be loaded. See Specifying Path Names

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMEM:LOAD 'MyFile.cst'</td>
<td></td>
</tr>
<tr>
<td>mmemory:load:state 'MyInstState'</td>
<td></td>
</tr>
<tr>
<td>MMEM:LOAD:ENR CAL, &quot;D:/data/calset/346C_16500.enr&quot;</td>
<td></td>
</tr>
<tr>
<td>MMEM:LOAD &quot;MyFile.s2p&quot;</td>
<td></td>
</tr>
</tbody>
</table>

Query Syntax  Not applicable

Default  Not applicable
MMEMory:LOAD:LIMIT <file>

Applicable Models: All

(Write-only) Load limit test data of the active trace of the active channel from a CSV file.

Parameters

<file> A file path by string format.

The CSV file shall have header lines and a title row as follows.

"# E5080 Limit Test"

"# Revision: 1.00"

<table>
<thead>
<tr>
<th>TYPE, BEGIN STIMULUS, END STIMULUS, BEGIN RESPONSE, END RESPONSE</th>
</tr>
</thead>
</table>

Examples

MMEM:LOAD:LIM 'MyFile.csv'

Query Syntax

Not Applicable

Default

Not Applicable

---

MMEMory:LOAD:PN:SPURious:OSSPur <file>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Write-only) Load user specified spurious frequency list of the active trace of the active channel.

Parameters

<file> String - Name of the user specified spurious frequency list. See Specifying Path Names

Examples

MMEM:LOAD:PN:SPUR:OSSP 'MyFile.csv'

mmemory:load:pn:spurious:osspur 'D:/MyFile.csv'

Query Syntax

Not applicable

Default

Not applicable

---

MMEMory:LOAD:PN:SPURious:THReshold <file>
Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Write-only) Load the threshold table of the active trace of the active channel.

Parameters

- `<file>` String - Name of the threshold table. See Specifying Path Names

Examples

- `MMEM:LOAD:PN:SPUR:THR 'MyFile.csv'`
- `mmemory:load:pn:spurious:threshold 'D:/MyFile.csv'`

Query Syntax

- Not applicable

Default

- Not applicable

---

**MMEMory:LOAD:RLIMit <file>**

Applicable Models: All

(Write-only) Load ripple limit test data of the active trace of the active channel from a CSV file.

Parameters

- `<file>` A file path by string format.

The CSV file shall have header lines and a title row as follows.

- "# E5080 Ripple Limit Test"
- "# Revision: 1.00"

Examples

- `MMEM:LOAD:RLIM 'MyFile.csv'`

Query Syntax

- Not Applicable

Default

- Not Applicable

---

**MMEMory:MDIRectory <folder>**
Applicable Models: All

(Write-only) Makes a folder.

**Note:** If Enable Remote Drive Access is unchecked in the Remote Interface dialog, this command will return an error.

### Parameters

- `<folder>`: String - Name of the folder to make. See Specifying Path Names

### Examples

- `MMEM:MDIR 'MyFolder`
- `mmemory:mdirectory 'D:/NewFolder'

### Query Syntax

Not applicable

### Default

Not applicable

---

**MMEMory:MOVE <file1>,<file2>**

Applicable Models: All

(Write-only) Renames `<file1>` to `<file2>`. File extensions must be specified.

### Parameters

- `<file1>`: String - Name of the file to be renamed. See Specifying Path Names
- `<file2>`: String - Name of the new file.

### Examples

- `MMEM:MOVE 'MyFile.cst','YourFile.cst'

### Query Syntax

Not applicable

### Default

Not applicable

---

**MMEMory:RDIREctory <folder>**
Applicable Models: All

(Write-only) Removes the specified folder.

**Note:** If Enable Remote Drive Access is unchecked in the Remote Interface dialog, this command will return an error.

**Parameters**

- `<folder>` String - Name of the folder to remove. See Specifying Path Names

**Examples**

```
MMEM:RDIR 'MyFolder'
```

**Query Syntax**

Not applicable

**Default**

Not applicable

---

MMEMory:STORe[:<char>] <file>

Applicable Models: All

(Write-only) Stores the specified file (.sta, .cal, .cst, .csa, .snp, s2px).

Learn about saving SNP files on the VNA.

Learn about saving S2Px files on the VNA.

To save other data files, use MMEM:STOR:DATA.

To save ENR files, use MMEMory:STORE:ENR

**Parameters**

- `<char>` Optional argument. The type of file to store. Choose from:
  - ASCFactor
  - BSCFactor
  - CORRection - Calibration Data (.cal)
  - CSARchive - Instrument state and calibration data (.csa)
  - CSTate - Instrument state and link to Calibration data (.cst)
  - CSV:FORMat
  - ENR
  - [:File]
- **PLOSs**
- **SEGment**
- **STATe** - Instrument states (.sta)
- **STATe:TRACe**
- **TRACe**

No `<char>` is specified for s1p, s2p, s2px and so forth.

Include either `<char>` or the file extension. If both `<char>` and the extension are specified, they must agree or an error is returned and no action is taken. See examples below.

Learn about File Types

<file> String - Name of any valid file that does not already exist. See [Specifying Path Names](#).

**Examples**

<table>
<thead>
<tr>
<th>MMEM:STOR:STAT 'myState'</th>
</tr>
</thead>
<tbody>
<tr>
<td>mmemory:store 'c:/bin/myState.sta'</td>
</tr>
<tr>
<td>MMEM:STOR 'MyData.S2P'</td>
</tr>
</tbody>
</table>

**Query Syntax** Not applicable

**Default** Not applicable

---

**MMEMory:STORe:CITifile:DATA <filename>** - Superseded

**Applicable Models:** All

This command is replaced with **MMEMory:STORe:DATA**.

(Write only) Saves UNFORMATTED trace data to .cti file. Learn more.

**Parameters**

- `<filename>` Any path that already exists with filename.

  If the same level as the default, then no path is required.

<table>
<thead>
<tr>
<th>MMEM:STOR:CIT:DATA 'MYFile.cti'</th>
</tr>
</thead>
<tbody>
<tr>
<td>mmemory:store:citifile:adata &quot;D:\myFile.cti&quot;</td>
</tr>
</tbody>
</table>

**Query Syntax** Not Applicable
**Specifying Path Names**

**MMEMory:STORe:CITifile:FORMat <filename> - Superseded**

**Applicable Models:** All

This command is replaced with **MMEMory:STORe:DATA**.

*(Write only)* Saves FORMATTED trace data to .cti file. [Learn more.](#)

**Parameters**

*<filename>* Any path that already exists with filename.

If the same level, then no path is required

```
MMEM:STOR:CIT:FORM 'MYFile.cti'
```

Of you can specify an absolute path and filename:

```
mmemory:store:citifile:format "D:\myFile.cti"
```

**Query Syntax** Not Applicable

**Default** See **Specifying Path Names**

---

**MMEMory:STORe:DATA <filename>,<type>,<scope>,<format>,<selector>**

**Applicable Models:** All

*(Write-only)* Stores trace data to the following file types: *.prn, *.cti, *.csv, *.mdf. Not all choices are valid with other arguments. See [Valid parameter combinations](#) below.

---

**Notes:**

To save snp files for standard channels (only), use **CALC:MEAS:Data:SNP:PORTs:SAVE**

To save state and calibration files, use **MMEM:STORE**

This command replaces the following:

- **MMEMory:STORe:CITifile:DATA**
- **MMEMory:STORe:CITifile:FORMat**
Parameters

<filename>  (String) Name and extension of the file to which data will be saved. If the extension does not agree with the file type, an error is NOT returned but the data may NOT be what you expect.

See rules for specifying a filename.

<type>  (String) File type to save. Choose from:

"PRN Trace Data" - *.prn data. Learn more.

"Citifile Data Data" - unformatted *.cti data. Learn more.

"Citifile Formatted Data" - formatted *.cti data.

"CSV Formatted Data" - formatted *.csv data. Learn more.

"MDIF Data" - *.mdf data. Learn more.

"GCA Sweep Data" - Gain compression data. Learn more.

"IMD Sweep Data" - Swept IMD data. Learn more.

<scope>  (String) How much data to save. Choose from:

"Trace" - only the specified measurement number is saved.

"Displayed" - all displayed measurements are saved.

"Channel" - all measurements that are in the channel in which the selected measurement reside are saved.

"Auto"

For all Standard Meas Class (S-parameter) channels:

- When correction is OFF, the specified trace is saved.
- When correction is ON, all corrected parameters associated with the calibrated ports in the Cal Set are saved.

For all other channels:
When correction is OFF or ON, the specified trace is saved.

<format> The format in which data is saved. Choose from:

"Displayed" - the format is the same as that in which it is displayed on the VNA screen.

"RI" - Real / Imaginary

"MA" - Magnitude / Angle

"DB" - LogMag / Degrees

<selector> (Integer) Choose from:

-1 Use when <scope> = "Displayed" (does NOT require a selected trace).

<measurement number> Use for all other <scope> selections. Use Calc:Par:MNUM? to read the measurement number of the selected trace.

The following are valid parameter combinations for ALL measurement classes:

<table>
<thead>
<tr>
<th>&lt;type&gt; (String)</th>
<th>&lt;scope&gt; (String)</th>
<th>&lt;format&gt; (String)</th>
<th>&lt;selector&gt; (Numeric)</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;PRN Trace Data&quot;</td>
<td>&quot;Trace&quot;</td>
<td>&quot;Displayed&quot;</td>
<td>Measurement number</td>
</tr>
<tr>
<td>Example: MMEMory:STORe:DATA &quot;myData.prn&quot;,&quot;PRN Trace Data&quot;,&quot;Trace&quot;,&quot;Displayed&quot;,2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;Citifile Data Data&quot;</td>
<td>&quot;Trace&quot; or &quot;Auto&quot; or &quot;Channel&quot;</td>
<td>&quot;RI&quot;</td>
<td>Measurement number</td>
</tr>
<tr>
<td>Example: MMEMory:STORe:DATA &quot;myData.cti&quot;,&quot;Citifile Data Data&quot;,&quot;AUTO&quot;,&quot;RI&quot;,3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;Citifile Formatted Data&quot;</td>
<td>&quot;Trace&quot; or &quot;Auto&quot;</td>
<td>&quot;RI&quot; or &quot;MA&quot; or &quot;DB&quot;</td>
<td>Measurement number</td>
</tr>
<tr>
<td>&quot;Channel&quot;</td>
<td>&quot;RI&quot; or &quot;MA&quot; or &quot;DB&quot; or &quot;Displayed&quot;</td>
<td>Measurement number</td>
<td></td>
</tr>
<tr>
<td>&quot;Displayed&quot;</td>
<td>&quot;RI&quot; or &quot;MA&quot; or &quot;DB&quot; or &quot;Displayed&quot;</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>Example: MMEMory:STORe:DATA &quot;myData.cti&quot;,&quot;Citifile Formatted Data&quot;,&quot;AUTO&quot;,&quot;MA&quot;,3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
"CSV Formatted Data"

"Trace" or "Auto" or "Channel"
"RI" or "MA" or "DB" or "Displayed"
Measurement number

"Displayed"
"RI" or "MA" or "DB" -1

Example: MMEMory:STORe:DATA "myData.csv","CSV Formatted Data","displayed","RI",-1

"MDIF Data"

"Trace" or "Auto" or "Channel"
"RI" or "Displayed" or "Display"
Measurement number

"Displayed"
"RI" or "Displayed" -1

Example: MMEMory:STORe:DATA "myData.mdf","MDIF Data","displayed","displayed",-1

The following parameter combinations save *.csv files in specific formats for GCA and Swept IMD classes:

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;type&gt;</td>
</tr>
<tr>
<td>(String)</td>
</tr>
<tr>
<td>&lt;scope&gt;</td>
</tr>
<tr>
<td>(String)</td>
</tr>
<tr>
<td>&lt;format&gt;</td>
</tr>
<tr>
<td>(String)</td>
</tr>
<tr>
<td>&lt;selector&gt;</td>
</tr>
<tr>
<td>(Numeric)</td>
</tr>
</tbody>
</table>

"GCA Sweep Data"

"Auto"
"DB"
GCA channel number

Example: MMEMory:STORe:DATA "myData","gca sweep data","displayed","displayed",-1

"IMD Sweep Data"

"Auto"
"DB"
Swept IMD channel number

Example: MMEMory:STORe:DATA "myData.mdf","MDIF Data","displayed","displayed",-1

Query Syntax Not applicable

Default Not applicable

MMEMory:STORe:ENR CAL, <file>
Applicable Models: All

(Write-only) Stores an ENR (Excess Noise Source) data. (Noise Figure App only)

To set and read ENR data, use SENS:CORR:ENR:CAL:TABLE:DATA.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;file&gt;</td>
<td>String - Name of any valid file that is not already in existence. See Specifying Path Names</td>
</tr>
</tbody>
</table>

**Examples**

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMMEM:STOR:ENR CAL, &quot;C:/data/calset/346C_16500.enr&quot;</td>
<td>Example of ENR data storage</td>
</tr>
</tbody>
</table>

**Query Syntax**

Not applicable

**Default**

Not applicable

---

**MMEMory:STORe:LIMit <file>**

Applicable Models: All

(Write-only) Saves limit test data of the active trace of the active channel into a CSV file.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;file&gt;</td>
<td>A file path by string format.</td>
</tr>
</tbody>
</table>

**The CSV file shall have header lines and a title row as follows.**

"# VNA Limit Test"

"# Revision: 1.00"

| TYPE,BEGIN STIMULUS,END STIMULUS,BEGIN RESPONSE,END RESPONSE |

**Examples**

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMMEM:STOR:LIM 'MyFile.csv'</td>
<td>Example of limit test data storage</td>
</tr>
</tbody>
</table>

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**MMEMory:STORe:RLIMit <file>**
Applicable Models: All

(Write-only) Saves ripple limit test data of the active trace of the active channel into a CSV file.

Parameters

<file>  A file path by string format.

The CSV file shall have header lines and a title row as follows.

"# VNA Ripple Limit Test"
"# Revision: 1.00"

Examples

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE,BEGIN STIMULUS,END STIMULUS,MAX RIPPLE</td>
<td>MMEM:STOR:RLIM 'MyFile.csv'</td>
</tr>
</tbody>
</table>

Query Syntax

Not Applicable

Default

Not Applicable

---

MMEMory:STORE:SSCReen <file>

Applicable Models: All

(Write-only) Stores the specified file as a bitmap file (.bmp).

Parameters

<file>  String - Name of any valid file that does not already exist. See Specifying Path Names

Examples

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMEM:STOR:SSCR 'myState'</td>
<td></td>
</tr>
<tr>
<td>mmemory:store:sscreen 'c:/bin/myState.bmp'</td>
<td></td>
</tr>
</tbody>
</table>

Query Syntax

Not applicable

Default

Not applicable

---

MMEMory:STORE:TRACe:FORMat:CITifile <char> - Superseded
Applicable Models: All

This command is replaced with **MMEMory:STORe:DATA**.

*(Read-Write)* Specifies the format of subsequent citifile save statements.

**Parameters**

- **<char>** Format in which the citifile will be saved with subsequent **MMEMory:STORe:CIT:FORMat** statements. Choose from:
  - **MA** - Linear Magnitude / degrees
  - **DB** - Log Magnitude / degrees
  - **RI** - Real / Imaginary
  - **AUTO** - Format in which the trace is already displayed. If other than Log Mag, Linear Magnitude, or Real/Imag, then the format will be in Real/Imag.
  - **DISP** - Displayed format.

**Examples**

<table>
<thead>
<tr>
<th>MMEM: STOR: TRAC: FORM: CIT MA</th>
</tr>
</thead>
</table>

**Query Syntax**

**MMEMory:STORe:TRACe:FORMat:CITifile?**

**Return Type**

Character

**Default**

Auto

---

**MMEMory:STORe:TRACe:CONTents:CITifile <char>** - **Superseded**

Applicable Models: All

This command is replaced with **MMEMory:STORe:DATA**.

*(Read-Write)* Specifies the contents of subsequent citifile save statements. *(See Data Define Saves)*

**Parameters**

- **<char>** Choose from:
  - **SING** - Single trace
  - **DISP** - All displayed traces
  - **AUTO** - All displayed traces

**Examples**

<table>
<thead>
<tr>
<th>MMEM: STOR: TRAC: CONT: CIT SING</th>
</tr>
</thead>
</table>

3977
Query Syntax: MMEMory:STORe:TRACe:CONTents?

Return Type: Character

Default: Auto

**MMEMory:STORe:TRACe:FORMat:SNP <char>**

**Applicable Models:** All

(Read-Write) Specifies the format of subsequent .s1p, .s2p, .s3p; s4p save statements. [Learn more.]

To save SNP data, use **CALC:MEAS:DATA:SNP:PORTs:SAVE**

**Parameters**

- `<char>` Choose from:
  - **MA** - Linear Magnitude / degrees
  - **DB** - Log Magnitude / degrees
  - **RI** - Real / Imaginary
  - **AUTO** - data is output in currently selected trace format. If other than LogMag, LinMag, or Real/Imag, then output is in Real/Imag.

**Examples**

- **MMEM:STOR:TRAC:FORM:SNP MA**

**Query Syntax: MMEMory:STORe:TRACe:FORMat:SNP?**

**Return Type:** Character

Default: Auto'

**MMEMory:TIME? <fileName>**
Applicable Models: All

(Read-only) Returns the (hour, minute, second) that the specified file was last saved. The time is returned in local time as setup in the VNA operating system.

To query the last date and time a cal set was modified, use CSET DATE? and CSET:TIME?

See Also

MMEM:DATE?

Parameters
<fileName>  String - File name. See Specifying Path Names

Example

```
MMEM:TIME? "myFile.txt"
'returns
+12,+34,+12

mmemory:time? "D:\Calset_18.pcs"
'returns
+12,+34,+12
```

Return Type  Comma-separated integers

Default  Not applicable

MMEMory:TRANsfer <fileName>,<dataBlock>

Applicable Models: All

(Read-Write) Transfers data between the VNA and an external controller. Other MMEM commands transfer data between the VNA application and the VNA hard drive. If <fileName> already exists, it will be overwritten.

To read trace data from the VNA in block format, use CALC:MEAS:DATA.

Note: If Enable Remote Drive Access is unchecked in the Remote Interface dialog, this command will return an error.

Parameters

<fileName>  String - File name. See Specifying Path Names

<dataBlock>  Block Data - The contents of the file.

The data block is a block of binary data. Use the following syntax:
```markdown
#<num digits><byte count><data bytes><NL><END>

where:

<num digits> specifies how many digits are contained in <byte count>

<byte count> specifies how many data bytes will follow in <data bytes>

Example:

#210ABCDE+WXYZ<nl><end>

Where:

- # - always sent before definite block data
- 2 - specifies that the byte count is two digits (2)
- 10 - specifies the number of data bytes that will follow, not counting <NL><END>

ABCDE+WXYZ - 10 digits of data

<NL><END> - always sent at the end of block data

<table>
<thead>
<tr>
<th>Example</th>
<th>See example program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Syntax</td>
<td>MMEMory:TRANsfer? &lt;fileName&gt;</td>
</tr>
<tr>
<td></td>
<td>Reads block data from the specified file location.</td>
</tr>
</tbody>
</table>

Default | Not applicable |
```
Output Commands

Controls two output functions: RF power and Noise Source.

```
OUTPut:
  | MANual:NOISE[:STATe]
  | [:STATe]
```

Click on a red keyword to view the command details.

See Also

- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

OUTPut:MANual:NOISE[:STATe] <bool> "usbNsSrcID"

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets and reads the noise source (28V) ON or OFF.

**Note:** “usbNsSrcID” is an optional parameter. If this optional parameter is specified, it turns the noise source on/off for the specified devices.

**Parameters**

- `<bool>`
  - **ON (1)** - Noise source ON
  - **OFF (0)** - Noise source OFF

**Examples**

- `OUTP:MAN:NOIS 0`
- `output:manual:noise:state 1`

**Query Syntax**

- `OUTPut:MANual:NOIS[e][:STATe]`?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

For VNA models with a Noise Figure option (028/029/H29), the 28V line is always ON. The ON/OFF state is also available from a VNA softkey menu.

For VNA models WITHOUT a Noise Figure option (028/029/H29), the 28V line is OFF by default and survives a preset. The ON/OFF state is NOT available from a VNA softkey menu.
OUTPut[:STATe] <ON | OFF>

Applicable Models: All

(Read-Write) Turns RF power from the source ON or OFF.

See note about source power state with instrument state save and recall.

Parameters

<ON | OFF>

ON (or 1) - turns RF power ON

OFF (or 0) - turns RF power OFF

Examples

OUTP ON
output:state off

Query Syntax

OUTPut[:STATe]?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON
**Route Command**

Learn about Frequency Offset

SCPI Command Tree

**ROUTe<cnum>:PATH:LOOP[:R1] <char>**

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Throws internal switch to reference receiver when the specified channel is measured.

N523xA models do NOT have the R1 reference receiver switch.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<char>`: Position of the switch. Choose from:
  - **INTernal**: bypass R1 Loop. Connects the port 1 source directly to the R1 receiver.
  - **EXTernal**: flow through R1 Loop. Allows direct access to the R1 receiver through the Reference 1 front-panel connectors.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROUT:PATH:LOOP INT</td>
<td>INTernal setting</td>
</tr>
<tr>
<td>route2:path:loop:r1 external</td>
<td>EXTernal setting</td>
</tr>
</tbody>
</table>

**Query Syntax**

ROUTe<cnum>:PATH:LOOP:R1?

**Return Type**

Character

**Default**

INTernal
Controls the Active Hot Parameters configuration.

```plaintext
SENSe:ACTive:
  DISP:
    | INTerpolate[:STATe]
    | TRACe
    | POWer
  SWEep:
    | PHASe
    | POINT
    | POWer
    | START
    | STEP
    | STOP
    | TYPE
  TTONe:
    | MODE
    | ABSolute
    | RELative
```

Click on a keyword to view the command details.

See Also

- Synchronizing the Analyzer and Controller
- SCPI Command Tree
- Programming Example

SENSe<ch>:ACTive:DISP:INTerpolate[:STATe] <bool>
Applicable Models: N524xB

(Read-Write) Sets whether or not interpolation is on for display. Frequency, power, and phase X axis are supported for display. Interpolation may be applied in the trace.

Parameters

- `<ch>` Any existing active match channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **ON** or (1) Interpolate the results
  - **OFF** or (0) Do NOT interpolate the results.

Examples

```
SENS:ACTive:DISP:INT 1
sense:active:disp:interpolation off
```

Query Syntax

```
SENSe<ch>:ACTive:DISP:INTERpolation[:STATe]?
```

Return Type

Boolean

Default

OFF

Applicable Models: N524xB

(Read-Write) Set and read a fixed trace input power level.

Parameters

- `<ch>` Any existing active match channel. If unspecified, value is set to 1.
- `<tnum>` The number of the trace to set input power. If unspecified, the trace number is set to 1.
- `<num>` Input power level. The range is -10 to 0. The units are Hz, dBm, or s.

Examples

```
SENS:ACTive:DISPlay:TRACe1:IPWer 0 dBm
sense:active:display:trace1:ipwer 0 dBm
```

Query Syntax

```
SENSe<ch>:ACTive:DISPlay:TRACe<tnum>:IPWer <num>
```

Return Type

Numeric

Default

Not Applicable
Applicable Models: N524xB

(Read-Write) Set and read the number of phase points. For the tuning tone at the output, a phase sweep is done for each point.

Parameters

- `<ch>` Any existing active match channel. If unspecified, value is set to 1.
- `<num>` Phase points. Do not exceed the max number of phase points (50).

Examples

- `SENS:ACTive:SWEep:PHAS:POIN 201`
- `sense:active:sweep:phase:point 101`

Query Syntax

- `SENS<ch>:ACTive:SWEep:PHASe:POINt?`

Return Type

- Numeric

Default

- 8

---

SENSe<ch>:ACTive:SWEep:POWeR:STARt <num>

Applicable Models: N524xB

(Read-Write) Set and read the start power level for a 3D sweep.

Parameters

- `<ch>` Any existing active match channel. If unspecified, value is set to 1.
- `<num>` Start power level in dBm. Choose a value from the min power to max power of the hardware.

Examples

- `SENS:ACTive:SWEep:POW:STAR 0`
- `sense:active:sweep:power:start 0`

Query Syntax

- `SENS<ch>:ACTive:SWEep:POWeR:STARt?`

Return Type

- Numeric

Default

- -10 dBm

---

SENSe<ch>:ACTive:SWEep:POWeR:STEP <num>
(Read-Write) Set and read the number of power steps for a 3D sweep.

Parameters

- `<ch>` Any existing active match channel. If unspecified, value is set to 1.
- `<num>` Number of power steps. The range is 2 to 20001.

Examples

```
SENS:ACTive:SWE:POW:STEP 201
sense:active:sweep:power:step 201
```

Query Syntax

SENS<ch>:ACTive:SWEep:POWer:STEP?

Return Type

Numeric

Default

201

SENSe<ch>:ACTive:SWEep:POWer:STOP <num>

(Read-Write) Set and read the stop power level for a 3D sweep.

Parameters

- `<ch>` Any existing active match channel. If unspecified, value is set to 1.
- `<num>` Stop power level in dBm. Choose a value from min power to max power of the hardware.

Examples

```
SENS:ACTive:SWE:POW:STOP 0
sense:active:sweep:power:stop 0
```

Query Syntax

SENS<ch>:ACTive:SWEep:POWer:STOP?

Return Type

Numeric

Default

0 dBm
**Applicable Models:** N524xB

**(Read-Write)** Set and read the sweep type.

**Parameters**

- `<ch>` Any existing active match channel. If unspecified, value is set to 1.
- `<char>` Sweep type. Choose from:
  - **LIN** - frequency linear sweep
  - **LOG** - frequency log sweep
  - **POW** - power sweep
  - **MULT** - Multiple sweep. Sweeps frequency with fixed power and has several sweeps with different power values. This is called a 3D sweep: frequency, power, and phase.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:ACTive:SWE:TYPE LIN</td>
<td>Sense:active:sweep:type lin</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe<ch>:ACTive:SWEep:TYPE?

**Return Type**

Character

**Default**

LIN

---

**SENSe<ch>:ACTive:TTONe:MODE <char>**

**Applicable Models:** N524xB

**(Read-Write)** Set and read the tuning tone mode. The tuning tone is the source at the output port to extract the X parameters.

**Parameters**

- `<ch>` Any existing active match channel. If unspecified, value is set to 1.
- `<char>` Tuning tone mode. Choose from:
  - **ABSolute** - tone power is an absolute power
  - **RELative** - tone power is a dBc power relative to the input power

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:ACTive:TTON:MODE ABS</td>
<td>Sense:active:ttone:mode abs</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe<ch>:ACTive:TTONe:MODE?

**Return Type**

Character

**Default**

ABSolute
SENSe<ch>:ACTive:TTONe:ABSolute <num>

Applicable Models: N524xB

(Read-Write) Set and read the absolute tone power level.

Parameters

- **<ch>** Any existing active match channel. If unspecified, value is set to 1.
- **<num>** Absolute tone power level.

Examples

```
SENS:ACTive:TTON:ABS -5
sense:active:ttone:absolute -5
```

Query Syntax

SENSe<ch>:ACTive:TTONe:ABSolute?

Return Type

Numeric

Default

-5 dBm

SENSe<ch>:ACTive:TTONe:RELative <num>

Applicable Models: N524xB

(Read-Write) Set and read the tone power relative to the input power (dBc).

Parameters

- **<ch>** Any existing active match channel. If unspecified, value is set to 1.
- **<num>** Relative tone power level.

Examples

```
SENS:ACTive:TTON:REL 10
sense:active:ttone:relative 10
```

Query Syntax

SENSe<ch>:ACTive:TTONe:RELative?

Return Type

Numeric

Default

-15 dBc
Sense Amplifier

When you use the M9485A, you can control the M9379A amplifier through the VNA firmware. The following commands are available when the launcher includes the M9379A.

```
SENSe:AMPLifier:M9379
  | COUNt?
  | MODule
    | :ATTenuation
    | :CHASsis
    | :CONTrol[:STATe]
    | :PATH
    | :POWer[:STATe]
    | :SLOT
    | :SWITch:PATH
```

Click on a keyword to view the command details.

---

**SENSe<cnum>:AMPLifier:M9379:COUNt?**

**Applicable Models:** M9485A

*(Read-only)* Returns the total number of M9379A amplifier modules that are connected to the VNA firmware.

**Parameters**

<table>
<thead>
<tr>
<th>&lt;cnum&gt;</th>
<th>Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.</th>
</tr>
</thead>
</table>

**Examples**

```
SENSe:AMPLifier:M9379:COUNt?
sense2:amplifier:m9379:count
```

**Return Type** Numeric

**Default** Not applicable

---

3990
SENSe<cnum>:AMPlifier:M9379:MODule<mod>:ATTenuation <att>

Applicable Models: M9485A

(Read-Write) Sets and reads the attenuation of the M9379A amplifier 1.

Parameters
- <cnum>: Any existing channel number; if unspecified, value is set to 1.
- <mod>: Module number of M9379A. The number starts from 1 for the leftmost module of M9379A.
- <att>: Attenuation in dB from 0 to 28 with 2 step

Examples
SENS:AMPL:M9379:MOD1:ATT 10
sense2:amplifier:m9379:module2:attenuation 5

Query Syntax
SENSe<cnum>:AMPlifier:M9379:MODule<mod>:ATTenuation?

Return Type
Numeric
Default
28

SENSe<cnum>:AMPlifier:M9379:MODule<mod>:CHASsis?

Applicable Models: M9485A

(Read Only) Returns the chassis number where the specified M9379A module is located.

Parameters
- <cnum>: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- <mod>: Module number of M9379A. The number starts from 1 for the leftmost module of M9379A.

Examples
SENS:AMPL:M9379:MOD1:CHAS?
sense2:amplifier:m9379:module2:chassis?

Return Type
Numeric
Default
Not applicable

SENSe<cnum>:AMPlifier:M9379:MODule<mod>:CONTrol[:STATe] <bool>
Applicable Models: M9485A

(Read-Write) Sets and reads the status of M9379A control.

Parameters

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<mod>`: Module number of M9379A. The number starts from 1 for the leftmost module of M9379A.
- `<bool>`: Module control state. Choose from:
  - 0 or OFF - Skips to control the M9379A at the specified channel.
  - 1 or ON - Enables to control the M9379A at the specified channel.

Examples

```
SENS:AMPL:M9379:MOD1:CONT ON
sense2:amplifier:m9379:module2:control 0
```

Query Syntax

```
SENSe<cnum>:AMPLifier:M9379:MODule<mod>:CONTrol[:STATe]?
```

Return Type: Boolean

Default: 1 or ON

---

Applicable Models: M9485A

(Read-Write) Sets and reads the path for the M9379A amplifier 1.

Parameters

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<mod>`: Module number of M9379A. The number starts from 1 for the leftmost module of M9379A.
- `<char>`: Path. Choose from:
  - THRU - Through.
  - AMPLifier - amplifier 1.
  - NFReceiver - NF receiver switch (NF measurement only)

Examples

```
SENS:AMPL:M9379:MOD1:PATH THRU
sense2:amplifier:m9379:module2:path amplifier
```
Query Syntax  
SENSe<cnum>:AMPLifier:M9379:MODule<mod>:PATH?

Return Type  
Char

Default  
THRU

SENSe<cnum>:AMPLifier:M9379:MODule<mod>:POWer[:STATe] <bool>

Applicable Models: M9485A

(Read-Write) Sets and reads the status of M9379A power.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

<mod> Module number of M9379A. The number starts from 1 for the leftmost module of M9379A.

<bool> power control state. Choose from:

0 or OFF - Power off

1 or ON - Power on

Examples

SENSe:AMPL:M9379:MOD1:POW ON
sense2:amplifier:m9379:module2:power 0

SENSe<cnum>:AMPLifier:M9379:MODule<mod>:SLOT?

Query Syntax  
SENSe<cnum>:AMPLifier:M9379:MODule<mod>:POWer[:STATe]?

Return Type  
Boolean

Default  
0 or OFF
Applicable Models: M9485A

(Read Only) Reads the slot number where the M9379A is located.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9379A. The number starts from 1 for the leftmost module of M9379A.

**Examples**

```
SENS:AMPL:M9379:MOD1:SLOT?
sense2:amplifier:m9379:module2:slot?
```

**Return Type** Numeric

**Default** Not applicable

---

SENSe<cnum>:AMPlifier:M9379:MODule<mod>:SWITch:PATH <char>

Applicable Models: M9485A

(Read-Write) Sets and reads the path for the M9379A switch

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<mod>`: Module number of M9379A. The number starts from 1 for the leftmost module of M9379A.
- `<char>`: Path. Choose from:
  - **A** - Path A
  - **B** - Path B
  - **NFSource** - NF source switch (NF measurement only)
  - **NFLO** - NF LO switch (Option 720 only)
  - **NFR Receiver** - NF receiver switch (NF measurement only)

**Examples**

```
SENS:AMPL:M9379:MOD1:SWIT:PATH A
sense2:amplifier:m9379:module2:switch:path b
```

**Query Syntax** 

SENSe<cnum>:AMPlifier:M9379:MODule<mod>:SWITch:PATH?
<table>
<thead>
<tr>
<th>Return Type</th>
<th>&lt;char&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>A</td>
</tr>
</tbody>
</table>
Sense: Average Commands

Sets sweep-to-sweep averaging parameters. Averaging is a noise reduction technique that averages each data point over a user-specified number of sweeps. Averaging affects all of the measurements in the channel.

SENSe:AVERage

| CLEar
| COUNt
| MODE
| [STATe]

Click on a keyword to view the command details.

See Also

- Example using some of these commands.
- Learn about Averaging
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

SENSe<cnum>:AVERage:CLEar

Applicable Models: All

(Write-only) Clears and restarts averaging of the measurement data. Does NOT apply to point averaging.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.

Examples

- SENS:AVER:CLE
- sense2:average:clear

Query Syntax

- Not applicable

Default

- Not applicable

SENSe<cnum>:AVERage:COUNt <num>
Applicable Models: All

(Read-Write) Sets the number of measurements to combine for an average. Must also set SENS:AVER[:STATe] ON

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<num>` Number of measurements to average. Choose any number between 1 and \(2^{16}\).

**Examples**

- SENS:AVER:COUN 999
- sense2:average:count 73

**Query Syntax**

SENSe<cnum>:AVERage:COUNt?

**Return Type**

Numeric

**Default**

1

---

SENSe<cnum>:AVERage:MODE <char>

Applicable Models: All

(Read-Write) Sets the type of averaging to perform: Point or Sweep.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<num>` Averaging Type. Choose from:
  
  **POINt** - Averaging measurements are made on each data point before stepping to the next data point.
  
  **SWEEP** - Averaging measurements are made on subsequent sweeps until the required number of averaging sweeps are performed.

**Examples**

- SENS:AVER:MODE POIN
- sense2:average:mode sweep

**Query Syntax**

SENSe<cnum>:AVERage:MODE?

**Return Type**

Character

**Default**

Sweep

---

SENSe<cnum>:AVERage[:STATe] <ON | OFF>
Applicable Models: All

(Read-Write) Turns trace averaging ON or OFF.

Parameters

<cnm> Any existing channel number; if unspecified, value is set to 1.

<ON | OFF> ON (or 1) - turns averaging ON.
OFF (or 0) - turns averaging OFF.

Examples

SENS:AVER ON
sense2:average:state off

Query Syntax SENSE<cnm>:AVERage[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default Off
SENSe:BANDwidth | BWIDth Commands

SENSe:BANDwidth | BWIDth:

  RESolution <num>
  TRACk:FORCe
  TRACk[:STATE] <bool>

See Also

- Example Programs
- Learn about IF Bandwidth
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

SENSe<cnum>:BANDwidth | BWIDth[:RESolution] <num>

Applicable Models: All

(Read-Write) Sets the bandwidth of the digital IF filter to be used in the measurement. (Use either Sense:Bandwidth or Sense:Bwidth)

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the VNA model. (Click to see the lists.) If an invalid number is specified, the analyzer will round up to the closest valid number.

This parameter supports MIN and MAX as arguments. Learn more.

Examples

- `SENS:BWID 1KHZ`
- `sense2:bandwidth:resolution 1000`

Query Syntax

SENSe<cnum>:BANDwidth | BWIDth[:RESolution]?

Return Type

Numeric

Default

Varies with VNA model.

SENSe<cnum>:BANDwidth | BWIDth:TRACk:FORCe <bool>
Applicable Models: N522xB, N5234B, N5235B, N524xB, E5080

(Read-Write) Enables/disables the Reduce IF BW at Low Frequencies feature in segments with IFBW arbitrary.

(Use either Sense:Bandwidth:Track:Force or Sense:Bwidth:Track:Force).

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<bool>`: Choose from:
  - **ON** or **1** - Enable reduce IF BW at Low Frequencies in segments with IFBW arbitrary.
  - **OFF** or **0** - Disable reduce IF BW at Low Frequencies in segments with IFBW arbitrary.

Examples

- SENS:BWID:TRAC:FORC OFF
-sense2:bandwidth:track:force 1

Query Syntax

SENSe<cnum>:BANDwidth | BWIDth:TRACk[:STATe] <bool>

Return Type

Boolean

Default

OFF

SENSe<cnum>:BANDwidth | BWIDth:TRACk[:STATe] <bool>

Applicable Models: N522xB, N5234B, N5235B, N524xB, E5080

(Read-Write) Sets and returns the state of the Reduce IF BW at Low Frequencies feature.

(Use either Sense:Bandwidth:Track or Sense:Bwidth:Track).

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<bool>`: Choose from:
  - **ON** or **1** - Reduce IF BW at Low Frequencies is set ON
  - **OFF** or **0** - Reduce IF BW at Low Frequencies is set OFF

Examples

- SENS:BWID:TRAC OFF
- sense2:bandwidth:track 1

Query Syntax

SENSe<cnum>:BANDwidth | BWIDth:TRACk[:STATe]?
<table>
<thead>
<tr>
<th>Return Type</th>
<th>Boolean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>OFF</td>
</tr>
</tbody>
</table>
SENSe<cnm>:CLASs:NAME?

Applicable Models: All

(Read-only) Returns the measurement class name of the specified channel. Use CALCulate:MEASure:DEFine and CALCulate:MEASure:PARameter commands to create measurements.

Parameters

<cnm>  Any existing channel number; if unspecified, value is set to 1.

Examples

SENSe:CLASs:NAME?
sense2:class:name?

For a standard S-Parameter channel, returns...
"Standard"

Default  Not applicable
When you use E5080B, you can control the Interface control through the VNA firmware.

SENSe:CONTRol:
  | [:STATE]
  | :DWELI
  | :HANDler
    | [:DATA]
    | [:STATE]
  | :DIO
    | [:STATE]
    | :VIO
    | :LEVEL
    | :IMMediate
    | :IOTYpe
  | :PIO
    | :TYPE
    | :LEVEL
  | :RFFE
    | :CLOCk
    | :CSEQUence
      | :SADDress
      | :TYPE
    | :BCOunt
    | :ADDRess
    | [:WRITe]:DATA
    | :READ:DATA
    | :COUNt
SENSe<cnum>:CONTrol[:STATe] <bool>

Applicable Models: E5080B

(Read-Write) Sets and read the state of interface control for all channels. Channel number is ignored.

Parameters
- <cnum> Channel number.
- <bool> Module control state. Choose from:
  - O or OFF - Interface control port signals won't be sent.
  - 1 or ON - Interface control port signals will be sent.

Examples

Query Syntax :SENSe<cnum>:CONTrol[:STATe]?
Return Type Boolean
Default OFF or 0
Preset 0
Save or Recall Yes

SENSe<cnum>:CONTrol:DWELl <char>, <num>
Applicable Models: E5080B

(Read-Write) Sets and read the delay time between the time all interface control port signals and all commands sent and the one measurements start. Set independently per channel and for forward and reverse sweep. Not set per IO type.

**Parameters**

- `<cnum>` Channel number.
- `<char>` Character - when to send remote commands. Choose from:
  
  **AFTER** - After the channel sweep ends.
  
  **BEFORE** - Before the channel sweep starts.
- `<num>` Wait time in milliseconds. Any positive integer is allowed.

**Examples**

SENS:CONT:DWEL BEF 10
sense2:control:dwell?

**Query Syntax**

:SENSe<cnum>:CONTrol:DWELl? <char>

**Return Type** Character

**Default** 0

**Preset** 0

**Save or Recall** Yes

SENSe<cnum>:CONTrol:HANDler:<grp>[::DATA] <char>, <num>

Applicable Models: E5080B

(Read-Write) Sends values to the respective Handler I/O port (A-D). Although ports C and D are normally bidirectional, ONLY Output mode is allowed using the Interface Control feature. It cannot read from these, or any other ports.

**Parameters**

- `<cnum>` Channel number.
- `<grp>` Port identifier to set bits for. Choose from:
  
  A , B , C and D .
<char> Character - when to send remote commands. Choose from:

**AFTER** - After the channel sweep ends.

**BEFORE** - Before the channel sweep starts.

<num> The number of the data bits to set. The range of the value is determined as follows.

- Port A: 0 - 255
- Port B: 0 - 255
- Port C: 0 - 15
- Port D: 0 - 15

### Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

### Query Syntax

:SENSecnum>:CONTr:HANDler<grp>[:DATA]? <char>

### Return Type

Numeric

- **Default**: 0
- **Preset**: 0
- **Save or Recall**: Yes

---

**SENSe<cnum>:CONTr:HANDler<STATE> <char>, <bool>**

**Applicable Models**: E5080B

*(Read-Write)* Set and read the control function state for each channel. If ON, Handler I/O port signals will be sent before the beginning of the sweep or after the end of the sweep. If OFF, these signals will not be changed.

### Parameters

- **<cnum>** Channel number.
- **<char>** Character - when to send remote commands. Choose from:
  - **AFTER** - After the channel sweep ends.
  - **BEFORE** - Before the channel sweep starts.
<bool> Choose from:

ON (1) - Handler I/O port signals will be sent.

OFF (0) - Handler I/O port signals won't be sent.

Examples

```
SENS:CONT:HAND AFT, ON
sense2:control:handler? after
```

Query Syntax

```
:SENSe<cnum>:CONTrol:HANDler[:STATe]? <char>
```

Return Type

Boolean

Default 1

Preset 1

Save or Recall Yes

SENS<cnum>:CONTrol:DIO<id>[:STATe] <char>, <bool>

Applicable Models: E5080B

(Read-Write) Set and read the control function state for each channel. If ON, DUT control signals will be sent before the beginning of the sweep or after the end of the sweep. If OFF, then DUT signals will not be changed.

Parameters

<cnm> Channel number.

<id> DIO number, 1 or 2

<char> Character - when to send remote commands. Choose from:

AFTER - After the channel sweep ends.

BEFORE - Before the channel sweep starts.

<bool> Function enable or disable control.

Choose from: ON or OFF

Examples

```
SENS:CONT:DI01 AFT, ON
sense2:control:di01? after
```

Query Syntax

```
:SENSe<cnum>:CONTrol:DIO<id>[:STATe]? <char>
```

Return Type

Boolean (1= ON, 0= OFF)

Default OFF
SENSe<cnum>:CONTrol:DIO<id>:VIO[:STATe] <char>, <bool>

Applicable Models: E5080B

(Read-Write) Set and read the VIO function state for each channel. If ON, then VIO's voltage is set to the value which is determined by SENSe:CONTrol:DIO:LEVel before the beginning of the sweep or after the end of the sweep. If OFF, the VIO's voltage is disabled.

Parameters
<cnun> Channel number.
{id> DIO number, 1 or 2
<char> Character - when to send remote commands. Choose from:

**AFTer** - After the channel sweep ends.

**BEFore** - Before the channel sweep starts.

<bool> Function enable or disable control.

Choose from: **ON** or **OFF**

Examples

```
SENSe:CONT:DIO1:VIO AFT, OFF
sense2:control:dio1:vio? after
```

Query Syntax

:SENSe<cnum>:CONTrol:DIO<id>:VIO[:STATe]? <char>

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON

SENSe<cnum>:CONT:DI<id>:LEVel <char>, <num>

Applicable Models: E5080B

(Read-Write) Specifies IO level of the DUT Control DIO1 or DIO2's 8-bit IO. The value of AFTer is overwritten by the one of BEFore and vice versa due to set the same value to both AFTer and BEFore.

Parameters

<cnun> Channel number.

{id> DIO number, 1 or 2
<char> Character - when to send remote commands. Choose from:

**AFTer** - After the channel sweep ends.

**BEFore** - Before the channel sweep starts.

4008
IO level in volt. Value range, 0.9 to 3.5, step 0.05.

Examples

SENS:CONT:DIO1:VIO AFT, OFF
sense2:control:dio1:vio? after

Query Syntax

:SENS<cnm>:CONTrol:DIO<id>:LEVel? <char>

Return Type

Numeric

Default

1.2

Note: The default value definition comes from MIPI RFFE standard. Referring to the VIO Supply Pin Requirements, 1.2V is the minimum typical voltage value of the definition.

SENSe<cnm>:CONTrol:DIO<id>:IMMediate <char>

Applicable Models: E5080B

(Write only) Specifies parameter set of DUT Control function for each channel. And, fetch E5080B Hardware status values (PIO input state, RFFE read command results) and stores the values in Firmware variables. If executed, the DUT signals will be sent immediately. It doesn't matter if Enable DUT Control is ON or OFF.

Parameters

<cnm> Channel number.
<br> <id> DIO number, 1 or 2
<br> <char> Character - when to send remote commands. Choose from:
<br> AFTer - After the channel sweep ends.
<br> BEF ore - Before the channel sweep starts.

Examples

SENS:CONT:DIO1:IMM BEF

Query Syntax

Not applicable

Default

Not applicable

SENSe<cnm>:CONTrol:DIO<id>:IOTYpe<iogroup> <char>, <enum>
Applicable Models: E5080B

(Read-Write) Specifies IO function type of the 8-bit IO pin, for each IO group. IO group1 is IO pin 1 and 2, group2 is pin 3 and 4, group3 is pin 5 and 6, group4 is pin 7 and 8.

Parameters

- `<cnum>` Channel number.
- `<id>` DIO number, 1 or 2
- `<iogroup>` IO group number. Value range, 1 to 4.
- `<char>` Character - when to send remote commands. Choose from:
  - `AFTER` - After the channel sweep ends.
  - `BEFORE` - Before the channel sweep starts.
- `<enum>` Set the IO function for the IO group. Choose from: `PARallel` or `RFFE`

Examples

```
SENS:CONT:DIO1:IOTY AFT, RFFE
sense2:control:dio1:ioty1? after
```

Query Syntax

`:SENSe<cnum>:CONTrol:DIO<id>:PIO<iopin>:TYPE <char>, <enum>`

Return Type

Character

Default

`PARallel`

SENS<cnue>:CONT<id>:PIO<iopin>:TYPE <char>, <enum>

Applicable Models: E5080B

(Read-Write) Set or read the signal direction type of Parallel IO, for each IO pin.. This setting is valid when the IO pin function is selected as parallel IO.

Parameters

- `<cnum>` Channel number.
- `<id>` DIO number, 1 or 2
- `<iopin>` IO pin number.
- `<char>` Character - when to send remote commands. Choose from:
  - `AFTER` - After the channel sweep ends.
  - `BEFORE` - Before the channel sweep starts.
- `<enum>` IO direction. Choose from: `IN` or `OUT`
Examples

SENS:CONT:DIO1:PIO:TYPE AFT, IN
sense2:control:dio1:pio2:type? after

Query Syntax

Return Type
Character

Default
OUT

SENSe<cnum>:CONTrol:DIO<id>:PIO<iopin>:LEVel <char>, <enum>

Applicable Models: E5080B

(Read-Write) Set or read the signal level of IO pin, high or low. This setting is valid when the IO pin function is selected as parallel IO. If the IO type is IN, this command shall be a read-only command. Write command will cause error.

Parameters

<cnum> Channel number.
<id> DIO number, 1 or 2
<iopin> IO pin number.
<char> Character - when to send remote commands. Choose from:

AFTER - After the channel sweep ends.

BEFORE - Before the channel sweep starts.

<enum> Signal level. Choose from: HIGH or LOW

Examples

SENS:CONT:DIO1:PIO:LEV AFT, HIGH
sense2:control:dio1:pio7:level? after

Query Syntax
:SENSe<cnum>:CONTrol:DIO<id>:PIO<iopin>:LEVel? <char>

Return Type
Character

Default
LOW

SENSe<cnum>:CONTrol:DIO<id>:RFFE:CLOCk <char>, <num>
**Applicable Models:** E5080B

*(Read-Write)* Set or read the RFFE clock rate.

**Parameters**

- `<cnum>` Channel number.
- `<id>` DIO number, 1 or 2
- `<char>` Character - when to send remote commands. Choose from:
  - **AFTer** - After the channel sweep ends.
  - **BEFore** - Before the channel sweep starts.
- `<num>` Clock rate in Hz. Value range, 25kHz to 25000kHz. Possible values are \((50000/n) \text{ kHz}\), with integer \(n\), 2000 to 2.

**Examples**

```
SENS:CONT:DIO1:RFFE:CLOC AFT, 25000
sense2:control:dio1:rffe:clock? after
```

**Query Syntax**

```
:SENSe<cnum>:CONTrol:DIO<id>:RFFE:CLOCk? <char>
```

**Return Type** Numeric

**Default** 50000

Note: The default value is the minimum integer value of clock rate, which meets the RFFE standard frequency range.

---

**SENSe<cnum>:CONTrol:DIO<id>:RFFE<rftech>:CSEQUence<csnum>:SADDress <char>, <num>**

**Applicable Models:** E5080B

*(Read-Write)* Set or read the secondary address (“SA” in GUI) for the specified command sequence.

**Parameters**

- `<cnum>` Channel number.
- `<id>` DIO number, 1 or 2
- `<rftech>` RFFE channel number. 1 to 4.
- `<csnum>` RFFE command sequence number. 1 to 16.
Character - when to send remote commands. Choose from:

**AFTer** - After the channel sweep ends.

**BEFore** - Before the channel sweep starts.

DUT RFFE secondary address. 0 to 15.

Examples

<table>
<thead>
<tr>
<th>Query</th>
<th>Syntax</th>
</tr>
</thead>
</table>

**Query Syntax**


**Return Type**

Numeric

**Default**

0

SENSe<ncnum>:CONT<cid>:DIO<rid>:RFFE<rffech>:CSEQ<csnum>:TYPE <char>, <enum>

**Applicable Models:** E5080B

*(Read-Write)* Set or read the command sequence type for the specified command sequence.

**Parameters**

- `<ncnum>` Channel number.
- `<id>` DIO number, 1 or 2
- `<rffech>` RFFE channel number. 1 to 4.
- `<csnum>` RFFE command sequence number. 1 to 16.
- `<char>` Character - when to send remote commands. Choose from:
  
  **AFTer** - After the channel sweep ends.

  **BEFore** - Before the channel sweep starts.

- `<enum>` RFFE command sequence type. Choose from:
  
  **R 0WRite** : Register 0 Write
  
  **RREad** : Register Read
  
  **RWRite** : Register Write
  
  **ERRead** : Extended Register Read
  
  **ERWRite** : Extended Register Write
Example:
```
SENS:CONT:DIO1:RFFE:CSEQ:TYPE AFT, R0WR
sense2:control:dio1:rffe1:csequence2:type? after
```

**Query Syntax:**
```
```

**Return Type:** Character

**Default:** RREad

---

SENSe<cnum>:CONTrol:DIO<id>:RFFE<rffech>:CSEQuence<csnum>:BCOunt <char>, <num>

**Applicable Models:** E5080B

(Read-Write) Set and read the byte count for the specified command sequence.

**Parameters**
- `<cnum>` Channel number.
- `<id>` DIO number, 1 or 2
- `<rffech>` RFFE channel number. 1 to 4.
- `<csnum>` RFFE command sequence number. 1 to 16.
- `<char>` Character - when to send remote commands. Choose from:
  - **AFTER** - After the channel sweep ends.
  - **BEFORE** - Before the channel sweep starts.
- `<num>` Byte Count value. Integer value. The value range is coupled with command sequence type setting.

<table>
<thead>
<tr>
<th>Command sequence type</th>
<th>Byte count range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register 0 Write</td>
<td>1 (fixed)</td>
</tr>
<tr>
<td>Register Read</td>
<td></td>
</tr>
<tr>
<td>Register Write</td>
<td></td>
</tr>
<tr>
<td>Extended Register Write</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Extended Register Read</td>
<td></td>
</tr>
</tbody>
</table>

**Examples**
```
SENS:CONT:DIO1:RFFE:CSEQ:BCO AFT, 4
```

Query Syntax
SENS<ch>:CONTro:DI<id>:RFFE<rfe>:CSEQ<cs>:BCOunt?
<ch>

Return Type Numeric
Default 1

SENS<ch>:CONTro:DI<id>:RFFE<rfe>:CSEQ<cs>:ADDRes<char>, <num>

Applicable Models: E5080B

(Read-Write) Set and read the address value for the specified command sequence.

Parameters
<ch> Channel number.
<id> DIO number, 1 or 2
<rfe> RFFE channel number. 1 to 4.
<cs> RFFE command sequence number. 1 to 16.
<char> Character - when to send remote commands. Choose from:
    AFTER - After the channel sweep ends.
    BEFORE - Before the channel sweep starts.
<num> Address value. Integer value. The value range is coupled with command sequence type setting.

<table>
<thead>
<tr>
<th>Command sequence type</th>
<th>Byte count range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register 0 Write</td>
<td>0 (fixed)</td>
</tr>
<tr>
<td>Register Read</td>
<td>#h00 to #h1F (0-31)</td>
</tr>
<tr>
<td>Register Write</td>
<td></td>
</tr>
<tr>
<td>Extended Register Write</td>
<td>#h00 to #hFF (0-255)</td>
</tr>
<tr>
<td>Extended Register Read</td>
<td></td>
</tr>
</tbody>
</table>

Examples
SEN:CONT:DI01:RFFE:CSEQ:ADD AFT, 4
sense2:control:diol:rffe1:csequence2:address? after

Query Syntax
SENS<ch>:CONTro:DI<id>:RFFE<rfe>:CSEQ<cs>:ADDRes<char>
SETSe<cnum>:CONTol:DIO<id>:RFFE<rffech>:CSEQ<csnum>:[:WRITe]:DATA <char>, <data>

**Applicable Models:** E5080B

*(Read-Write)* Set and read the data values for the specified command sequence.

This command works if the command sequence type is “Register 0 Write” or “Register Write” or “Extended Register Write”. If the command sequence type is “Register Read” or ”Extended Register Read”, this command will cause error.

**Parameters**

- `<cnum>`: Channel number.
- `<id>`: DIO number, 1 or 2
- `<rffech>`: RFFE channel number. 1 to 4.
- `<csnum>`: RFFE command sequence number. 1 to 16.
- `<char>`: Character - when to send remote commands. Choose from:
  - **AFTer** - After the channel sweep ends.
  - **BEFore** - Before the channel sweep starts.
- `<data>`: Comma separated list of data values. The value length is coupled with byte count setting. If data list length does not match with byte count setting, write command will cause error.

**Examples**

```
SENS:CONT:DI1:RFFE:CSEQ:WRIT:DATA AFT, 10
```

**Query Syntax**

```
```

**Return Type**

- Numeric
- **Default** 0

```
```
Applicable Models: E5080B

**(Read only)** Read the data and parity value pairs from DUT for the specified command sequence.

**Parameters**

- `<cnum>` Channel number.
- `<id>` DIO number, 1 or 2
- `<rffech>` RFFE channel number. 1 to 4.
- `<csnum>` RFFE command sequence number. 1 to 16.
- `<char>` Character - when to send remote commands. Choose from:
  - **AFTER** - After the channel sweep ends.
  - **BEFORE** - Before the channel sweep starts.

**Examples**

```
```

**Query Syntax**

- Comma separated numeric values, list of data and parity pairs.
- Ex. Byte count is 3 case, return values are below:
  - `[data#1],[parity#1],[data#2],[parity#2],[data#3],[parity#3]`

**Default** Not applicable

---

**SENS<cnum>:CONTrol:DIO<id>:RFFE<rffech>:CSEQuence:coun**

**Applicable Models: E5080B**

**(Read-Write)** Set and read the RFFE Command Sequence count. If user set the larger value than previously set, new RFFE Command Sequences will be added with default parameter value.

**Parameters**

- `<cnum>` Channel number.
- `<id>` DIO number
- `<rffech>` RFFE channel number. 1 to 4.
Character - when to send remote commands. Choose from:

**AFTER** - After the channel sweep ends.

**BEFORE** - Before the channel sweep starts.

**Examples**

```
SENS:CONT:DI01:RFFE:CSEQ:COUN AFT
```

**Query Syntax**

```
:SENSe<cnum>:CONTrol:DI0<id>:RFFE<rffe>:CSEQunce:COUNt?<char>
```

**Return Type**

Numeric

**Default**

0

---

**SENSe<cnum>:CONTrol:MACR:o[:STATe] <char>, <bool>**

**Applicable Models:** E5080B, N522xB, N523xB, N524xB

(Read-Write) Enables or disables software interface controls.

**Parameters**

- `<cnum>` Channel number.
- `<char>` Character - when to send remote commands. Choose from:
  - **AFTER** - After the channel sweep ends.
  - **BEFORE** - Before the channel sweep starts.
- `<bool>` **ON** or 1 - Turns software interface control ON.
  - **OFF** or 0 - Turns software interface control OFF.

**Examples**

```
SENS:CONT:MACR AFT, ON
sense2:control:macro? after
```

**Query Syntax**

```
:SENSe<cnum>:CONTrol:MACR:o[:STATe]? <char>
```

**Return Type**

Boolean

**Default**

OFF

---

**SENSe<cnum>:CONTrol:MACR:o:COMMand <char>, <cmdList>**
**Applicable Models:** E5080B, N522xB, N523xB, N524xB

*(Read-Write)* Set and read SCPI commands with target GPIB addresses (numbers) or VISA addresses. The specified SCPI commands are sent to the target instruments before the first trace on the channel begins sweeping. It is the end user’s responsibility to use this command.

**Parameters**

- `<cnum>` Channel number.
- `<char>` Character - when to send remote commands. Choose from:
  - **AFTer** - After the channel sweep ends.
  - **BEFore** - Before the channel sweep starts.
- `<cmdList>` The string of “\n” separates a pair of GPIB/VISA addresses and SCPI commands, and the string of “ ” separates GPIB/VISA address and SCPI in the following format;

  “address1 command1\naddress2 command2\n …”

**Examples**

- `SENS:CONT:MACR:COMM AFT`
- `sense2:control:macro? after`

**Query Syntax**

`:SENSe<cnum>:CONTro:MACRo:COMMand? <char>`

**Return Type**

String of comma-separated GPIB/VISA addresses and SCPI commands

**Default**

" "

**SENSe<cnum>:CONTro:MACRo:FILE:PATH <char>, <path>**

**Applicable Models:** E5080B, N522xB, N523xB, N524xB

*(Read-Write)* Set and read a file path to a macro. The macro is executed before the first trace on the channel begins sweeping. It is the end user’s responsibility to use this command. It’s needed to check the check box of “Enable Drive Access” in the SCPI dialog to execute the actual macro file.

**Parameters**

- `<cnum>` Channel number.
- `<char>` Character - when to send remote commands. Choose from:
  - **AFTer** - After the channel sweep ends.
  - **BEFore** - Before the channel sweep starts.
<path> Command line strings

Examples

SENSe:CONT:MACR:FILE:PATH AFT, "cscript D:\temp\test.vbs"

sense2:control:file:path? after

Query Syntax

:SENSe<cnum>:CONTrol:MACRo:FILE:PATH? <char>

Return Type

String

Default

" "

SENSe<cnum>:CONTrol:MACRo:FILE:ARGuments <char>, <arg>

Applicable Models: E5080B, N522xB, N523xB, N524xB

(Read-Write) Set and read arguments for a macro. The macro is executed before the first trace on the channel begins sweeping. It is the end user’s responsibility to use this command. It’s needed to check the check box of “Enable Drive Access” in the SCPI dialog to execute the actual macro file.

Parameters

<cnun> Channel number.

<char> Character - when to send remote commands. Choose from:

AFTer - After the channel sweep ends.

BEFore - Before the channel sweep starts.

<arg> Arguments for a macro

Examples

SENSe:CONT:MACR:FILE:ARG AFT, "localhost"

sense2:control:file? after

Query Syntax


Return Type

String

Default

" "

4020
Sense:Correction Commands

Performs and applies calibration and other error correction features.

- To perform a Guided Calibration, use ONLY the Sens:Corr Coll:GUIDed commands.
- To perform an Unguided Calibration, do NOT use the Sens:Corr:Coll:Guided commands.
- See the "Unguided" example programs for clarification.

```
SENSe:CORRection
CACHe:MODE
CCHeck
  | [ACQuire]
  | DONE
  | PARameter
CKIT - More Commands
COLLect
  | [ACQuire]
  | APPLy
  | CKIT - More Commands
  | DISPlay:WINDow
    | AOFF
    | [STATe]
  | GUIDed - More Commands
  | IDLE
    | TRIGger
  | ISOLation:
    | AVER:INCRement
    | ECAL:[STATe]
  | METHod
    | PORT:SUBS:
      | FULL:VAL
      | RESet
      | RESPonse:VAL
      | STAT
    | NOISe
      | ENR:ADAP:DEEMbed:[STATe]
      | LO:PCAL:[STATe]
      | PSEN:ADAP:DEEMbed:[STATe]
      | THRU:ADAPpter:DEEMbed:[STATe]
```
SENSe<cnum>:CORRection:CACHE:MODE <num>
Applicable Models: All

(Read-Write) Set this mode at ON for SNP file fast saving. This must be enabled before the SNP data is acquired by CALC:MEAS:DATA:SNP:PORTS:SAVE or CALC:DATA:SNP:PORTS:SAVE. This value is channel specific. If the channel does not exist, the default value is returned (1).

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<num>`
  - 0: Never correct all results.
  - 1: Always correct all results.
  - 2: Correct all results when correction is a full 2P and greater.
  - 3: Correct all results when correction is a full 4P and greater.
  - N: Correct all results when correction is a full NP and greater.

Examples

```
SENS2:CORR:CACHE:MODE 2  'any value > 0 and < 9 will work for this example
SENS:SWE:MODE SING;*OPC?
CALC:MEAS1:DATA:SNP:PORTS:SAVE
"1,2,3,4,5,6,7,8","multiportdevice.s8p", FAST
```

Query Syntax

SENSe:CORRection:CACHe:MODE?

Return Type

Numeric

Default

1 (Preset does not affect the setting.)

SENSe<cnum>:CORRection:CCheck[:ACQuire] <mod>[,.char]
Applicable Models: All

(Write-only) Reads the 'confidence data' associated with the specified ECal module and puts it into memory. The measurement is selected using `SENS:CORR:CCH:PAR`. This command is compatible with *OPC.

**Note:** A confidence check can NOT be performed remotely from User Characterizations that are stored on the VNA disk.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<mod>` ECal Module that contains the confidence data. Choose from:
  - **ECAL1**
  - ..through..
  - **ECAL50**

- `[char]` Optional argument. Specifies which characterization within the ECal module that the confidence data will be read from.
  - **CHAR0** Factory characterization (data that was stored in the ECal module by Keysight). Default if not specified.
  - **CHAR1** User characterization #1
  - **CHAR2** User characterization #2
  - ...and so forth up to:
  - **CHAR12** User characterization #12

**Examples**

- `SENS:CORR:CCH:acquire ecal1,char1`

**Query Syntax**

- Not applicable

**Default**

Not applicable

---

`SENSe<cnum>:CORR:CCHeck:DONE`
Applicable Models: All

(Write-only) Concludes the Confidence Check and sets the ECal module back into the idle state.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

Examples

- `SENS:CORR:CCH:DONE`
- `sense2:correction:ccheck:done`

Query Syntax

- Not applicable

Default

- Not applicable

SENSe<cnum>:CORRection:CCHeck:PARameter <Mname>

Applicable Models: All

(Read-Write) Specifies an existing measurement to be used for the Confidence Check.

**Note:** A confidence check can NOT be performed remotely from User Characterizations that are stored on the VNA disk.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<Mname>` Name of the measurement you are selecting for the confidence check. The measurement must already exist.

Examples

- `SENS:CORR:CCH:PAR 'TEST'`
  'selects the measurement "test" on channel 1 for the confidence check'
- `sense2:correction:ccheck:parameter 'test'`
  'selects the measurement "test" on channel 2 for the confidence check'

Query Syntax

- `SENSe<cnum>:CORRection:CCHeck:PARameter?`

  Returns the name of the selected measurement on channel `<cnum>`.

Return Type

- String

Default

- Not applicable

SENSe<cnum>:CORRection:COLLect[:ACQuire] <class>[,subclass][,sync]
Applicable Models: All

(Write-only) For UNGUIDED calibration, measures the specified standards from the selected calibration kit. The calibration kit is selected using the Sense:Correction:Collect:CKIT command.

For using two sets of standards, see SENS:CORR:TST.

Note: Before using this command you must select two items:
1. Select a calibration method using SENS:CORR:COLL:METH
2. Select a measurement using CALCulate:MEASure:DEFine. You can select one measurement for each channel.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<class>` Measures the standards associated with these class labels. Choose from:

<table>
<thead>
<tr>
<th>Label</th>
<th>SOLT (Forward)</th>
<th>SOLT (Reverse)</th>
<th>TRL</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAN1</td>
<td>SA</td>
<td>SA</td>
<td>TRL &quot;R&quot;</td>
</tr>
<tr>
<td>STAN2</td>
<td>SB</td>
<td>SB</td>
<td>N/A</td>
</tr>
<tr>
<td>STAN3</td>
<td>SC</td>
<td>SC</td>
<td>TRL &quot;L&quot;</td>
</tr>
<tr>
<td>STAN4</td>
<td>FWD TRANS</td>
<td>REV TRANS</td>
<td>TRL &quot;T&quot;</td>
</tr>
<tr>
<td>STAN5</td>
<td></td>
<td></td>
<td>Generic Isolation; not associated with calibration kit definition.</td>
</tr>
</tbody>
</table>

ECAL1 through ECAL modules

ECAL50 RESPONSE Same as Normalize selection in Unguided Cal. (subclass is ignored)
POWER Take a receiver power cal sweep and turn correction ON
SLSET Sets 'sliding load type', and increments the "number of slides" count. The total number of slides is critical to the correct calculation of the sliding load algorithm. See a sliding load cal example.
SLDONE Computes the sliding load using a circle fit algorithm.

[subclass] Optional argument. For mechanical calibration kits, choose from the following to specify the standard to be acquired from the SENS:CORR:COLL:CKIT:ORDer list. If not specified, subclass is set to SST1.
If an ECAL module (1 through 8) is specified for `<class>`, choose one of the following for specifying which characterization within the ECAL module will be used for the acquire. If not specified, the default is **CHAR0**.

**CHAR0** Factory characterization (data that was stored in the ECAL module by Keysight)

**CHAR1** User characterization #1

**CHAR2** User characterization #2

...and so forth up to:

**CHAR12** User characterization #12

[asynchronous] Optional argument. Choose from:

**SYNchronous** - blocks SCPI commands during standard measurement (default behavior)

**ASYNchronous** - does NOT block SCPI commands during standard measurement.

Learn more about this argument

**Examples**

```plaintext
SENS:CORR:COLL STAN1

'If SENS:CORR:COLL:CKIT:ORDer2 5,3,7 was specified, the following command measures standard 3 (the second in the order list)
sense1:correction:collect:acquire stan3,sst2

SENS:CORR:COLL ECAL4,ASYN; *OPC?
sense2:correction:collect:acquire ecal2,char1
```

**Query Syntax** Not applicable

**Default** Not applicable
SENSe<cnum>:CORRection:COLLect:APPLy

Applicable Models: All

(Write-only) Applies error terms to the measurement that is selected using CALCulate:MEASure:PARameter.

**Note:** Before using this command you must select a measurement using CALCulate:MEASure:DEFine. You can select one measurement for each channel.

**Note:** This command is only necessary if you need to modify error terms. If you do not need to modify error terms, SENSe<cnum>:CORRection:COLLect:SAVE calculates and then automatically applies error terms after you use SENS:CORR:COLL:ACQuire to measure cal standards.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Example**

1. CALCulate2:PARameter:SELect S21_2 'select the measurement to apply terms to
2. SENSe2:CORRection:COLLect:METHod SPARSOLT 'set type of cal method.
3. CALCulate2:DATA? SCORR1 'download the error term of interest
4. 'Modify the error term here
5. CALCulate2:DATA SCORR1 'upload the error term of interest
6. SENSe2:CORRection:COLLect:APPLy 'applies the error terms to the measurement

**Query Syntax**

- Not applicable

**Default**

- Not applicable

SENSe:CORRection:COLLect:DISPlay:WINDow:AOFF
Applicable Models: All

(Write-only) Clears the flags for windows to be shown during calibrations. To flag a window to be shown see SENS:CORR:COLL:DISP:WIND.

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Example</th>
</tr>
</thead>
</table>

See an example using this command.

Query Syntax Not Applicable

Default Not Applicable

SENSe:CORRection:COLLect:DISPlay:WINDow<wNum>[:STATe] <bool>

Applicable Models: All

(Write-only) Set the 'show' state of the window to be displayed during a calibration to view the measurements/channels. Learn more.

When this command is sent, the specified window is 'flagged' to be shown during calibration. The flag is cleared when the window is closed. A Preset or Instrument State Recall also closes the window. If the same window number is reopened, this command must be sent again to show the window during a calibration. The flag is NOT saved with an instrument state.

Send this command for each additional window to show during a calibration.

Parameters

<wNum> Window number to show during a calibration. The calibration window will also be shown with this window.

The window must already be created.

Use DISPlay:CATalog? to read all existing window numbers.

<bool> Window state. Choose from:

ON (or 1) - Show the specified window during calibration.

OFF (or 0) - Do NOT show the specified window during calibration.

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:COLL:DISP:WIND1 1</td>
<td>sense:correction:collect:display:window2:state off</td>
</tr>
</tbody>
</table>

See an example using this command.
SENSe:CORRection:COLLect:IDLE:TRIGger \<bool>\n
**Applicable Models:** All

*(Read-Write)* This command enables/disables asynchronous sweeping during a calibration acquisition.

**Parameters**

\<bool> Choose from:

- **ON** (or 1) - Enable asynchronous sweeping during a calibration acquisition (the default legacy behavior).
- **OFF** (or 0) - Disable asynchronous sweeping during a calibration acquisition.

**Examples**

SENS:CORR:COLL:IDLE:TRIG 1

**Query Syntax**

SENSe:CORRection:COLLect:IDLE:TRIGger?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON

---

SENS:CORRection:COLLect:ISOLation:AVERage:INCRement \<num>\n
**Applicable Models:** All

*(Read-Write)* Specifies amount to increment (increase) the channel averaging factor during isolation measurement of the ECal module during an unguided ECal calibration.

**Note:** if the channel currently has averaging turned OFF and \<num> is greater than 1, averaging will be turned ON only during the isolation measurements and with the averaging factor equal to \<num>.

**Parameters**

\<num> Incremental Averaging factor. The maximum averaging factor is 65536 (2^16).

**Examples**

SENS:CORR:COLL:ISOL:AVER:INCR 16

sense:correction:collect:isolation:average:increment 0

**Query Syntax**

SENSe:CORRection:COLLect:ISOLation:AVERage:INCRement?

**Return Type**

Numeric
SENSe<cnum>:CORRection:COLLect:ISOLation:ECAL[:STATe] <bool>

**Applicable Models:** All

(Read-Write) Specifies whether or not the isolation state of the ECal module will be measured as part of an unguided ECal calibration.

An unguided calibration is performed using the SENS:CORR:COLL:METH and SENS:CORR:COLL:ACQ commands.

**Note:** The inherent isolation of the VNA is better than that attained with this command. ONLY use this command when using an external test set, and ONLY using a 8509x ECal module.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` ON (or 1) - isolation is measured during the unguided ECal calibration.
  - OFF (or 0) isolation is NOT measured during the unguided ECal calibration.

**Examples**

```plaintext
SENS1:CORR:COLL:ISOL:ECAL ON
sense2:correction:collect:isolation:ecal:state 0
```

**Query Syntax**

SENSe:CORRection:COLLect:ISOLation:ECAL:STATe?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

---

SENSe<cnum>:CORRRection:COLLect:METHod <char>

**Applicable Models:** All

(Read-Write) For UNGUIDED calibration, sets the calibration method (also known as 'Calibration Type' on calibration dialog box.) To select a Cal Type from a Cal Set, use CALC:MEAS:CORR:TYPE.

**Note:** Before using this command you must select a measurement using CALCulate:PARameter:SELect. You can select one measurement for each channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Choose from:
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>No Cal method</td>
</tr>
<tr>
<td>REFL1OPEN</td>
<td>Response Open</td>
</tr>
<tr>
<td>REFL1SHORT   or REFL1</td>
<td>Response Short</td>
</tr>
<tr>
<td></td>
<td>Full 1 port</td>
</tr>
<tr>
<td>RESPONSE</td>
<td>Same as Normalize selection in Unguided Cal.</td>
</tr>
<tr>
<td>RPOWER</td>
<td>Receiver Power Cal - Used only with receiver measurements.</td>
</tr>
<tr>
<td>TRAN1</td>
<td>Response Thru - Requires a Thru standard.</td>
</tr>
<tr>
<td>TRAN2</td>
<td>Response Thru and Isolation - Requires a Thru standard.</td>
</tr>
<tr>
<td>SPARSOLT</td>
<td>Full SOLT 2 port</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:CORR:COLL:METH REFL1
sense2:correction:collect:method sparsolt
```

**Query Syntax**

```
SENSe<cnum>:CORRection:COLLect:METHod?
```

**Return Type**

Character

**Default**

Not Applicable


**Applicable Models:** M9485A

*(Read-Write)*: Specify the ports allowed to participate in a multiport calibration.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port numbers>`: comma separated list of ports
Calset is 16 port VNA with a 16 port calibration.
Full 3Port (1,2,3) error correction on ports 1,2,3
All other port parameters are uncorrected.

Query Syntax
Return Type Character
Default Not Applicable

SENSe<cnum>:CORRection:COLLect:METHod:PORT:SUBS:RESet
Applicable Models: M9485A
(Write Only): Reset the full and response list of the port subset correction.

Parameters
<cnum> Any existing channel number. If unspecified, value is set to 1


Query Syntax Not Applicable
Return Type Not Applicable
Default Not Applicable

Applicable Models: M9485A
(Read-Write): Specify the ports that can be corrected using lesser corrections (enhanced response). This list is not allowed to overlap with the full multiport calibration list specified by :SENS:CORR:COLL:METH:PORT:FULL:VAL.

Parameters
<cnum> Any existing channel number. If unspecified, value is set to 1
<port numbers> comma separated list of ports
Examples

Calset: 16 port VNA with a 16 port calibration

Example 1
SENS:CORR:METH:PORT:SUBS:FULL:VAL 1,2,3,4,5,6
SENS:CORR:METH:PORT:SUBS:RESP:VAL 7,8,
Full 6 Port on ports 1 to 6
Enhanced Response for parameters involving ports 7 and 8
No correction for ports 9 to 16

Example 2
SENS:CORR:METH:PORT:SUBS:FULL:VAL 0
SENS:CORR:METH:PORT:SUBS:RESP:VAL 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16
All parameters are enhance response corrected

Query Syntax

Return Type Character
Default Not Applicable

SENSe<cnum>:CORRection:COLLect:METHod:PORT:SUBS:STAT <bool>

Applicable Models: M9485A

(Read-Write) Enabling the port subset correction.

Parameters
<cnun> Any existing channel number. If unspecified, value is set to 1
<bool> Devolve Calibration state. Choose from:
OFF or 0 - Off.
ON or 1 - On

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

*(Read-Write)* Set and read the state of ENR Adapter de-embedding. Learn more.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` ENR Adapter de-embed state. Choose from:
  - **OFF** or **0** - Do not force de-embedding.
  - **ON** or **1** - Force de-embedding.

**Examples**

<table>
<thead>
<tr>
<th>Sensing Channel</th>
<th>Query Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSENSe</td>
<td>SENS:CORR:COLL:NOIS:ENR:ADAP:DEEM 0</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe:CORRection:COLLect:NOISe:ENR:ADAPter:DEEMbed:[STATe]?  

**Return Type** Boolean

**Default** O - OFF

SENSe<ch>:CORRection:COLLect:NOISe:<LO<n>>:PCAL[:STATe] <bool>
Applicable Models: VNAs with Noise Figure Option (S9x029A/B, 028, 029) (Excepts M9485A, M980xA, P50xxA)

(Read-Write) Enables and disables LO power calibration for NFX.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` LO Stage (number). Choose 1 for NFX.
- `<bool>` LO Power Cal state. Choose from:
  - OFF or 0 - Disable LO Power Cal
  - ON or 1 - Enable LO Power Cal

Examples

- `SENS:CORR:COLL:NOIS:LO1:PCAL 0`
- `sense2:correction:collect:noise:lo1:pcal:state ON`

Query Syntax

- `SENSe:CORRection:COLLect:NOISe:LO<n>:PCAL:STATe?`

Return Type

- Boolean

Default

- O - OFF


Applicable Models: VNAs with Noise Figure Option (S9x029A/B, 028, 029)

(Read-Write) Set and read the state of power sensor adapter de-embedding. Learn more.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` Power sensor adapter de-embed state. Choose from:
  - OFF or 0 - Do not force de-embedding.
  - ON or 1 - Force de-embedding.

Examples

- `SENS:CORR:COLL:NOIS:PSEN:ADAP:DEEM 0`

Query Syntax

- `SENSe:CORRection:COLLect:NOISe:PSENsor:ADAPter:DEEMbed:[STATe]?`

Return Type

- Boolean

Default

- 0 - OFF

4037

**Applicable Models:** VNAs with Noise Figure Option (S9x029A/B, 028, 029)

*(Read-Write)* Set and read the state of the thru adapter de-embedding. Learn more.

**Parameters**

- **<ch>** Any existing channel number. If unspecified, value is set to 1
- **<bool>** Thru adapter de-embed state. Choose from:
  - **OFF** or **0** - Do not force de-embedding.
  - **ON** or **1** - Force de-embedding.

**Examples**

```plaintext
SENS:CORR:COLL:NOIS:THRU:ADAP:DEEM 0
```

**Query Syntax**

`SENSe:CORRection:COLLect:NOISe:THRU:ADAPter:DEEMbed:[STATe]?`

**Return Type** Boolean

**Default** O - OFF

---

SENSe<cnum>:CORRection:COLLect:SAVE

**Applicable Models:** All

*(Write-only)* For UNGUIDED calibrations ONLY. This command does the following:

- calculates the error terms using the selected :METHod
- applies the error terms to the selected measurement (turns error correction ON.)
- saves the calibration error-terms to the channels Cal Register or a User Cal Set.

The Cal Register or User Cal Set is determined by the setting of the

Do NOT use this command during an ECAL. When performing an ECAL calibration using `SENSe:CORR:COLL:ACQuire`, this SAVE operation is performed automatically before the completion of a successful ACQuire.
Before using this command you must select a measurement using `CALCulate:MEASure:PARameter`. You can select one measurement for each channel.

**Parameters**

- `<cnum>`
  - Any existing channel number. If unspecified, value is set to 1

**Examples**

- `SENS:CORR:COLL:SAVE`
- `sense2:correction:collect:save`

**Query Syntax**

Not applicable

**Default**

Not applicable

---

**SENSe:CORRection:COLLect:SWEep:CHANnel:AOFF**

**Applicable Models:** All

(Write-only) Clears ALL flags for channels to sweep during calibration. To flag a channel, see `SENS:CORR:COLL:SWE:CHAN`.

**Examples**

- `SENS:CORR:COLL:SWE:CHAN:AOFF`
- `sense:correction:collect:sweep:channel:aoff`

**Default**

Not applicable

---

**SENSe<cnum>:CORRection:COLLect:SWEep:CHANnel<enum2>[:STATE] <bool>**

**Applicable Models:** All

(Write-only) Specifies an alternate channel (other than the calibration channel) to be viewed while using the calibration wizard.

When this command is sent, the `<cnum2>` channel is 'flagged' to be swept during calibration.

The flag is cleared when the channel is deleted, if the Measurement Class is changed, or if all measurements are deleted from the channel.

If the same channel number is recreated, this command must be sent again to sweep the channel during a calibration. The flag is NOT saved with an instrument state.

A Preset or Instrument State Recall deletes the channel.

**Note:** This command is intended to be used in conjunction with the interactive calibration wizard.

**Parameters**
The channel to be calibrated. If unspecified, value is set to 1.

The channel to sweep when waiting to measure a standard.

This channel must already exist with at least one measurement in the channel. If this channel is in continuous sweep mode, it must have the same attenuator settings and path configuration (PNA-X only).

Channel sweep state. Choose from:

ON (or 1) - Sweep the channel during calibration.

OFF (or 0) - Do NOT sweep the channel during calibration.

Examples

```
SENSe:CORRection:COLL:SWE:CHAN2 1
sense2:correction:collect:sweep:channel3:state off
```

See an example using this command.

Query Syntax

Not Applicable

Default

OFF

SENSe:CORRection:ENR:CALibration:TABLe:DATA <freq, value, freq, value...>

Applicable Models: All

(Read-Write) Set and read the ENR (Excess Noise Ratio) calibration data. All of the frequency and ENR data must be sent at the same time. Use MMEM:LOAD to load, and MMEM:STORE:ENR CAL to save ENR table data from disk. Learn more about Noise Source ENR files.

Parameters

<freq, value> (Numeric) ENR data. Frequency value in Hz followed by a ENR noise value in dB. Enter as many pairs as necessary.

Examples

```
SENSe:CORRection:ENR:CALibration:TABLe:DATA 1.0E9,14.37,2.5E9,15.28
sense:correction:enr:calibration:table:data 1.0E9,14.37,2.5E9,15.28
```

Query Syntax

SENSe:CORRection:ENR:CALibration:TABLe:DATA?

Return Type

Comma separated numeric values

Default

Not Applicable

SENSe:CORRection:ENR:CALibration:TABLe:ID:DATA <id>
Applicable Models: All

(Read-Write) Sets and returns ID of ENR table. While this is for informational purposes only, it can be used to record the model of the noise source. Learn more about ENR files.

**Parameters**

<id> (String) Identifier for the ENR table.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:ENR:CAL:TABL:ID:DATA &quot;346C&quot;</td>
<td>ENR Table ID: 346C</td>
</tr>
<tr>
<td>sense:correction:enr:calibration:table:id:data &quot;ENR Table&quot;</td>
<td>ENR Table ID: 346C</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe:CORRection:ENR:CALibration:TABLe:ID:DATA?

**Return Type**

String

**Default**

Not Applicable

---

SENSe:CORRection:ENR:CALibration:TABLe:SERial:DATA <sn>

Applicable Models: All

(Read-Write) Sets and returns the serial number of noise source. This is for informational purposes only to identify the specific noise source for which the data pertains. Learn more about ENR files.

**Parameters**

<sn> Serial number of the noise source for which the data applies, enclosed in quotes.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

**Query Syntax**

SENSe:CORRection:ENR:CALibration:TABLe:SERial:DATA?

**Return Type**

String

**Default**

Not Applicable

---

SENSe<ch>:CORRection:GCSetup:POWer <num>
**Applicable Models:** N522xB, N524xB, M9485A, E5080

*(Read-Write)* Set and read the power level at which to perform the Source Power Cal portion of a Gain Compression (Opt S93086A/B) Calibration.  Learn more about this setting.

**Parameters**

<table>
<thead>
<tr>
<th>&lt;num&gt;</th>
<th>Power level in dB. Choose a value from +30 to (-30).</th>
</tr>
</thead>
</table>

**Examples**

```
SENS:CORR:GCS:POW 0
sense:correction:gcsetup:power 5
```

**Query Syntax**

```
SENSe:CORRection:GCSetup:POWer?
```

**Return Type**

Numeric

**Default**

0

---

**SENSe<ch>:CORRection:GCSetup:SENSor:CKIT <string>**

**Applicable Models:** N522xB, N524xB, M9485A, E5080

*(Read-Write)* Set and read the cal kit to be used for calibrating at the port 1 reference plane when the power sensor connector is different from the DUT port 1.  Learn more.

**Parameters**

|----------|---------------------------------------------------------------------------------------|

**Examples**

```
SENS:CORR:GCS:SENS:CKIT "85052B"
```

**Query Syntax**

```
SENSe:CORRection:GCSetup:SENSor:CKIT?
```

**Return Type**

String

**Default**

Not Applicable

---

**SENSe<ch>:CORRection:GCSetup:SENSor:CONNector<string>**
Applicable Models: N522xB, N524xB, M9485A, E5080

(Read-Write) Set and read the power sensor connector type which is used to perform the Source Power Cal portion of a Gain Compression Calibration. Learn more.

Parameters


Select "Ignored" to NOT compensate for the adapter.

Examples

```
SENS:CORR:GCS:SENS:CKIT "3.5 mm (50) male"
```

Query Syntax

`SENSe:CORRection:GCSetup:SENSor:CKIT?`

Return Type

String

Default

Not Applicable

SENSe:CORRection:IMPedance:INPut:MAGNitude <num>

Applicable Models: All

(Read-Write) Sets and returns the system impedance value for the analyzer.

Parameters

<num> System Impedance value in ohms. Choose any number between 0.001 and 1000 ohms.

Examples

```
SENS:CORR:IMP:INP:MAGN 75
sense:correction:impedance:input:magnitude 50.5
```

Query Syntax

`SENSe:CORRection:IMPedance:INPut:MAGNitude?`

Return Type

Numeric

Default

50

SENSe<ch>:CORRection:INTerpolate[:STATe] <ON | OFF>
Applicable Models: All

(Read-Write) Turns correction interpolation ON or OFF.

Note: Before using this command you must select a measurement using CALC:PAR:SEL. You can select one measurement for each channel.

Parameters

- `<ch>`: Any existing channel number. If unspecified, value is set to 1.
- `<ON | OFF>`: 
  - ON (or 1) - turns interpolation ON.
  - OFF (or 0) - turns interpolation OFF.

Examples

- `SENS:CORR:INT ON`
- `sense2:correction:interpolate:state off`

Query Syntax

- `SENSe<cnum>:CORRection:INTerpolate[:STATe]`?

Return Type

- Boolean (1 = ON, 0 = OFF)

Default

ON

SENSe<ch>:CORRection:ISOLation[:STATe] <ON | OFF> OBSOLETE

This command no longer works beginning in the VNA 5.2 release. The set and query of this command will NOT return an error.

To perform isolation as part of an unguided calibration, you must explicitly measure the isolation standard using `SENS:CORR:COLL:ACQ Stan5`.

To measure isolation as part of an ECal, use `SENS:CORR:COLL:ISOL:ECAL`.

(Read-Write) Turns isolation cal ON or OFF during Full 2-port calibration. If this command is not sent, the default state is to disable Isolation.

Parameters

- `<ch>`: Any existing channel number. If unspecified, value is set to 1.
- `<ON | OFF>`: 
  - ON (or 1) - turns isolation ON.
  - OFF (or 0) - turns isolation OFF.

Examples

- `SENS:CORR:ISOL ON`
- `sense2:correction:isolation:state off`

Query Syntax

- `SENSe<cnum>:CORRection:ISOLation[:STATe]`?

Return Type

- Boolean (1 = ON, 0 = OFF)
**SENSe<ch>:CORRection:METHods:MATCh <bool>**

**Applicable Models:** All

*(Read-Write)* Turns match-correction ON or OFF. Use this command AFTER performing an Guided Power Cal. Learn more.

**Parameters**

- `<ch>` Channel number on which Guided Power Cal was performed. If unspecified, value is set to 1.
- `<bool>`
  - **ON** (or 1) - Turns match-correction ON
  - **OFF** (or 0) - Turns match-correction OFF.

**Examples**

```
SENS:CORR:METH:MATC 0
sense2:correction:methods:match off
```

**Query Syntax**

`SENSe<cnum>:CORRection:METHods:MATCh?`

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default** ON

---

**SENSe<ch>:CORRection:METHods:PORT:SUBSet:FULL[:VALue] <port numbers>**

**Applicable Models:** All

*(Read-Write)* Sets and returns the selected ports to include in a full NPort correction. If other ports are selected using the `SENSe:CORRection:METHods:PORT:SUBSet:RESPonse[:VALue]` command, they will be corrected with enhanced response calibration. Ports not selected using the `SENSe:CORRection:METHods:PORT:SUBSet:FULL[:VALue]` or `SENSe:CORRection:METHods:PORT:SUBSet:RESPonse[:VALue]` command are uncorrected. Learn more.

The default is all ports are selected for a full NPort correction. To assign a port to be corrected with enhanced response calibration using the `SENSe:CORRection:METHods:PORT:SUBSet:RESPonse[:VALue]` command, they first have to be removed from the full NPort correction list. Ports can be removed from the full NPort correction list by setting a new list that excludes the ports to be corrected with enhanced response calibration.

For information about port sub-setting, refer to Port Sub-Setting Examples.

**Note:** The `SENSe:CORRection:METHods:PORT:SUBSet[:STATe]` must be set to ON to enable the full
command.

**Parameters**

- `<ch>` Channel number.
- `<port numbers>` Comma separated list of ports to include in the full correction. The default is all ports selected.

**Examples**

16-port VNA with an active 16-port calibration

```plaintext
SENS:CORR:METH:PORT:SUBS:STAT 1
SENS:CORR:METH:PORT:SUBS:FULL:VAL 1,2,3
sense2:correction:methods:port:subset:full:value 1,2,3

Result: Full correction on ports 1, 2, and 3
All other port parameters are uncorrected
```

**Query Syntax**

`SENSe<ch>:CORRection:METHods:PORT:SUBSet:FULL[:VALue]?`

**Return Type**

Array_int

**Default**

All ports included

---

**SENSe<ch>:CORRection:METHods:PORT:SUBSet:RESSet**

**Applicable Models:** All

(Write) Resets the full and response list to their default values. Learn more.

For information about port sub-setting, refer to Port Sub-Setting Examples.

**Parameters**

- `<ch>` Channel number.

**Examples**

```plaintext
SENS:CORR:METH:PORT:SUBS:RES
sense2:correction:methods:port:subset:reset
```

**Return Type**

Not applicable

**Default**

Not applicable

---

**SENSe<ch>:CORRection:METHods:PORT:SUBSet:RESPonse[:VALue] <port numbers>**

**Applicable Models:** All

(Read-Write) Sets and returns the selected ports to be corrected with enhanced response calibration. If other ports are selected using the `SENSe:CORRection:METHods:PORT:SUBSet:FULL[:VALue]` command, they will be corrected in a full NPort correction. Ports not selected using the `SENSe:CORRection:METHods:PORT:SUBSet:RESPonse[:VALue]` or

To assign a ports to be corrected with enhanced response calibration ensure that they are not selected for the full NPort correction list. Ports can be removed from the full NPort correction list by setting a new list that excludes the ports to be corrected with enhanced response calibration.

For information about port sub-setting, refer to Port Sub-Setting Examples.

**Note:** The SENSe:CORRection:METHods:PORT:SUBSet[:STATe] must be set to ON to enable the response command.

**Parameters**

- `<ch>` Channel number.
- `<port numbers>` Comma separated list of ports to include for enhanced response correction. The default is no ports selected.

**Examples**

**Example #1:**
16-port VNA with an active 16-port calibration

SENSe:CORR:METH:PORT:SUBS:STAT 1
SENSe:CORR:METH:PORT:SUBS:FULL:VAL 1,2,3,4,5,6
SENSe:CORR:METH:PORT:SUBS:RESP:VAL 7,8

Result: Full correction on ports 1-6
Enhanced response corrected for parameters involving ports 7 and 8
No correction for ports 9-16

**Example #2:**
16-port VNA with an active 16-port calibration

SENSe:CORR:METH:PORT:SUBS:FULL:VAL 0
SENSe:CORR:METH:PORT:SUBS:RESP:VAL 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16

Result: Enhanced response correction for parameters involving any ports

**Query Syntax**  SENSe<ch>:CORRection:METHods:PORT:SUBSet:RESPonse[:VALue]?

**Return Type** Array_int

**Default** Empty list

SENSe<ch>:CORRection:METHods:PORT:SUBSet[:STATe] <bool>
Applicable Models: All

(Read-Write) Set and return the ON/OFF subset correction state. Learn more.

For information about port sub-setting, refer to Port Sub-Setting Examples.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Channel number.</td>
</tr>
<tr>
<td>&lt;bool&gt;</td>
<td>Choose from:</td>
</tr>
<tr>
<td>0- OFF</td>
<td>Subset correction OFF.</td>
</tr>
<tr>
<td>1- ON</td>
<td>Subset correction ON.</td>
</tr>
</tbody>
</table>

Examples

```
SENS:CORR:METH:PORT:SUBS:STAT 1
sense2:correction:methods:port:subset:state 1
```

Query Syntax

```
SENSe<cnum>:CORRection:METHods:PORT:SUBSet:[;STATE]?
```

Return Type

Boolean

Default

0

SENSe<ch>:CORRection:MODel <param>

Applicable Models: All except E5080A, M9485A

(Write-Read) Sets and returns the error term model.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;parm&gt;</td>
<td>Error term method. Choose from:</td>
</tr>
<tr>
<td>TERM8</td>
<td></td>
</tr>
<tr>
<td>TERM10</td>
<td></td>
</tr>
</tbody>
</table>

Examples

```
SENS1:CORR:MOD TERM8
```

Query Syntax

```
SENSe<cnum>:CORRection:MODel?
```

Return Type

String

Default

TERM10
SENSe:CORRection:PREFerence:CALibration[:FOM]:RANGe <char>

Applicable Models: All

(Read-Write) Specifies the FOM frequency range to use when performing calibration.

Parameters

<char> Choose from:

PRIMary - Used for calibrating at the mmWave frequencies when NOT using a test set. Learn more.

AUTO - All other calibration situations.

Examples

SENSe:CORRection:PREFerence:CALibration[:FOM]:RANGe?

Query Syntax

SENSe:CORRection:PREFerence:CALibration[:FOM]:RANGe AUTO

Default AUTO

SENSe:CORRection:PREFerence:CSET:SAVE <char>

Applicable Models: All

Important Notes:

- This command replaces SENSe:CORRection:PREFerence:CSET:SAVE
- With 6.0 we implemented a change that defaults to saving completed calibrations to Cal Registers instead of User Cal Sets. To revert to the old behavior, send this command with the USER argument.

(Read-Write) Specifies the default manner in which calibrations that are performed using SCPI or COM are to be stored. Cal data is ALWAYS stored to the channel Cal Register regardless of this setting.

This setting survives instrument preset and reboot. It remains until changed by another execution of this command.

Note: Cal Set arguments used with commands such as SENSe:CORRection COLL:GUID:INIT, SENSe:CORRection COLL:GUID:SAVE and SENSe:CORRection COLL:GUID:SAVE:CSET will override any of these default preference settings.

Learn about Cal Registers and User Cal Sets.
### Parameters

**<char> CALRegister** - Each Cal is saved ONLY to the channel Cal Register. If the error terms from a new Cal can co-exist with those in the Cal Register, they are appended.

**USER** - Each Cal is saved to its own new User Cal Set file. The Cal Set name is automatically generated. To change the name, send `SENS:CORR:CSET:NAME` after the cal is complete. This reverts to pre-6.0 behavior.

**REUSE** - The cal is saved to the Cal Set that is currently selected on the specified channel, which could be the channel Cal Register. If the channel does not yet have a selected Cal Set, the cal will be saved to a new User Cal Set with an automatically-generated name. If the error terms from a new Cal can co-exist with those in the Cal Set, they are appended.

### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:PREF:CSET:SAVE USER</td>
</tr>
<tr>
<td>sense:correction:preference:cset:save reuse</td>
</tr>
</tbody>
</table>

### Query Syntax

`SENSe:CORRection:PREFerence:CSET:SAVE?`

### Return Type

Character

### Default

CALRegister

---

**SENSe:CORRection:PREFerence:CSET:SAVUser <bool> Superseded**

**Applicable Models:** N522xB, N523xB, N524xB, M9485A

This command is replaced with `SENS:CORR:PREF:CSET:SAVE`

**NOTE:** With 6.0 we implemented a change that defaults to saving completed calibrations to Cal Registers instead of User Cal Sets. To revert to the old behavior, send this command as ON (1). For UI and COM use, this can be done from the GPIB console.

(Read-Write) Specifies whether cal data is automatically saved to a User Cal Set file after performing a SCPI calibration. Cal data is always saved to a Cal Register regardless of this setting.

This setting survives instrument preset and reboot. It remains until changed by another execution of this command.

Learn about [Cal Registers and User Cal Sets](#).

### Parameters

**<bool> ON or 1** - Cal is automatically saved to a User Cal Set file when performing a SCPI calibration. The Cal Set name is automatically generated. To change the
name, send SENS:CORR:CSET:NAME after the cal is complete. Reverts to pre-6.0 behavior.

**OFF** or **0** - Cal is NOT automatically saved to a User Cal Set. To save a calibration to a User Cal Set, use SENS:CORR:COLL:GUID:INIT.

**Examples**

<table>
<thead>
<tr>
<th>SENS:CORR:PREF:CSET:SAV</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>sense:correction:preference:cset:savuser</td>
<td>0</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENS:CORR:PREF:CSET:SAVUser?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF (0)

---

**SENS:CORR:PREF:ECAL:ORIentation[:STATe] <ON|OFF>**

**Applicable Models:** All

(Read-Write) Specifies whether or not the VNA should perform orientation of the ECal module during calibration. Orientation is a technique by which the VNA automatically determines which ports of the module are connected to which ports of the VNA. Orientation begins to fail at very low power levels or if there is much attenuation in the path between the VNA and the ECal module. If orientation is turned OFF, the SENS:CORR:PREF:ECAL:PMAP command must be used to specify the port connections before performing a cal.

**Note:** For 3-port or 4-port measurements, when orientation is OFF, you are not allowed to specify how the ECAL module is connected. Instead, the VNA determines the orientation. Use SENS:CORR:COLL:GUID:DESC? to query the orientation. The VNA does not verify that you made the connection properly.

This setting remains until the VNA is restarted or this command is sent again.

**Parameters**

- `<bool>`
  - ECAL orientation state. Choose from:

  **ON** or **1** - VNA performs orientation of the ECal module.

  **OFF** or **0** - VNA does NOT performs orientation of the ECal module.

**Examples**

<table>
<thead>
<tr>
<th>SENS:CORR:PREF:ECAL:ORI</th>
<th>OFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>sense:correction:preference:ecal:orientation:state</td>
<td>on</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENS:CORR:PREF:ECAL:ORIentation[:STATe]?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON (1)
SENSe:CORRection:PREFerence:ECAL:OVERrange[:STATE] <ON|OFF>

Applicable Models: All

(Read-Write) Sets and returns the ECAL over range state.

Parameters

<bool> ECAL over range state. Choose from:

ON or 1 - Enable ECAL over range.

OFF or 0 - Disable ECAL over range.

Examples

SENSe:CORRection:PREFerence:ECAL:OVER range:state on

Query Syntax

SENSe:CORRection:PREFerence:ECAL:OVERrange[:STATE]?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON (1)

SENSe:CORRection:PREFerence:ECAL:PMAP <module>,<string>

Applicable Models: All

(Read-Write) When ECal module orientation is turned OFF (SENSe:CORRection:PREFerence:ECAL:ORI), this command specifies the port mapping (which ports of the module are connected to which ports of the VNA) prior to performing ECAl calibrations.

This setting remains until the VNA is restarted or this command is sent again.

Parameters

<module> Specifies which ECAl module this port map is being applied to. Choose from:

ECAL1

...through.

ECAL50

<string> Format this parameter in the following manner:

Aw,Bx,Cy,Dz

where
- A, B, C, and D are literal ports on the ECAL module
- w,x,y, and z are substituted for VNA port numbers to which the ECAL module port is connected.

Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECal module with

- port A connected to VNA port 2
- port B connected to VNA port 3
- port C not connected
- port D connected to VNA port 1

the string would be: A2,B3,D1

If either the receive port or source port (or load port for 2-port cal) of the CALC:PAR:SELelected measurement is not in this string and orientation is OFF, an attempt to perform an ECal calibration will fail.

### Examples

- sense:correction:preference:ecal:pmap ecal3, 'a2,b1,c3'

### Query Syntax

SENSe:CORRection:PREFerence:ECAL:PMAP? <module>

### Return Type

String

### Default

Null string ()
This command is no longer supported. Learn more about old and new behaviors.

(Read-Write) Sets and returns a preference for the Unguided Cal behavior described below. This setting persists until it is changed.

This preference can also be set ON by executing the script on the VNA at C:\Program Files(x86)\Keysight\Network Analyzer\System\wincal32.reg.

Parameters
- `<bool>` Boolean - Choose from:
  - **0 - OFF** - Reverts to new (preferred) behavior. An error is returned if standard data is not acquired before sending `SENS:CORR:COLL:SAVE`.
  - **1 - ON** - (WinCal compatible) Prevents `SENS:CORR:COLL:SAVE` from failing when standard data has not, and will not, be acquired.

Examples
```
SENS:CORR:PREF:SIMC 0
sense:correction:preference:simcal 1
```

Query Syntax
```
SENSe:CORRection:PREFerence:SIMCal?
```

Return Type
- Boolean

Default
- 0

---

SENSe:CORRection:PREFerence:TRIG:FREE `<char>`, `<bool>`

Applicable Models: All

(Read-Write) Sets and returns the preference for the trigger behavior during a calibration. This setting persists until it is changed.

Note: If `TRIGger:SOURce` = Manual, during a calibration the VNA ALWAYS switches to Internal for one trigger, then back to Manual, regardless of this preference command.

Parameters
- `<char>` Character - Calibration type. Choose from:
  - `GUIDed` - preference setting pertains to a Guided calibration.
  - `UNGuided` - preference setting pertains to an Unguided calibration.

- `<bool>` Boolean - Choose from:
  - **0 - OFF** - The trigger behavior during the specified calibration type DOES
respect the setting of the TRIGger:SOURce command. For example, when Trigger source = External, the single trigger method will wait for the External trigger signal and then allow only one sweep.

1 - ON - (Pre-6.0 behavior) The trigger behavior during the specified calibration type does NOT respect the setting of the TRIGger:SOURce command. For example, when Trigger source = External, during calibration the VNA switches to Internal sweep, responds to one trigger signal to measure the standard, then switches back to External.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
</table>

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Offset Value in dB. Specify loss as a negative number; and gain as a positive number. Choose a number between -200 and 200.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:CORR:RPOW:OFFS .5</td>
<td>sense:correction:rpower:offset:amplitude .-5</td>
<td>0</td>
</tr>
</tbody>
</table>

SENSe<cnum>:CORRection:RPOWer:OFFSet[:AMPLitude] <num>

(Read-Write) Adjusts a receiver power cal to account for components or adapters that are added between the source port and receiver while performing this cal. For more information, see Receiver Cal.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Offset Value in dB. Specify loss as a negative number; and gain as a positive number. Choose a number between -200 and 200.
Applicable Models: All

(Read-Write) Sets the velocity factor to be used with Electrical Delay and Port Extensions.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Velocity factor. Choose a number between 0 and 10 (.66 polyethylene dielectric; .7 PTFE dielectric)

Examples

- `SENS:CORR:RVEL:COAX .66`
- `sense2:correction:rvelocity:coax .70`

Query Syntax

- `SENSe<cnum>:CORRection:RVELocity:COAX?`

Return Type

- Numeric

Default

- 1

SENSe<cnum>:CORRection:SFORward[:STATe] <boolean>

Applicable Models: All

(Read-Write) Sets the direction a calibration will be performed when only one set of standards is used.

Use `SENSe:CORRection:TSTandards[:STATe] OFF` to specify that only one set of standards will be used.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<boolean>` ON (1) - FORWARD direction of a 2-port calibration will be performed
  - OFF (0) - REVERSE direction of a 2-port calibration will be performed

Examples

- `SENS:CORR:SFOR 1`
- `sense2:correction:sforward:state 0`
- See an example using this command

Query Syntax

- `SENSe<cnum>:CORRection:SFORward[:STATe]?`

Return Type

- Boolean

Default

- ON
Applicable Models: All

(Read-Write) Turns error correction ON and OFF for the specified channel.

**Note:** Before using this command you must select a measurement using \CALCulate:MEASure:PARameter. You can select one measurement for each channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<ON | OFF>` **ON** (or 1) - correction is applied to the channel.
  - **OFF** (or 0) - correction is NOT applied to the channel.

**Examples**

```
SENS:CORR ON
sense2:correction:state off
```

**Query Syntax**

```
SENSe<cnum>:CORRection[:STATe]?
```

To query the error correction state for a measurement, use \CALC:MEAS:CORR:STATe?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

---

**SENSe<cnum>:CORRection:TCOLd:USER:VALue <num>**

Applicable Models: All

(Read-Write) Sets and returns the temperature of the noise source connector. Learn more about Noise Figure Calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<num>` Noise source temperature in Kelvin.

**Examples**

```
sense2:correction:tcold:user:value 298
```

See an example using this command

**Query Syntax**

```
SENSe<cnum>:CORRection:TCOLd:USER:VALue?
```

**Return Type**

Numeric

**Default**

Not Applicable
SENSe<cnum>:CORRection:TSTandards[:STATe] <boolean>

Applicable Models: All

(Read-Write) Specifies the acquisition of calibration data using ONE or TWO sets of standards.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1.
- <boolean>
  - ON (1) - TWO identical sets of standards will be used to simultaneously calibrate two ports (for both Forward and Reverse parameters).
  - OFF (0) - ONE set of standards will be used to perform a full 2-port calibration, one port at a time.

When specifying ON (use two sets), the SENS:CORR:COLL:ACQuire command uses the same standard index for each calibration class. To specify the calibration standard gender for each port, you must first ensure that the order of calibration class accurately reflects the configuration of your DUT. For example, for a DUT with a male connector on port 1 and a female connector on port 2, order the devices within the S11 classes (A, B, and C) such that the MALE standards are first in the list. Then order the S22 classes specifying the FEMALE standards as the first in the list.

Examples

```
SENS:CORR:TST 1
sense2:correction:tstandard:state 0
```

See an example using this command

Query Syntax

SENSe<cnum>:CORRection:TSTandards[:STATe]?

Return Type

Boolean

Default

ON

SENSe:CORRection:TYPE:CATalog? <char>
Applicable Models: All

(Read-Write) Lists the Cal Types in the VNA by either GUID or registered name. Learn more about applying Cal Type using SCPI.

**Note:** Before using this command you must select a measurement using `CALCulate:MEASure:PARameter`. You can select one measurement for each channel.

**Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;char&gt;</code></td>
<td>Specifies the type of list. Choose from:</td>
</tr>
<tr>
<td><strong>GUID</strong></td>
<td>the registered GUID of the Cal Type</td>
</tr>
<tr>
<td><strong>NAME</strong></td>
<td>the registered name of the Cal Type</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:CORR:TYPE:CAT? GUID
```

**Query Syntax**

```
SENSe<cnum>:CORRection:TYPE:CATalog? <char>
```

**Return Type**

Comma-separated string

**Default**

Not Applicable

---

**SENSe<cnum>:CORR:WAVE[:METHod] <param>**

**Applicable Models:** N522xB, N523xB, N524xB

(Write-Read) Sets and returns the wave correction method.

**Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Any existing channel number. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td><code>&lt;parm&gt;</code></td>
<td>Wave correction method. Choose from:</td>
</tr>
<tr>
<td><strong>ACTual</strong></td>
<td>Full error corrected actual waves at DUT plane.</td>
</tr>
<tr>
<td><strong>MATCH</strong></td>
<td>Matched corrected waves at DUT plane.</td>
</tr>
<tr>
<td><strong>RESPonse</strong></td>
<td>Response corrected wave at DUT plane.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS1:CORR:WAVE ACT
```

**Query Syntax**

```
SENSe<cnum>:CORR:WAVE[:METHod]? 
```

**Return Type**

String

**Default**

MATC
SENSe<cnum>::CORRrection::WAVE::NORM[:METHod] <param>

Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the wave normalization method.

Parameters

   <cnum>   Any existing channel number. If unspecified, value is set to 1.
   <parm>   Wave normalization method. Choose from:

       POWER - Waves are computed according to Kurokawa's power wave
               definition.

       TRAVeling – Waves are computed according to a traveling wave definition.

Examples

   SENS1::CORR::WAVE::NORM POW

Query Syntax

   SENS<channel>::CORR::WAVE::NORM[:METHod]?

Return Type

   String

Default

   POW
Manages the list of cal kits that are installed in the VNA.

```
SENSe:CORR:CKIT
  CLEar
  COUNT?
  ECAL
    | CHARacterize More commands
    | CLIS?t?
    | DMEMory
      | CLEar
      | IMPort
    | EXPort
    | INFormation
    | KNAme
      | INFormation
    | LIST?
    | ORlent?
    | PATH
      | COUNT?
      | DATA?
EXPort
IMPort
INITialize
LOAD
```

- Click on a red keyword to view the command details.
- Red is a superseded command
SENSe:CORRection:CKIT:CLEar[:IMMediate] [ckit]

Applicable Models: All

(Write-only) Deletes installed cal kits.

Parameters

[ckit] Optional String. Cal Kit to delete. If not specified, all VNA Cal kits are deleted, including custom kits.

Examples

SENSe:CORRection:CKIT:CLE
sense:correction:ckit:clear:immediate "85052B"

Query Syntax

Not Applicable

Default

Not Applicable

SENSe:CORRection:CKIT:COUNt?

Applicable Models: All

(Read-only) Returns the number of installed cal kits.

Examples

SENSe:CORRection:CKIT:COUNt?

Query Syntax

SENSe:CORRection:CKIT:COUNt?

Return Type

Numeric

Default

Not Applicable

SENSe:CORRection:CKIT:ECAL<mod>:CLIS?
Applicable Models: All

(Read-only) Returns a list of characterizations stored in the specified ECal module.

Parameters

<mod> ECal module from which to read user characterization numbers. Choose from 1 to 50. If unspecified, value is set to 1.

Examples

Module 1 contains User Characterizations 1 and 3.

SENSe:CORRection:CKIT:ECAL:CLIST?

'Returns the following (0 always indicates the factory characterization):

0,1,3

Return Type

Numeric list, separated by commas.

Default

Not Applicable

SENSe:CORRection:CKIT:ECAL:DMEMory:CLEar <kitName>

Applicable Models: All

(Write-only) Deletes user characterizations from VNA disk memory.

Parameters

<kitName> Optional String argument. ECal Model, User Characterization name + " ECal", and serial number of the ECal module, separated by spaces. See examples below.

If unspecified, ALL User Characterizations that are stored in VNA disk memory are deleted.

Examples

'These examples all use "MyUserChar" as the User characterization name.

'The "My User Char" characterization is deleted from disk memory.


'All User characterizations are deleted from disk memory.

Default

Not Applicable
SENSe:CORRection:CKIT:ECAL:DMEMory:IMPort <file>

Applicable Models: All

(Write-only) After the VNA disk memory is Exported to a file, use this command to Import the file into VNA disk memory, which allows the User Characterization to be used with the VNA and ECal module.

**Note:** An ECal confidence check can NOT be performed remotely from User Characterizations that are stored on the VNA disk.

**Parameters**

*<file>* String. Full path and file name of file that was exported.

**Examples**

```
SENS:CORR:CKIT:ECAL:DMEM:IMP "D:\myDiskUserChar.euc"
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

SENSe:CORRection:CKIT:ECAL:EXPort <kit>[,<file>][,<NewName>]]

Applicable Models: All

(Write-only) Saves an existing ECal characterization to a file. Use this command to archive the user characterization or to move the characterization to a different VNA for use with the specified ECal module. After exporting the user characterization, use **SENS:CORR:CKIT:ECAL:DMEM:IMPort** to make the user characterization available for use.

**Parameters**

*<kit>* String. Not case sensitive. ECal Model, User char name + " ECal", and serial number of the ECal module used for the characterization, separated by spaces. See examples below.

If the model and serial number of the module is not found, an error is returned.

*<file>]* Optional String argument. Path and filename of the user characterization. If not specified, the file is saved using characterization name + ".euc". If the path is not specified, it is stored in C:\ProgramData\Keysight\Network Analyzer\ECal User Characterizations\. The extension ".euc" is appended if one is not specified.

*<NewName>*] Optional String argument. This allows you to change the name for the User Characterization. When specified, the new name is saved in the file with the characterization. If unspecified, the existing user characterization name is
saved.

**Note:** If this argument is specified, the second argument (<file>) must also be specified.

### Examples

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>These examples all use &quot;MyUserChar&quot; as the User characterization name.</em></td>
<td>SENS:CORR:CKIT:ECAL:EXP &quot;N4433A MyUserChar ECal 00001&quot;,&quot;myUserChar.euc&quot;,&quot;NewUserChar&quot;</td>
</tr>
<tr>
<td><em>All parameters specified</em></td>
<td>sense:correction:ckit:ecal:export &quot;N4691B MyUserChar ECal 00500&quot;,&quot;myUserChar.euc&quot;</td>
</tr>
<tr>
<td><em>First two parameters are specified</em></td>
<td>SENS:CORR:CKIT:ECAL:EXP &quot;N4433A MyUserChar ECal 00001&quot;</td>
</tr>
<tr>
<td><em>Only first parameter is specified</em></td>
<td></td>
</tr>
</tbody>
</table>

### Query Syntax

**Default**

Not Applicable

**SENSe:CORRection:CKIT:ECAL<mod>:INFormation? [<char>]**

**Applicable Models:** All

*(Read-only)* Reads the identification and characterization information from the specified ECaI module.

**Note:** To read user-characterization information that is stored in VNA disk memory, then use SENS:CORR:CKIT:ECAL:KNAM:INF?

**Parameters**

- **<mod>**
  - ECaI module from which to read characterizations. Choose from 1 through 50. If unspecified, value is set to 1.
  - Do NOT assume the <mod> number is the order in which ECaI modules were connected.
  - Use SENS:CORR:CKIT:ECAL:LIST? to read a list of <mod> numbers of currently-attached ECaI modules.

- **<char>**
  - Optional argument. Specifies which characterization to read information from. If not specified, value is set to CHAR0.
Choose from:

- CHAR0  Factory characterization (data that was stored in the ECal module by Keysight)
- CHAR1  User characterization #1
- CHAR2  User characterization #2
- - through -
- CHAR12 User characterization #12

Examples

```

'Example return string:
"ModelNumber: 85092-60007, SerialNumber: 01386, ConnectorType: N5FN5F, PortAConnector: Type N (50) female, PortBConnector: Type N (50) female, MinFreq: 30000, MaxFreq: 9100000000, NumberOfPoints: 250, Calibrated: July 4 2002"
```

Return Type

Character

Default  Not Applicable

SENSe:CORRection:CKIT:ECAL:KNAMe:INFormation? <kitName>

Applicable Models: All

(Read-only) Reads the identification and characterization information from the specified ECal module or VNA disk memory.

Learn more about User Characterization in VNA Disk Memory.

Parameters

- <kitName>  String. ECal model and characterization to read information from, enclosed in quotes, in the following format:

  `<model> <name> ECal <serial number>`

Where:

- <model>: Always required
- <name>:

  - For the factory characterization, do not specify.
For a user-characterization stored in the module, use **User <n>** in the string, where <n> is the user-characterization number. Not case sensitive. Separate User and <n> with a space.

For a user-characterization stored in VNA disk memory, use **<charName>** from `SENS:CORR:CKIT:ECAL:CHAR:DMEM:SAVE <charName>`

**ECal** - not case sensitive

<serial number>: Optional. Include when two or more ECal modules with same model number are attached to the VNA,

Each item is separated with a space.

**Examples**

- 'For a factory characterization in module memory:

- 'For user characterization in module memory with optional serial number:

- 'For user characterization "foo" in disk memory:

- 'Example return string:
  "ModelNumber: N4433A, SerialNumber: 00028, ConnectorType: N5FN5F, PortAConnector: Type N (50) female, PortBConnector: Type N (50) female, MinFreq: 30000, MaxFreq: 9100000000, NumberOfPoints: 250, Calibrated: July 4 2002"

**Return Type**  String

**Default**  Not Applicable

`SENSe:CORRection:CKIT:ECAL:LIST?`
**Applicable Models:** All

*(Read-only)* Returns a list of index numbers for ECal modules that are currently attached to the VNA. Use these numbers (called `<mod>` in VNAHelp) to refer to the ECal module using SCPI commands.

### Examples

```scpi
SENS:CORR:CKIT:ECAL:LIST?

'If 2 modules are attached to the VNA
'then the returned list will be:
+1,+2

'If NO modules are attached to the VNA
'then the returned list will be:
+0
```

See example program using this command.

**Return Type** Numeric list, separated by commas.

**Default** Not Applicable

---

**SENSe<ch>:CORRection:CKIT:ECAL<n>:ORIent? <pnaPort>[,<charN>]**

**Applicable Models:** All

*(Read-only)* Returns the ECal port that is connected to the specified VNA port. A calibration does not have to be in process.

- `<ch>` Channel number that contains the frequency range to be calibrated.
- `<n>` ECal module number. Choose from 1 through 50.

If unspecified (only one ECal module is connected to the USB), `<n>` is set to 1. If two or more modules are connected, use `SENS:CORR:CKIT:ECAL:LIST?` to determine how many, and `SENS:CORR:CKIT:ECAL:INF?` to verify their identities.

- `<pnaPort>` VNA port number.
- `<charN>` Optional argument. If unspecified, factory data (CHAR0) is used. User Characterization number that matches the physical adapters/fixtures that are on the ECal module. This aids in determining the orientation of the ECal module. Choose from:

  - **CHAR0** Factory characterization (data that was stored in the ECal module by Keysight)
  - **CHAR1** User characterization #1
Beginning with A.08.33, up to 12 User Characterizations can be stored in a single ECal module. Previous releases allowed up to 5. Learn more.

Examples

sense2:correction:ckit,ecal1:orient? 2, char2

Return Type
The returned ECal port number is a 1-based number: 1 = Port A, 2 = Port B, 3 = Port C, 4 = Port D.

Zero (0) is returned when the auto-orientation routine is unable to resolve the orientation.

Default
Not Applicable

SENSe:CORRection:CKIT:ECAL<n>:PATH:COUNt? <path>

Applicable Models: All

(Read-only) Returns the number of unique states that exist for the specified path name on the selected ECal module.

This command performs exactly the same function as CONT:ECAL:MOD:PATH:COUNt?

Use the CONT:ECAL:MOD:PATH:STAT command to set the module into one of those states.

Use SENSe:CORR:CKIT:ECAL:PATH:DATA? to read the data for a state.

Parameters

<n> USB number of the ECal module. Choose from 1 to 50.

If unspecified (only one ECal module is connected to the USB), <n> is set to 1. If two or more modules are connected, use SENSe:CORR:CKIT:ECAL:LIST? to determine how many, and SENSe:CORR:CKIT:ECAL:INF? to verify their identities.

<path> Name of the path for which to read number of states. Choose from:

Reflection paths

• A
Transmission paths

- **AB**
- **AC** (4-port modules)
- **AD** (4-port modules)
- **BC** (4-port modules)
- **BD** (4-port modules)
- **CD** (4-port modules)

**Examples**

```
CONT:ECAL:MOD:PATH:COUNT?
control:ecal:module2:path:count?
```

**Return Type** Integer

**Default** Not Applicable


**Applicable Models:** All

(Read-only) Returns the data for a state from the memory of the selected ECal module. The returned data is interpolated if necessary to have the same stimulus values as the specified channel <ch>.

- For a reflection path state, the data is reflection S-parameter data. The number of values equals the number of stimulus points on the channel multiplied by 2 (because they are complex numbers).
- For a transmission path state, the data is all 4 S-parameters of the state. The number of values returned is 4 times that of a reflection state.

The data is returned in the same format as **CALC:MEAS:DATA:SNP?**

**Note:** This command returns SNP data without header information, and in columns, not in rows as .SnP files. This means that the data returned from this command sends all frequency data, then all Sx1 magnitude or real data, then all Sx1 phase or imaginary data, and so forth.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
<num> Optional argument. USB number of the ECal module. Choose from 1 through 50.

If unspecified (only one ECal module is connected to the USB), <num> is set to 1. If two or more modules are connected, use SENS:CORR:CKIT:ECAL:LIST? to determine how many, and SENS:CORR:CKIT:ECAL:INF? to verify their identities.

<path> Name of the path for which to read number of states. Choose from:

Reflection paths

- A
- B
- C (4-port modules)
- D (4-port modules)

Transmission paths

- AB
- AC (4-port modules)
- AD (4-port modules)
- BC (4-port modules)
- BD (4-port modules)
- CD (4-port modules)

<stateNum> Number of the state to set. Refer to the following table to associate the <stateNum> with a state in your ECal module.

In addition, CONT:ECAL:MOD:PATH:COUNT? returns the number of states in the specified ECal module.
<table>
<thead>
<tr>
<th>&lt;stateNum&gt;</th>
<th>N4432A and N4433A States</th>
<th>N4431A States</th>
<th>N469x and N755x States**</th>
<th>8509x States</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>One-Port Reflection States</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Open</td>
<td>Open</td>
<td>Impedance 1</td>
<td>Open</td>
</tr>
<tr>
<td>2</td>
<td>Short</td>
<td>Short</td>
<td>Impedance 2</td>
<td>Short</td>
</tr>
<tr>
<td>3</td>
<td>Impedance 1</td>
<td>Impedance 1</td>
<td>Impedance 3</td>
<td>Impedance 1</td>
</tr>
<tr>
<td>4</td>
<td>Impedance 2</td>
<td>Impedance 2</td>
<td>Impedance 4</td>
<td>Impedance 2</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Two-Port Transmission States</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Thru</td>
<td>Thru</td>
<td>Thru</td>
<td>Thru</td>
</tr>
<tr>
<td>2</td>
<td>Confidence</td>
<td>Confidence</td>
<td>Confidence</td>
<td>Confidence</td>
</tr>
</tbody>
</table>

** The following modules have only FOUR Impedance states (1, 2, 3, 4): N4690B, N4691B, N4692A, N4696B, N7550A - N7556A.

<char> Optional argument. Specifies which characterization within the ECal module to read information from. If not specified, value is set to CHAR0.

Choose from:

- **CHAR0** Factory characterization (data that was stored in the ECal module by Keysight)
- **CHAR1** User characterization #1
- **CHAR2** User characterization #2
- and so forth up to:
- **CHAR12** User characterization #12
### SENSE:CORRection:CKIT:EXPort <kit>[,<file>]

**Applicable Models:** All

(Write-only) Saves an existing cal kit definitions to a file. Use this command to archive or move a user-defined or modified cal kit to a different VNA. After exporting the cal kit, use SENSE:CORR:CKIT:IMPort to make the cal kit available for use on the VNA. This command provides the same behavior as the Installed Kits - Save As button on the Edit VNA Cal Kits dialog.

**Parameters**

- `<kit>`: String. Not case sensitive. Name of the cal kit to export, as seen in the Cal Kits field of the Select DUT Connectors and Cal Kits dialog of a SMART Cal.
- `<file>`: Optional String argument. Path and filename to where the Cal Kit file is to be saved. If not specified, the file is saved using `<kit> + ".ckt". If the path is not specified, it is stored in C:/Program Files/Keysight/Network Analyzer/PNACalKits/User.

**Examples**

- 'File unspecified
  
  SENS:CORR:CKIT:EXP "MyCalKit"
  
- Both parameters are specified
  
  sense:correction:ckit:export "MyCalKit","C:\myBackupCalKit.ckt"

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

### SENSE:CORRection:CKIT:IMPort <string>

---
Applicable Models: All

(Write-only) Imports the specified cal kit (.ckt file) and appends the imported kit to the end of the list of kits.

Note: Although there is no limit to the number of cal kits that can be imported, during an Unguided cal, you can access ONLY mechanical cal kits #1 through #95.

**Parameters**

| <string> | Path and cal kit name. |

**Examples**

SENSe:CORRection:CKIT:IMPort "c:\users\public\public documents\network analyzer\documents/85033D.ckt"

**Query Syntax**

Not Applicable

**Default**

Not Applicable

SENSe:CORRection:CKIT:INITialize[:IMMediate] [ckit]

Applicable Models: All

(Write-only) Restores default factory installed cal kits. This command also selects kit number 1, as you would using SENS:CORR:COLL:CKIT:SEL 1. Therefore, if you intend to work with a Cal Kit remotely, select the Cal Kit AFTER sending this command.

Note: This command can also delete all existing User-defined Cal Kits. However, if saved using Save As, these kits can be restored in the same manner as after a VNA firmware upgrade. Learn more about saving modified Cal Kits.

**Parameters**

| [ckit] | Optional String. Cal Kit to restore. If not specified, all VNA factory Cal kits are restored. |

**Examples**

SENSe:CORR:CKIT:INITialize

SENSe:CORRection:ckit:initialize:immediate "85052B"

**Query Syntax**

Not Applicable

**Default**

Not Applicable

SENSe:CORRection:CKIT:LOAD <string>
Applicable Models: All

(Write-only) Loads the specified collection of cal kits from a .wks file. You can make your own collection of cal kits from the Advanced Modify Cal Kit menu.

**Parameters**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;string&gt;</td>
<td>Path and file name of the cal kit collection.</td>
</tr>
</tbody>
</table>

**Examples**

```
sense:correction:ckit:load "C:\ProgramData\Keysight\Network Analyzer\PnaCalKits\factory\wMyCalKits.wks"
```

**Query Syntax** Not Applicable

**Default** Not Applicable
Sense:Correction:Collect:Ckit Commands

Use to change the definitions of calibration kit standards.

SENSe:CORRection:COLLect:CKIT:

| CATalog? |
| CLISl |
| CLABel |
| CONNector |
| ADD |
| CATalog? |
| DELeTe |
| FNAMe |
| SNAMe |
| DESCription |
| INFormation? |
| NAME |
| OLAB |
| OLIST? |
| ORDer |
| PORT[:SELect] |
| RESet |
| SELect |
| STANdard |
| CO, C1, C2, C3 |
| CHARacter |
| DATA |
| LOAD |
| DELay |
Click on a keyword to view the command details.

**Blue** keywords are superseded commands.

Most of these commands act on the currently selected standard from the currently selected calibration kit.

- To select a Calibration kit, use `SENS:CORR:COLL:CKIT:SEL`.
- See an example program that **CREATES a New Cal Kit**
- See an example program that **MODIFIES an Existing Cal Kit**
- Learn about Modifying Cal Kits
- Synchronizing the Analyzer and Controller
### SCPI Command Tree

**Note:** You should provide data for every definition field - for every standard in your calibration kit. If a field is not set, the default value may not be what you expect.

For more information, read *Specifying Calibration Standards and Kits for Keysight Vector Network Analyzers (Application Note 1287-11)*

---

**SENSe:CORRection:COLLect:CKIT:CATalog?**

**Applicable Models:** All

*(Read-only)* Returns the names of the first 95 mechanical cal kits in your VNA that can be used for unguided calibrations.

<table>
<thead>
<tr>
<th>Example</th>
<th>SENS:COR:COLL:CKIT:CAT?</th>
</tr>
</thead>
</table>

**Return Type** A comma-separated string

**Default** Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:CLISt <class>, <list_stds>**

**Applicable Models:** All

*(Write-Read)* Assigns the calibration class for the standard

**Parameters**

- `<class>` Specify the calibration classes:
  - **SA** - OPEN Standards (standards in the SA class are not always Opens)
  - **SB** - SHORT Standards
  - **SC** - LOAD Standards
  - **THRU** - Thru
  - **FWDT** - Forward Trans
  - **FWDM** - Forward Match
  - **REVT** - Reverse Trans
  - **REVM** - Reverse Match
**TRLT** - TRL Thru

**TRLR** - TRL Reflection

**TRLLm** - TRL Line

**UTHRu** - Unknown Thru

**ISOL** - Isolation

*<list_stds>* List of standard IDs (Same as **SENS:CORR:COLL:CKIT:OLIS**). A standard ID can be between 1 and 1000.

**Examples**

`SENS:CORR:COLL:CKIT:CLIST SA,1,2,5`

`SENS:CORR:COLL:CKIT:CLIST? SA`

**Return Type** Numeric; returns standard number of the selected class

**Default** Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:CLABel <class>, <label>**

**Applicable Models:** All

*(Write-Read)* Defines the label for the standard classes.

**Parameters**

- **<class>** Specify the calibration classes:
  - **SA** - OPEN Standards (standards in the SA class are not always Opens)
  - **SB** - SHORT Standards
  - **SC** - LOAD Standards
  - **TRAN** - Thru
  - **FWDT** - Forward Trans
  - **FWDM** - Forward Match
  - **REVT** - Reverse Trans
  - **REVM** - Reverse Match
  - **TRLT** - TRL Thru
  - **TRLR** - TRL Reflection
**TRLLm** - TRL Line

**UTHRu** - Unknown Thru

**ISOL** - Isolation

<label> Label of standard

### Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

**Return Type**  String; returns the label.

**Default**  Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:CONNeector:ADD**

<fami>le>,<start>,<stop>,<z0>,<gender>,<media>,<cutoff>

**Applicable Models:** All

*(Write only)* Creates a new connector. The connector is automatically added to the list of available connectors for the currently selected cal kit. If a connector includes both male and female connectors, each connector must be added separately.

**Parameters**

- `<family>`  (String) Name of connector family. Limited to 50 characters.
- `<start>` Start frequency
- `<stop>` Stop frequency
- `<z0>` Characteristic Impedance of the connector in ohms.
- `<gender>` Connector gender. Choose from:
  - MALE
  - FEMALE
  - NONE
- `<media>` Media of the connector. Choose from:
  - COAX - coaxial
  - WAVE - waveguide
- `<cutoff>` Cutoff frequency of the connector (waveguide only).

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:COLL:CKIT:CONN:ADD &quot;PSC 1.8 mm&quot;,0 HZ,999.9 GHZ,50,FEMALE,COAX,0.0</td>
<td></td>
</tr>
<tr>
<td>SENS:CORR:COLL:CKIT:CONN:ADD &quot;PSC 1.8 mm&quot;,0 HZ,999.9 GHZ,50,MALE,COAX,0.0</td>
<td></td>
</tr>
</tbody>
</table>

**Query Syntax**  Not applicable
SENSe:CORRection:COLLect:CKIT:CONNector:CATalog?

Applicable Models: All

(Read-only) Returns a comma-separated list of all connectors defined within the currently selected cal kit. The returned string includes the connector family name followed by the connector gender, if any. Kits may include a primary connector family name and additional connector family names.

Connector family names are case sensitive. A connector family named "PSC 2.4" is different from a connector family named "psc 2.4".

Learn more about Connector Family Name.

Examples

```
'Returned string
"Type-N (50) male, Type-N (50) female"
```

Default Not Applicable

---

SENSe:CORRction:COLLect:CKIT:CONNector:DELeete

Applicable Models: All

(Write-only) Deletes the primary connector family name from the selected kit. The VNA allows multiple connector families for each kit. If a kit includes multiple connector families, only the first listed (primary) connector family name is deleted.

Once the connector family is deleted, the connector may not be assigned to any new or existing standard within the kit.

The previously defined standards retain their association to the deleted connector name. To reassign standards to a new connector family name, use SENS:CORR:COLL:CKIT:CONN:SNAMe.

Examples

```
SENS:CORR:COLL:CKIT:CONN:DEL
```

Query Syntax Not Applicable

Default Not Applicable

---

SENSe:CORRrection:COLLect:CKIT:CONNector:FNAMe <name>
Applicable Models: All

(Read-Write) Replaces the primary connector family name from the selected kit with a new connector family name. The connector family name is replaced in all standards in the kit that share that name. The VNA allows multiple connector families for each kit. If a kit includes multiple connector families, only the first listed (primary) connector family name is replaced. Use the query form of this command to return the name of the primary connector family.

Parameters

<name>  New connector family name. Limited to 50 characters.

Examples

SENS:CORR:COLL:CKIT:CONN:FNAM 'MYPSC35'

Sense:correction:collect:ckit:connector:name 'My Type N'

Query Syntax

SENSe:CORRection:COLLect:CKIT:CONNector:FNAMe?

Return Type

String

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:CONNector:SNAMe <family>,<gender>,<port>

Applicable Models: All

(Read-Write) Assigns a family name to the currently selected standard from the currently selected kit. Specify each port of a 2-port standard individually. Use the query form of this command to read the connector family name assigned to the current standard. The name is not assigned unless the connector family name is previously defined within the selected kit.

Parameters

?family>  String. Connector family name.

<gender>  Connector gender. Choose from:

MALE

FEMALE

NONE

<port>  Number of the connector port to be assigned the connector family name. 2-port standards such as a thru line must be assigned separately. It is not relevant which connector is port 1 or port 2.

1  Specifies a 1-port standard or the first port of a 2-port standard.

2  Specifies the second port of a 2-port standard.

Examples

SENS:CORR:COLL:CKIT:CONN:SNAME "Type-N (50)",MALE,1
SENSe:CORRection:COLLect:CKIT:DESCription <string>

Applicable Models: All

(Read-Write) Modifies the cal kit description field of the selected kit. This description appears in the Edit VNA Cal Kit dialog box.

Parameters

<string> Description of the cal kit. Limited to 50 characters.

Examples

```
SENSe:CORRection:COLLect:CKIT:DESC "My New CalKit"
```

Query Syntax

SENSe:CORRection:COLLect:CKIT:DESCription?

Return Type

String

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:INFormation? <module>[,char]

Applicable Models: All

(Read Only) Reads characterization information from an ECal module.

Parameters

<string> Specifies which ECal module to read from. Choose from:

- **ECAL1**
- through.

- **ECAL50**

[char] Optional argument.

Specifies which characterization within the ECal module to read information from. If this argument is not used, the default is **CHAR0**. **CHAR1** through **CHAR5** are for user characterizations that may have been written to the module by the User Characterization feature on the VNA. Choose from:

- **CHAR0** Factory characterization (data that was stored in the ECal module by
SENSe:CORRection:COLLect:CKIT:NAME <name>

**Applicable Models:** All

*(Read-Write)* Sets a name for the selected calibration kit.

**Parameters**

- **<name>**  
  Calibration Kit name. Any string name, can include numerics, period, and spaces; any length (although the dialog box display is limited to about 30 characters).

**Examples**

```
SENSe:CORRection:COLLect:CKIT:NAME 'MYAPC35'
SENSe:correction:collect:ckit:name 'mytypen'
```

**Query Syntax**

SENSe:CORRection:COLLect:CKIT:NAME?

**Return Type**  
String

**Default**  
Not Applicable

---

SENSe:CORRection:COLLect:CKIT:OLAbel<class> <name> - **Superseded**
This command is replaced by `SENS:COR:CKIT:CLAB`.

**Applicable Models:** All

*(Read-Write)* Sets the label for the calibration class designated by `<class>`. The label is used in the prompts for connecting the calibration standards associated with that `<class>`.

**Parameters**

- `<class>` Number of the calibration class. Choose a number between: 1 and 18. The `<class>` numbers are associated with the following calibration Classes:

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td></td>
</tr>
<tr>
<td>1 SA</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>2 SB</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>3 SC</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>4 FWD TRANS</td>
<td>Thru/Delay standard</td>
</tr>
<tr>
<td>Port 2</td>
<td></td>
</tr>
<tr>
<td>5 SA</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>6 SB</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>7 SC</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>8 REV TRANS</td>
<td>Thru/Delay standard</td>
</tr>
</tbody>
</table>

**3-port analyzers only**

<table>
<thead>
<tr>
<th>Port 3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>9 S33A</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>10 S33B</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>11 S33C</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>12 S32T</td>
<td>Thru/Delay standard</td>
</tr>
<tr>
<td>13 S23T</td>
<td>Thru/Delay standard</td>
</tr>
<tr>
<td>14 S31T</td>
<td>Thru/Delay standard</td>
</tr>
<tr>
<td>15 S13T</td>
<td>Thru/Delay standard</td>
</tr>
</tbody>
</table>

**TRL Calibrations**

<table>
<thead>
<tr>
<th>TRL Calibrations</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>16 TRL &quot;T&quot;</td>
<td>Thru standard</td>
</tr>
<tr>
<td>17 TRL &quot;R&quot;</td>
<td>Reflect standard</td>
</tr>
<tr>
<td>18 TRL &quot;L&quot;</td>
<td>Line standard</td>
</tr>
</tbody>
</table>
<name>  Label for the calibration class. Must be enclosed in quotes. Any string between 1 and 12 characters long. Cannot begin with a numeric.

Examples
SENS:CORR:COLL:CKIT:OLAB3 'LOADS'
sense:correction:collect:ckit:olabel4 'Thru'

Return Type  String
Default  Not Applicable

SENSe:CORRection:COLLect:CKIT:OLiSt[class]? - Superseded
This command is replaced by SENS:CORR:CKIT:CLIS.

Applicable Models: All

(Read-only) Returns seven values of standards that are assigned to the specified class.
This command ALWAYS applies to the Cal Kit that is selected (using SENS:CORR:COLL:CKIT:SEL) when this ORDer command is sent.

Parameters

<class>  Number of the calibration class to be queried. The <class> numbers are associated with the following calibration Classes:

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>SA</td>
</tr>
<tr>
<td>2</td>
<td>SB</td>
</tr>
<tr>
<td>3</td>
<td>SC</td>
</tr>
<tr>
<td>4</td>
<td>FWD TRANS</td>
</tr>
<tr>
<td>Port 2</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>SA</td>
</tr>
<tr>
<td>6</td>
<td>SB</td>
</tr>
<tr>
<td>7</td>
<td>SC</td>
</tr>
<tr>
<td>8</td>
<td>REV TRANS</td>
</tr>
</tbody>
</table>

3-port analyzers ONLY (N3381A/2A/3A)

4-port analyzers use S11 and S22 classes (see example program)

Port 3

<p>| 9       | S33A                   | Reflection standard |
| 10      | S33B                   | Reflection standard |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Reflection standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>S33C</td>
<td>Through/Delay standard</td>
</tr>
<tr>
<td>12</td>
<td>S32T</td>
<td>Through/Delay standard</td>
</tr>
<tr>
<td>13</td>
<td>S23T</td>
<td>Through/Delay standard</td>
</tr>
<tr>
<td>14</td>
<td>S31T</td>
<td>Through/Delay standard</td>
</tr>
<tr>
<td>15</td>
<td>S13T</td>
<td>Through/Delay standard</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Thru standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>TRL &quot;T&quot;</td>
<td>Through standard</td>
</tr>
<tr>
<td>17</td>
<td>TRL &quot;R&quot;</td>
<td>Reflect standard</td>
</tr>
<tr>
<td>18</td>
<td>TRL &quot;L&quot;</td>
<td>Line standard</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:CORR:COLL:CKIT:OLIS?
```

Always returns 7 standard numbers. Unassigned standards return 0

**Return Type**

Numeric; returns the `<class>` number of the selected standard.

**Default**

Not Applicable

---


This command is replaced by **SENS:CORR:CKIT:CLIS**.

**Applicable Models:** All

*(Read-Write)* Sets a standard number to a calibration class. This command does **NOT** set or dictate the order for measuring the standards. For more information, see Assigning Standards to a Calibration Class.

This command **ALWAYS** applies to the Cal Kit that is selected (using **SENS:CORR:COLL:CKIT:SEL**) when this **ORDer** command is sent.

**Parameters**

`<class> ` Number of the calibration class that is assigned to `<standard>`. Choose a number between: **1** and **18**. The `<class>` numbers are associated with the following calibration Classes:
<table>
<thead>
<tr>
<th>Port</th>
<th>Class</th>
<th>Description</th>
<th>STAN#</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SA</td>
<td>Reflection standard</td>
<td>STAN1</td>
</tr>
<tr>
<td>2</td>
<td>SB</td>
<td>Reflection standard</td>
<td>STAN2</td>
</tr>
<tr>
<td>3</td>
<td>SC</td>
<td>Reflection standard</td>
<td>STAN3</td>
</tr>
<tr>
<td>4</td>
<td>FWD TRANS</td>
<td>Thru/Delay standard</td>
<td>STAN4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>SA</td>
<td>Reflection standard</td>
<td>STAN1</td>
</tr>
<tr>
<td>6</td>
<td>SB</td>
<td>Reflection standard</td>
<td>STAN2</td>
</tr>
<tr>
<td>7</td>
<td>SC</td>
<td>Reflection standard</td>
<td>STAN3</td>
</tr>
<tr>
<td>8</td>
<td>REV TRANS</td>
<td>Thru/Delay standard</td>
<td>STAN4</td>
</tr>
</tbody>
</table>

**3-port analyzers ONLY (N3381A/2A/3A)**

4-port analyzers use S11 and S22 classes *(see example program)*

<table>
<thead>
<tr>
<th>Port</th>
<th>Class</th>
<th>Description</th>
<th>STAN#</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>S33A</td>
<td>Reflection standard</td>
<td>STAN1</td>
</tr>
<tr>
<td>10</td>
<td>S33B</td>
<td>Reflection standard</td>
<td>STAN2</td>
</tr>
<tr>
<td>11</td>
<td>S33C</td>
<td>Reflection standard</td>
<td>STAN3</td>
</tr>
<tr>
<td>12</td>
<td>S32T</td>
<td>Thru/Delay standard</td>
<td>STAN4</td>
</tr>
<tr>
<td>13</td>
<td>S23T</td>
<td>Thru/Delay standard</td>
<td>STAN4</td>
</tr>
<tr>
<td>14</td>
<td>S31T</td>
<td>Thru/Delay standard</td>
<td>STAN4</td>
</tr>
<tr>
<td>15</td>
<td>S13T</td>
<td>Thru/Delay standard</td>
<td>STAN4</td>
</tr>
</tbody>
</table>

**TRL Calibration**

<table>
<thead>
<tr>
<th>Port</th>
<th>Class</th>
<th>Description</th>
<th>STAN#</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>TRL &quot;T&quot;</td>
<td>Thru standard</td>
<td>STAN4</td>
</tr>
<tr>
<td>17</td>
<td>TRL &quot;R&quot;</td>
<td>Reflect standard</td>
<td>STAN1</td>
</tr>
<tr>
<td>18</td>
<td>TRL &quot;L&quot;</td>
<td>Line standard</td>
<td>STAN3</td>
</tr>
</tbody>
</table>
**Examples**

<table>
<thead>
<tr>
<th>Standard number to be assigned to the class; Choose a standard between 1 and 30. One standard is mandatory; up to six additional standards are optional.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 1: Assigns standard 3 to S11A class:</td>
</tr>
<tr>
<td>SENS:CORR:COLL:CKIT:ORD1 3</td>
</tr>
<tr>
<td>Example 2: Assigns standard 2 and 5 to S21T class:</td>
</tr>
<tr>
<td>sense:correction:collect:ckit:order4 2,5</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe:CORRection:COLLect:CKIT:ORDer<class>?


**Return Type** Numeric

**Default** Not Applicable

---

**SENSe<cnum>:CORRection:COLLect:CKIT:PORT<n>[:SELe]ct <string>**

**Applicable Models:** All

(Read-Write) Sets and returns the name of the Cal Kit to use for Unguided cal.

This command effectively does the same task as SENS:CORR:COLL:CKIT but specifies the cal kit by name.

**Note:** During an Unguided cal, you can access ONLY mechanical cal kits #1 through #95. However, there is no limit to the number of cal kits that can be imported.

**Parameters**

- `<cnum>`: Currently not used. The unguided cal kit selection is for all ports on all channels.
- `<n>`: Currently not used. The unguided cal kit selection is for all ports on all channels.
- `<string>`: Cal Kit name enclosed in quotes. Use SENS:CORR:COLL:CKIT:CAT? to read a list of all available Cal Kits in the VNA.

**Examples**

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:COLL:CKIT:PORT &quot;85052B&quot;</td>
</tr>
<tr>
<td>sense2:correction:collect:ckit:port:select &quot;85052D&quot;</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe<cnum>:CORRection:COLLect:CKIT:PORT<n>[:SELe]ct?

**Return Type** String

**Default** Last kit selected
SENSe:CORRection:COLLect:CKIT:RESet <num> - Superseded

Applicable Models: All

This command is replaced by Sens:Corr:CKit:Init.

(Write-only) Resets the selected calibration kit to factory default definition values.

Parameters

<num> The number of the calibration kit to be reset. Choose any integer between: 1 and 8

Examples

SENS:CORR:COLL:CKIT:RESet 1
sense:correction:collect:ckit:reset 4

Query Syntax

Not Applicable

Default

Not Applicable

SENSe<cnun>:CORREction:COLLect:CKIT[:SElect] <num>

Applicable Models: All

(Read-Write) Selects (makes active) a calibration kit for performing an UNGUIDED calibration or for modifying standards. All subsequent "CKIT" commands that are sent apply to this selected calibration kit. Select a calibration standard using SENS:CORR:COLL:CKIT:STAN <num>. Kits 1 to approximately kit 37 are factory installed Cal Kits.

Note: During an Unguided cal, you can access ONLY mechanical cal kits #1 through #95. However, there is no limit to the number of cal kits that can be imported.

This command effectively does the same task as SENS:CORR:COLL:CKIT:PORT which specifies the cal kit by name instead of this command which specifies by number.

Parameters

<cnun> Any existing channel number; if unspecified, value is set to 1.

<num> The number of the calibration kit. Choose from:

Use SENSe:CORRection:COLLect:CKIT:RESet to restore Cal Kits to default values.

Name

1 Cal Kit 1
2 Cal Kit 2
3 Cal Kit 3
Cal Kit 94
Cal Kit 95
ECal module

Note: Always check the list of available cal kits using SENSE:CORRection:COLLect:CKIT:CATalog? to ensure that the correct cal kit is selected.

Examples
SENS:CORR:COLL:CKIT 2
sense2:correction:collect:ckit:select 7

Query Syntax
SENSe<cnum>:CORRection:COLLect:CKIT?

Return Type
Numeric

Default
Last kit selected

SENSe:CORRection:COLLect:CKIT:STANdard:C0 <num>

Applicable Models: All

(Read-Write) Sets the C0 value (the first capacitance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Keysight.com.

Parameters

<num> Value for C0 in femtofarads (1E-15)

Examples
The following commands set C0=15 femtofarads:

SENS:CORR:COLL:CKIT:STAN:d:C0 15
sense:correction:collect:ckit:standard:c0 15

Query Syntax
SENSe:CORRection:COLLect:CKIT:STANdard:C0?

Return Type
Numeric

Default
Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:C1 <num>
Applicable Models: All

(Read-Write) Sets the C1 value (the second capacitance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Keysight.com.

Parameters
<num> Value for C1.

Examples
The following two commands set C1=15:

SENSe:CORRection:COLLect:CKIT:STANdard:C1 15
sense:correction:collect:ckit:standard:c1 15

Query Syntax
SENSe:CORRection:COLLect:CKIT:STANdard:C1?

Return Type
Numeric

Default
Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:C2 <num>

Applicable Models: All

(Read-Write) Sets the C2 value (the third capacitance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Keysight.com.

Parameters
<num> Value for C2.

Examples
The following two commands set C2:

SENSe:CORRection:COLLect:CKIT:STANdard:C2 15
sense:correction:collect:ckit:standard:c2 15

Query Syntax
SENSe:CORRection:COLLect:CKIT:STANdard:C2?

Return Type
Numeric

Default
Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:C3 <num>
Applicable Models: All

(Read-Write) Sets the C3 value (the fourth capacitance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Keysight.com.

**Parameters**

| <num> | Value for C3. |

**Examples**
The following two commands set C3

```
SENS:CORR:COLL:CKIT:STAN:C3 15
sense:correction:collect:ckit:standard:c3 15
```

**Query Syntax**
SENSe:CORRection:COLLect:CKIT:STANdard:C3?

**Return Type**
Numeric

**Default**
Not Applicable

---

SENSe:CORRection:COLLect:CKIT:STANdard:CHARacter <char>

Applicable Models: All

(Read-Write) Sets the media type of the selected calibration standard.

**Parameters**

| <char> | Media type of the standard. Choose from: Coax - Coaxial Cable Wave - Waveguide |

**Examples**
SENSe:CORRection:COLLect:CKIT:STANdard:CHARacter COAX
SENSe:CORRection:COLLect:CKIT:STANdard:CHARacter WAVE

**Query Syntax**
SENSe:CORRection:COLLect:CKIT:STANdard:CHARacter?

**Return Type**
Numeric

**Default**
Coax

---

SENSe:CORRection:COLLect:CKIT:STANdard:DATA:LOAD <string>
**Applicable Models:** All

*(Write-only)* Loads response (and optionally accuracy) data from a file for the selected database standard. Accuracy data is loaded if the file being loaded contains accuracy data. [Learn more.](#)

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;string&gt;</td>
<td>Path and filename.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:CORR:COLL:CKIT:STAN:DATA:LOAD "c:\users\public\network\ analyzer\database_file.dat"
```

**Default** Not Applicable

---

**SENSe:CORRRection:COLLect:CKIT:STANdard:DELay <num>**

**Applicable Models:** All

*(Read-Write)* Sets the electrical delay value for the selected standard.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;num&gt;</td>
<td>Electrical delay in picoseconds</td>
</tr>
</tbody>
</table>

**Examples**

```
The following two commands set delay to 50 picoseconds
SENS:CORR:COLL:CKIT:STAN:DEL 50e-12
sense2:correction:collect:ckit:standard:delay 50ps
```

**Query Syntax**

```
SENSe:CORRection:COLLect:CKIT:STANdard:DELay?
```

**Return Type** Numeric

**Default** Not Applicable

---

**SENSe:CORRRection:COLLect:CKIT:STANdard:FMAXimum <num>**

**Applicable Models:** All

*(Read-Write)* Sets the maximum frequency for the selected standard.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;num&gt;</td>
<td>Maximum frequency in Hertz.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:CORR:COLL:CKIT:STAN:FMAX 9e9
sense:correction:collect:ckit:standard:fmaximum 9Ghz
```

**Query Syntax**

```
SENSe:CORRection:COLLect:CKIT:STANdard:FMAXimum?
```

**Return Type** Numeric

**Default** Not Applicable
SENSe:CORRection:COLLect:CKIT:STANdard:FMINimum <num>

Applicable Models: All

(Read-Write) Sets the minimum frequency for the selected standard.

**Parameters**

<num> Minimum frequency in Hertz.

**Examples**

SENSe:CORR:CORR:COLL:CKIT:STAN:FMIN 1e3
sense:correction:collect:ckit:standard:fminimum 1khz

**Query Syntax**

SENSe:CORRection:COLLect:CKIT:STANdard:FMINimum?

**Return Type**

Numeric

**Default**

Not Applicable

---

SENSe:CORRection:COLLect:CKIT:STANdard:IMPedance <num>

Applicable Models: All

(Read-Write) Sets the characteristic impedance for the selected standard.

**Parameters**

<num> Impedance in Ohms

**Examples**

SENSe:CORR:CORR:COLL:CKIT:STAN:IMP 75
sense:correction:collect:ckit:standard:impedance 50.3

**Query Syntax**

SENSe:CORRection:COLLect:CKIT:STANdard:IMPedance?

**Return Type**

Numeric

**Default**

50

---

SENSe:CORRection:COLLect:CKIT:STANdard:L0 <num>
### SENSE:CORRection:COLLect:CKIT:STANdard:L0 <num>

**Applicable Models:** All

*(Read-Write)* Sets the L0 value (the first inductance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Keysight.com.

**Parameters**

- `<num>` Value for L0 in femtohenries (1E-15)

**Examples**

The following two commands set L0=15 femtohenries:

```plaintext
SENS:CORR:COLL:CKIT:STAN:L0 15
sense:correction:collect:ckit:standard:l0 15
```

**Query Syntax**

```
SENSe:CORRection:COLLect:CKIT:STANdard:L0?
```

**Return Type** Numeric

**Default** Not Applicable

### SENSE:CORRection:COLLect:CKIT:STANdard:L1 <num>

**Applicable Models:** All

*(Read-Write)* Sets the L1 value (the second inductance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Keysight.com.

**Parameters**

- `<num>` Value for L1.

**Examples**

The following two commands set L1=15:

```plaintext
SENS:CORR:COLL:CKIT:STAN:L1 15
sense:correction:collect:ckit:standard:l1 15
```

**Query Syntax**

```
SENSe:CORRection:COLLect:CKIT:STANdard:L1?
```

**Return Type** Numeric

**Default** Not Applicable

### SENSE:CORRection:COLLect:CKIT:STANdard:L2 <num>

- **4096**
Applicable Models: All

(Read-Write) Sets the L2 value (the third inductance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Keysight.com.

Parameters

<num> Value for L2.

Examples

The following two commands set L2=15:

```
SENS:CORR:COLL:CKIT:STAN:L2 15
sense:correction:collect:ckit:standard:l2 15
```

Query Syntax

SENSe:CORRection:COLLect:CKIT:STANdard:L2?

Return Type

Numeric

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:L3 <num>

Applicable Models: All

(Read-Write) Sets the L3 value (the fourth inductance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Keysight.com.

Parameters

<num> Value for L3.

Examples

The following two commands set L3=15:

```
SENS:CORR:COLL:CKIT:STAN:L3 15
sense:correction:collect:ckit:standard:l3 15
```

Query Syntax

SENSe:CORRection:COLLect:CKIT:STANdard:L3?

Return Type

Numeric

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:LABel <name>
Applicable Models: All

(Read-Write) Sets the label for the selected standard. The label is used to prompt the user to connect the specified standard.

Parameters

- `<name>`: Label for the standard; Must be enclosed in quotes. Any string between 1 and 12 characters long. Cannot begin with a numeric.

Examples

```
SENS:CORR:COLL:CKIT:STAN:LAB 'OPEN'
sense:correction:collect:ckit:standard:label 'Short2'
```

Query Syntax

```
SENSe:CORRection:COLLect:CKIT:STANdard:LABel?
```

Return Type: String

Default: Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:LOSS <num>

Applicable Models: All

(Read-Write) Sets the insertion loss for the selected standard.

Parameters

- `<num>`: Insertion loss in Gohms / sec. (GigaOhms per second of electrical delay)

Examples

```
SENS:CORR:COLL:CKIT:STAN:LOSS 3.5e9
sense:correction:collect:ckit:standard:loss 3
```

Query Syntax

```
SENSe:CORRection:COLLect:CKIT:STANdard:LOSS?
```

Return Type: Numeric

Default: Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:REMove

Applicable Models: All

(Write only) Deletes the selected standard from the selected cal kit.

Examples

```
SENS:CORR:COLL:CKIT:STAN:REMove
```

Default: Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:SDEScription <string>
Applicable Models: All

(Read-Write) Modifies the description of the selected standard of the selected kit. This description appears in the edit kit dialog box.

**Parameters**

- `<string>` Description of the standard.

**Examples**


**Query Syntax**

SENS:CORR:COLL:CKIT:STAN:SDESRIPTION?

**Return Type**

String

**Default**

Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:STANdard[:SELECT] <num>**

Applicable Models: All

(Read-Write) Selects the calibration standard. All subsequent "CKIT" commands to modify a standard will apply to the selected standard. Select a calibration kit using SENS:CORR:COLL:CKIT:SEL

**Parameters**

- `<num>` Number of the standard. Choose any number between: 1 and 1000

**Examples**

SENS:CORR:COLL:CKIT:STAN 3
sense:correction:collect:ckit:standard:select 8

**Query Syntax**

SENSe:CORRection:COLLect:CKIT:STANdard[:SELect]?

**Return Type**

Numeric

**Default**

1

---

**SENSe:CORRection:COLLect:CKIT:STANdard:TYPE <char>**
Applicable Models: All

(Read-Write) Sets the type for the selected standard.

Parameters

<char>  Choose from:
OPEN
SHORT
LOAD
SLOAD (sliding load)
THRU (through)
ARBI (arbitrary)
DATabased (data-based)

Examples

SENS:CORR:COLL:CKIT:STAN:TYPE LOAD
sense:correction:collect:ckit:standard:type short

Query Syntax

SENSe:CORRection:COLLect:CKIT:STANdard:TYPE?

Return Type

Character

Default  Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:TZReal <num>

Applicable Models: All

(Read-Write) Sets the TZReal component value of the Terminal Impedance for the selected standard.

Note: Only applicable when the Standard Type is set to ARBI

Parameters

<num>  Value for TZReal in Ohms

Examples

The following commands set TZReal=15 Ohms:

SENS:CORR:COLL:CKIT:STAN:TZReal 15
sense:correction:collect:ckit:standard:TZReal 15

Query Syntax

SENSe:CORRection:COLLect:CKIT:STANdard:TZReal?

Return Type

Numeric

Default  Not Applicable
**SENSe:CORRection:COLLect:CKIT:STANdard:TZImag <num>**

**Applicable Models:** All

**(Read-Write)** Sets the TZImag component value of the Terminal Impedance for the selected standard.

**Note:** Only applicable when the Standard Type is set to **ARBI**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;num&gt;</td>
<td>Value for TZImag in Ohms</td>
</tr>
</tbody>
</table>

**Examples**
The following two commands set TZImag=15 Ohms:

```plaintext
SENS:CORR:COLL:CKIT:STAN:TZImag 15
sense:correction:collect:ckit:standard:TZImag 15
```

**Query Syntax**

```
SENSe:CORRection:COLLect:CKIT:STANdard:TZImag?
```

**Return Type**

Numeric

**Default**

Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:TRLoption:IMPedance <char>**

**Applicable Models:** All

**(Read-Write)** Sets the reference impedance when using this TRL cal kit. Learn more.

Before sending this command, select a cal kit using **SENSe:CORRe:COLLect:CKIT:SELect**.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;char&gt;</td>
<td>Choose from:</td>
</tr>
</tbody>
</table>

**SYSTem** - The system impedance is used as the reference impedance. During a Guided or Unguided Cal, the Z0 of the Cal standard's connector definition sets the System Z0.

Make this selection when the desired test port impedance differs from the impedance of the LINE standard. Also, make this selection when skin effect impedance correction is desired for coax lines.

**LINE** The impedance of the line standard is used as the reference impedance, or center of the Smith Chart. Any reflection from the line standard is assumed to be part of the directivity error.

**Examples**

```
SENS:CORR:COLL:CKIT:TRL:IMP SYST
sense:correction:collect:ckit:TRLoption:impedance line
```
### SENSe:CORRection:COLLect:CKIT:TRLoption:LRLChar <bool>

**Applicable Models:** All

(Read-Write) This setting ONLY applies if an LRL Cal Kit is being modified AND Testport Reference Plane is set to THRU AND the TRL Thru class standard and the TRL Line/Match class standard both have the same values for Offset Z0 and Loss. Otherwise, this setting is ignored.

Before sending this command, select a cal kit using `SENS:CORR:COLL:CKIT:SELect`.

**Parameters**

<bool> Choose from:

- **1** or **ON** - Automatically correct for line loss and dispersion characteristics.
- **0** or **OFF** - Select when anomalies appear during a calibrated measurement which may indicate different loss and impedance values for the Line standards.

**Examples**

```plaintext
SENS:CORR:COLL:CKIT:TRL:LRLC 1
SENSe:CORrection:COLLect:CKIT:TRLoption:LRLChar off
```

### Query Syntax

SENSe:CORRection:COLLect:CKIT:TRLoption:LRLChar?

**Return Type** Boolean

**Default** OFF

---

### SENSe:CORRection:COLLect:CKIT:TRLoption:RPLane <char>

---

4102
Applicable Models: All

(Read-Write) Sets the reference impedance when using this cal kit. Learn more.

Before sending this command, select a cal kit using SENS:CORR:COLL:CKIT:SELect.

Parameters

<char> Choose from:

**THRU**  The THRU standard definition is used to establish the measurement reference plane. Select if the THRU standard is zero-length or very short.

**REFLect**  The REFLECT standard definition is used to establish the position of the measurement reference plane. Select if the THRU standard is not appropriate AND the delay of the REFLECT standard is well defined. Also, select If a flush short is used for the REFLECT standard because a flush short provides a more accurate phase reference than a Thru standard.

Examples

SENS:CORR:COLL:CKIT:TRL:RPL THRU

SENSe:CORRection:COLLect:CKIT:TRLoption:RPLane?

Return Type  Character

Default  THRU
Perform actions on calibration sets.

<table>
<thead>
<tr>
<th>SENSE:CORRection:CSET</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTivate</td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>COPY</td>
</tr>
<tr>
<td>CREate</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>DATA</td>
</tr>
<tr>
<td>DEAcivate</td>
</tr>
<tr>
<td>DELeete</td>
</tr>
<tr>
<td>DESCription</td>
</tr>
<tr>
<td>ETERm</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>FLATten</td>
</tr>
<tr>
<td>GENerate</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>GUID</td>
</tr>
<tr>
<td>ITEM</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>SAVE</td>
</tr>
<tr>
<td>[SELect]</td>
</tr>
</tbody>
</table>
SENSe<cnum>:CORRection:CSET:ACTivate <string>, <bool>

**Applicable Models:** All

This command replaces **SENS:CORR:CSET:GUID**

*(Read-Write)* Selects and applies a Cal Set to the specified channel.

Use **SENS:CORR:CSET:CAT?** to list the Cal Sets.

**Parameters**

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<string>`  Cal Set to make active. Specify the Cal Set by **GUID** or **Name**. Use **SENS:CORR:CSET:CAT?** to list the available Cal Sets in either format.
- `<bool>`  Should the Cal Set stimulus values be applied to the channel. Choose from:
ON (1) Apply the Cal Set stimulus values to the channel.

OFF (0) Do NOT apply the Cal Set stimulus values. If the Cal Set stimulus values do not match the channel stimulus values, then the following will occur:

- If interpolation is ON, then interpolation will be attempted. This may fail if the channel frequency is outside the range of the Cal Set.
- If interpolation is OFF, the selection will be abandoned and an error is returned:

### Examples

```plaintext
SENS:CORR:CSET:ACT "My2Port",1
```

```plaintext
sense:correction:cset:activate? name
'returns
"My2Port"
```

### Query Syntax

SENSe<num>:CORRection:CSET:ACTivate? [GUID|NAME]

Returns the name of the Cal Set that is applied to the specified channel. Choose from GUID or NAME to specify which string is returned. If unspecified, the GUID of the Cal Set is returned. If no Cal Set is applied to the specified channel, then "No Calset Selected" is returned.

**Return Type** String

**Default** Not Applicable

---


**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

This command is replaced by CSET:CAT?

(Read-only) Returns a list of Cal Sets.

**Parameters**

- `<char>` Optional argument. The list is returned in one of the following formats. Both return comma-separated string lists.

  - **GUID** Cal Sets are listed by GUID (Default if unspecified).
  - **NAME** Cal Sets are listed by Name

**Examples**

```plaintext
SENS:CORR:CSET:CAT?
'returns:
{FD6F633E-9719-11d5-8D6C-00108334AE96},{1B03B2CE-971A-11d5-8D6C-00108334AE96}
```
SENSe<cnum>:CORRection:CSET:COPY <string>

Applicable Models: All

(Write-only) Creates a new Cal Set and copies the current Cal Set data into it. Use this command to manipulate data on a Cal Set without corrupting the original cal data.

Parameters

- `<cnum>` Channel number using the Cal Set to be copied. If unspecified, value is set to 1
- `<string>` Name of the new Cal Set.

Examples

```
SENS2:CORR:CSET:COPY 'My2Port'
```

Query Syntax Not Applicable

Default Not Applicable

---

SENSe<cnum>:CORRection:CSET:CREate [name]

Applicable Models: All

(Write-only) Creates an empty Cal Set and attaches it to the specified channel. This command is ONLY necessary before remotely filling the Cal Set with error term data. (For Advanced Users).

A Cal Set is automatically created, applied to the channel, and saved at the completion of a guided cal according to the preference setting `SENS:CORR:PREF:CSET:SAVE`.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `[name]` Optional argument. Name of the Cal Set. Spaces or punctuation are NOT allowed. If unspecified, a unique name is chosen in the form "Calset_N" where N is a unique number.

Examples

```
SENS:CORR:CSET:CRE 'My2Port'
```

Query Syntax Not Applicable

Default Not Applicable
SENSe<cnum>:CORRection:CSET:CREate:DEFault [<csetname>], [<correctiontype>]

Applicable Models: All

(Write-only) Creates a unity Cal Set useful for debugging or to quickly test a prototype of automation software.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `[<csetname>]` Optional argument. Name of the Cal Set. Spaces or punctuation are NOT allowed. If unspecified, a unique name is chosen in the form "Calset_N" where N is a unique number.
- `[<correctiontype>]` Optional argument. Specifies the correction type to use as the default. In general, “Full NP(X,Y,Z,...)” where N is the number of ports covered in the calset, and “X,Y,Z” are a port list.

  - "Full 2P(1,2)"
    - 2 port S-parameter correction including Port 1 and Port 2
  - "Full 3P(2,3,4)"
    - 3 port S-parameter correction including Port 2, Port 3, and Port 4
  - "Full 2P+(1,2)"
    - 2 port S-parameter correction with power correction including Port 1 and Port 2
  - "Full 1P(3)"
    - 1 port S-parameter correction on Port 3
  - "ResponseTracking(a1)"
    - Response tracking term for the a1 receiver

Examples

- 'This example applies 2-port error correction as if the system were perfect. All error terms will be 1 or 0. SENS:CORR:CSET:CRE:DEF 'My2Port', 'Full 2P(1,2)'

Query Syntax

Not Applicable

Default

Not Applicable

SENSe<cnum>:CORRection:CSET:DATA <eterm, portA, portB>[<rec>,] <block>

Applicable Models: All
(Read-Write) Read or Write a specific error term from/to the Cal Set currently attached to the specified channel. (For Advanced Users). The command can be used only for the error terms listed. See **SENS:CORR:CSET:ETERM** to get and put error term data using a string argument for all error terms.

**Parameters**

<**cnun**> Any existing channel number. If unspecified, value is set to 1

<**eterm, portA, portB**> Error Term, Port pair of the specified error term.

Although not all error terms use two port numbers, two are required by the VNA in all cases. Each port number must be between 1 and the number of ports on the VNA.

**EDIR** - directivity

portA: the port at which directivity is measured.

portB: Not used, but must be a valid VNA port number.

**ESRM** - source match

portA: the port at which source match is measured.

portB: Not used, but must be a valid VNA port number.

**ERFT** - reflection tracking

portA: the port at which reflection tracking is measured.

portB: Not used, but must be a valid VNA port number.

**ELDM** - load match

portA: the port at which load match is measured.

portB: the source port.

Load match is measured with a cable connected between the measured port (portA) and the source port (portB).

The cal system requires that the complete matrix of loadmatch arrays be filled. In most cases you can measure loadmatch once at a port, driven by any other port. Then use that data for all variations of the receive port. (The exception is the 3-port VNA models, which requires the loadmatch-measured port to be driven by every other port.)

For example: Measure the loadmatch at port2 while driving port1. Then upload
this same data to the following arrays:

ELDM,2,1,<data>
ELDM,2,3,<data>
ELDM,2,4,<data>

**ETRT** - transmission tracking

portA: the receive port
portB: the source port for this measurement

**EXTLK** - crosstalk

portA: the receive port
portB: the source port for this measurement

**ERSPT** - response tracking.

portA: Not used, but must be a valid VNA port number.
portB: Not used, but must be a valid VNA port number.

**ERSPI** - response isolation.

portA: Not used, but must be a valid VNA port number.
portB: Not used, but must be a valid VNA port number.

**<rec> <string>** - Specify the VNA receiver for which the Eterm applies.

Required ONLY when Eterm is response tracking (**ERSPT**) or response isolation (**ERSPI**).

Logical receiver notation is allowed.

A full 4-port calibration requires the following terms be uploaded:
Reflection terms

Transmission terms

<Block> (Block). Error term data. A Real / Imaginary data pair for each data point.

Format is set using FORM:DATA command.

For REAL binary formats, refer to Getting Data from the Analyzer using SCPI.

Example

'Set the directivity term with a cal set using 5 points


Query Syntax

SENSe<cnum>:CORRection:CSET:DATA? <eterm,portA, portB>,<rec>

Query Examples

'Read the response isolation eterms for the port 1 reference receiver

sens:corr:cset:data? ERSPI,1,1,'R1'

'Same receiver using logical receiver notation
SENSe<cnum>:CORRection:CSET:DEACtivate

**Applicable Models:** All

*(Write-only)* Unselects a Cal Set from the specified channel.

**Parameters**

- `<cnum>`: Channel number to have Cal Set unselected.

**Examples**

```
SENS:CORR:CSET:DEAC
sense2:correction:cset:deactivate
```

**Query Syntax**

Not Applicable

Default

Not Applicable

---

SENSe:CORRection:CSET:DELete <string> - **Superseded**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

This command is replaced by `CSET:DEL`.

*(Write-only)* Deletes a Cal Set from the set of available Cal Sets. This method immediately updates the Cal Set file on the hard drive. If the Cal Set is currently being used by a channel or does not exist, this request will be denied and an error is returned.

**Parameters**

- `<string>`: Cal Set to be deleted. Specify the Cal Set by **GUID** or **Name**. Use `SENS:CORR:CSET:CAT?` to list the available Cal Sets in either format.

**Examples**

```
sense2:correction:cset:delete 'MyCalSet'
```

**Query Syntax**

Not Applicable

Default

Not Applicable

---

SENSe<cnum>:CORRection:CSET:DESCription <string>
Applicable Models: All

(Read-Write) Sets or returns the descriptive string assigned to the selected Cal Set. Change this string so that you can easily identify each Cal Set. Apply and select the Cal Set using `SENS:CORR:CSET:ACT`.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<string>`: The descriptive string associated with the currently-selected Cal Set

**Examples**

```
SENS:CORR:CSET:DESC 'MyCalSet'
sense2:correction:cset:description 'thisCalSet'
```

**Query Syntax**

`SENS<channel>:CORRection:CSET:DESCription?`

**Return Type**

String

**Default**

Not Applicable

---

**SENS<channel>:CORRection:CSET:ETERm[:DATA] <string>,<data>**

Applicable Models: All

(Read-Write) Sets or returns error term data for all VNA measurements. This command modifies a calset that is currently in use by the channel. To see the effects of this modification you need to save the calset and turn correction off and then on again. The commands are as follows:

`SENS:CORR:CSET:SAVE`

`SENS:CORR:STATe OFF`

`SENS:CORR:STATe ON`

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<string>`: (String) Error term to read or write. The error term is specified using the EXACT case-sensitive string displayed in the Cal Set Viewer utility. See `SENS:CORR:CSET:DATA` for a description of port numbers.
- `<data>`: (Block) Error term data. A Real / Imaginary data pair for each data point.

Format is set using `FORM:DATA` command.

For REAL binary formats, refer to [Getting Data from the Analyzer using SCPI](#)

**Examples**

```
SENS:CORR:CSET:ETERM "Directivity(1,1)" , 0.237,-1.422, 0.513, 0.895  ' set directivity(source error term for 2 points
SENS:CORR:CSET:ETERM? "Directivity(1,1)"  'read
```
## Query Syntax

**SENSe<cnum>:CORRection:CSET:ETERm[:DATA]? <string>**

### Return Type
- **Block data**
- **Default**: Not Applicable

###-ref

###SENSe<cnum>:CORRection:CSET:ETERm:CATalog?**

**Applicable Models**: All

*(Read-only)* Returns a list of error term names found in the current Cal Set that is applied to the specified channel.

#### Parameters

#### Examples

```
SENs:CORr:CSET:ETER:CAT?

'For a 1-port cal, returns

"Directivity(1,1),ReflectionTracking(1,1),SourceMatch(1,1)"
```

### Return Type
- **String**
- **Default**: Not Applicable

---

## SENSe<cnum>:CORRection:CSET:FLATten <string>

**Applicable Models**: All

*(Write-only)* When a Cal Set that was produced by a calibration has been interpolated or otherwise modified (for example, by Fixturing operations) this command saves the modified Cal Set to the VNA hard drive so that it can be reused. There is no User Interface equivalent for this command.

### Background

When a Cal Set is selected for use by a channel, the channel reads the Cal Set from disk (primary Cal Set). If the channel aligns perfectly with the Cal Set, the primary Cal Set is used directly. In this case, the active Cal Set is the primary Cal Set.

When processing occurs on the error terms due to interpolation or modification due to the use of fixturing, the channel will generate a temporary "memory-resident" Cal Set. In this case, the active Cal Set is the memory-resident Cal Set. This FLATten command allows you to save the active Cal Set to disk.

Depending on the measurement conditions, this flattening of the Cal Set can improve performance, especially if the Cal Set is applied often (using multiple recall states) or used by many channels. Flattening a version of the Cal Set for each channel can avoid the interpolation or the fixturing operations.
processing that would otherwise occur when the Cal Set is selected or the instrument state is recalled.

You will have to manage the application of such a Cal Set as the VNA itself will have no way to determine what processing had been done once the flatten command is used. For example, if fixture de-embedding occurred prior to the flatten command, that Cal Set should then be applied WITHOUT fixturing on, because fixturing is already embedded in that Cal Set. It is your responsibility to apply the Cal Set properly.

If you want to repeatedly de-embed multiple networks (i.e. concatenate multiple 2-port de-embedding files) you can use the flatten command to create a new primary Cal Set after each de-embed, and sequentially add additional de-embed networks.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number on which the modified Cal Set resides. If unspecified, value is set to 1</td>
</tr>
<tr>
<td>&lt;string&gt;</td>
<td>Name of the new Cal Set. Spaces or punctuation NOT allowed.</td>
</tr>
</tbody>
</table>

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:CSET:FLAT &quot;MyCalSet&quot;</td>
<td></td>
</tr>
</tbody>
</table>

Query Syntax: Not Applicable

Default: Not Applicable

SENSe<cnum>:CORRection:CSET:GENerate:RECeiver <receiverName>

Applicable Models: All Models

This command converts the selected Cal Set from an S-parameter Cal Set to an S-parameter+Power Cal Set. This command requires a Cal Set to be selected.

There are 2 modes for using this command:

Mode 1:

The <receiverName> is optional. If not specified, then ResponseTracking(a1) is set to 1, and the rest of ResponseTracking() terms are computed to be consistent with the S-parameter calibration terms.

Mode 2:

Use this pattern when there is already a receiver calibration for one of the receivers. In that case, this command can be used to transfer the receiver calibration to the other receivers.

If <receiverName> is specified, it must be either 'a1','a2','a3', etc or 'b1','b2','b3'. The ResponseTracking term for this receiver must already be added to the Cal Set, or else this
command will generate an error. This command will then compute the ResponseTracking() terms for all of the other receivers in a manner consistent with the S-parameter calibration terms.

**Parameters**

- `<cnum>` Channel number on which the modified Cal Set resides. If unspecified, value is set to 1.
- `<receiverName>` Name of the receiver ('a1', 'a2', 'a3', etc., or 'b1', 'b2', 'b3', etc.)

**Examples**

```
SENS:CORR:CSET:GEN:REC 'a1'
```

**Query Syntax**

```
SENSe<cnum>:CORRection:CSET:GENerate:RECeiver?
```

**Default** Not Applicable

---

**SENSe<cnum>:CORRection:CSET:GUID <string> Superseded**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

This command is replaced by **SENS:CORR:CSET:ACTivate**.

**(Read-Write)** Selects the Cal Set identified by the string parameter (GUID) and applies it to the specified channel.

- A Cal Set cannot be selected for a channel which is not ON.
- If the stimulus settings of the selected Cal Set differ from those of the selected channel, the instrument will automatically change the channel's settings to match the Cal Set.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<string>` GUID of the desired Cal Set. The curly brackets and hyphens must be included.

**Examples**

```
```

**Query Syntax**

```
SENSe<cnum>:CORRection:CSET:GUID?
```

Returns the GUID of the currently-selected Cal Set for the specified channel.

**Return Type** String

**Default** Not Applicable
SENSe<cnum>:CORRection:CSET:ITEM:CATalog?

Applicable Models: All

(Read-only) Returns the names of the items in the Cal Set.

Parameters

Examples

SENS:CORR:CSET:ITEM:CAT?
"Created By,Firmware Revision,Model Number,Serial Number" 'Example returned item names.'

Return Type String

If no Cal Set is applied on the current channel, the following error message is displayed:

+163, "Requested Cal Set was not found in Cal Set Storage."

Default Not Applicable

SENSe<cnum>:CORRection:CSET:ITEM[:DATA]? <itemName>

Applicable Models: All

(Read-only) Read the value of the Cal Set item. The Cal Set item is added by the VNA firmware to every Cal Set.

About Cal Set Items

A Cal Set item is a named value. You can list the named values using CSET:ITEM:CATalog? or SENS:CORR:CSET:ITEM:CATalog?

You can query the value of a specific item by asking for its data: CSET:ITEM:DATA?

For example, one of the items added by the VNA firmware to every Cal Set is named 'Created By'. The value attached to this item is the name of the VNA Measurement Class or Channel that created the Cal Set. When an SMC cal is performed, you can query the Cal Set for the 'Created By' item, and it will return 'Scalar Mixer/Converter'. The same query on an NFx channel returns 'Noise Figure Converters'.

Parameters

<cnun> Any existing channel number. If unspecified, value is set to 1

<itemName> (String) Item added by the VNA firmware to the currently loaded Cal Set.

Examples

SENS:CORR:CSET:ITEM? "Model Number"

"N5242B" 'Example returned Model Number value.'

Return Type String

If no Cal Set is applied on the current channel, the following error message is
displayed:

+163, "Requested Cal Set was not found in Cal Set Storage."

**Default** Not Applicable

---

### SENSE<cnump>:CORRection:CSET:NAME <string>

**Applicable Models:** All

(Read-Write) Sets or queries the name of the Cal Set currently applied to the specified channel.

**Parameters**

- `<cnump>`: Any existing channel number. If unspecified, value is set to 1
- `<string>`: Name of the Cal Set. Spaces or punctuation NOT allowed.

**Examples**

```
SENS:CORR:CSET:NAME 'MyCalSet'
sense2:correction:cset:name 'thisCalSet'
```

**Query Syntax**

SENSe<cnump>:CORRection:CSET:NAME?

**Return Type** String

**Default** Not Applicable

---

### SENSE<cnump>:CORRection:CSET:SAVE [<char>]

**Applicable Models:** All

This command is NOT necessary after completion of a calibration. A Cal Set is automatically created, applied to the channel, and saved at the completion of a guided cal according to the preference setting SENS:CORR:PREF:CSET:SAVE.

(Read Write)

Saves the channel's Cal Set to the VNA hard drive. For example, use this command after writing data to a Cal Set using SENS:CORR:CSET:DATA (For Advanced Users).

The file name is saved as "CSETx.cst" where x is the user number assigned to `<char>`, and .cst specifies a Cal Set and instrument state. This is not the same syntax as a file saved through the default choices from the front panel, which is "at00x.cst". For more information on the file naming syntax, see the MMEMory subsystem. Learn more about Instrument/Cal States.

**Parameters**

- `<cnump>`: Any existing channel number. If unspecified, value is set to 1
[<char>] Optional argument. Choose from:

USER01
USER02...
and so forth, until...
USER10

If <char> is NOT specified, changes that may have been made are saved to the cal set and NOT to the *.cst file.

**Examples**

```
SENS:CORR:CSET:SAVE USER03
sense2:correction:cset:save user09
'save changes to only the cal set
SENS:CORR:CSET:SAVE
```

**Query Syntax**

`SENS<cnun>:CORRection:CSET:SAVE?`

Queries the last correction set saved.

**Return Type**

Character

**Default**

Not applicable

---

`SENS<cnun>:CORRection:CSET[:SELect] <char>` **Superseded**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

This command is replaced by `MMEM:LOAD`

(Read-Write) Recalls a *.cst file from memory. The file name is "CSETx.cst" where x is the user number assigned to <char>. Learn more about .cst files

For more information on the file naming syntax, see the **MMEMory** subsystem.

**Note:** This command does NOT select a Cal Set for a channel. To select a Cal Set, use `SENS:CORR:CSET:ACTivate`

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Choose from:
  - **DEF** - Presets the analyzer
  - USER01 - Restores User01 calibration data
<table>
<thead>
<tr>
<th>USER02</th>
<th>Restores User02 calibration data through...</th>
</tr>
</thead>
<tbody>
<tr>
<td>USER10</td>
<td>Restores User10 calibration data</td>
</tr>
</tbody>
</table>

**Examples**

| SENS:CORR:CSET DEF | sense2:correction:cset:select user02 |

**Query Syntax**

SENS<cnm>:CORRection:CSET[:SElect]?

**Return Type**

- **Default**: DEF

---

**SENS<cnm>:CORRection:CSET:STANdard[:DATA] <string>,<data>**

**Applicable Models:** All

*(Read-Write)* Sets or returns standard data. Standard data is available for Unguided Cals ONLY.

**Note:** The “Standards data” container in the calset is intended for internal use only. External access is provided for use in diagnosing calibration problems. Users should not form any expectations as to the presence of the data or the naming conventions used.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<string>`: (String) Cal standard to read or write. The standard is specified using the EXACT case-sensitive string displayed in the Cal Set Viewer utility. See SENS:CORR:CSET:DATA for a description of port numbers.
- `<data>`: (Block). Acquisition data. A Real / Imaginary data pair for each data point. Format is set using FORM:DATA command.

For REAL binary formats, refer to Getting Data from the Analyzer using SCPI.

**Examples**

| SENS:CORR:CSET:STAN 'S11C(1,1), 0.237,-1.422, 0.513, 0.895 ' | Set acquisition data for two points. |
| SENS:CORR:CSET:STAN:DATA? "S11C(1,1)" | *Read data* |

**Query Syntax**

SENS<cnm>:CORRection:CSET:STANdard[:DATA]? (string)

**Return Type**

- **Default**: Block data
  - **Not Applicable**
**SENSe:CORRection:CSET:STAND:d:CATalog?**

**Applicable Models:** All

*(Read-only)* Returns a list of available standard name found in the current Cal Set that is applied to the specific channel.

**Parameters**

**Examples**

```
SENS:CORR:CSET:STAN:CAT?
```

**Return Type** String

**Default** Not Applicable

---

**SENSe<ch>:CORRection:CSET:STIMulus? [num]**

**Applicable Models:** All

*(Read-only)* Returns the source or response stimulus values for the Cal Set that is currently used by channel `<ch>`. Values are returned in the format specified by `FORM:DATA` (Block or ASCII).

**Parameters**

- `<ch>` Channel number to query Cal Set stimulus values. If unspecified, value is set to 1
- `[num]` Optional argument. Range of frequencies to return. These values would be different when FOM (Opt S93080A) is enabled.

  - 0 - returns source frequencies. Default setting if not specified.
  - 1 - returns response frequencies.
  - 2 - returns primary frequencies.

**Examples**

```
SENS:CORR:CSET:STIM?
```

```
sense:correction:cset:stimulus 1
```

**Return Type** Numeric

**Default** Not Applicable

---

**SENSe:CORRection:CSET:TSET:ALLPorts? <cset>**
Applicable Models: All

(Read-only) Reads the port mapping used for the specified Cal Set. The returned values are the physical ports. The POSITION of the returned values corresponds to the logical ports.

For example, with an N44xx test set, if the returned string is "PNA 1,TS 2,PNA 2, TS 4" this means:

- VNA 1 is assigned to logical port 1
- TS 2 is assigned to logical port 2
- VNA 2 is assigned to logical port 3
- TS 4 is assigned to logical port 4

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cset&gt;</code></td>
<td>(String) Name or GUID of the Cal Set. Use SENS:CORR:CSET:CAT? to read the list of available Cal Set names or GUIDs.</td>
<td></td>
</tr>
</tbody>
</table>

Examples

SENS:CORR:CSET:TSET:ALLP? "MyCalSet"

Return Type: String

Default: Not Applicable

SENSe:CORRrection:CSET:TSET:TYPe? `<cset>`

Applicable Models: All

(Read-only) Reads the test set type (model) used for the specified Cal Set.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cset&gt;</code></td>
<td>(String) Name or GUID of the Cal Set. Use SENS:CORR:CSET:CAT? to read the list of available Cal Set names or GUIDs.</td>
<td></td>
</tr>
</tbody>
</table>

Examples

SENS:CORR:CSET:TSET:TYPe? "MyCalSet"
'returns "N44xx"

Return Type: String

Default: Not Applicable
SENSe<ch>:CORRection:CSET:TYPE:CATalog? [format]

Applicable Models: All

(Read-only) Query the Cal Types available in the selected Cal Set. The output is a comma separated list of Guids or a Cal Type names. Learn more about applying Cal Types using SCPI.

Use CALC:MEAS:CORR:TYPE to apply a Cal Type.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

[format] (Optional) Format of the output of cal types. choose from:

NAME - (default) returns a list of cal type string names.

GUID - returns a list of cal type GUIDs

Examples

SENS:CORR:CSET:TYPE:CAT? NAME
SENS2:CORRection:CSET:TYPE:CAT?

Return Type String

Default Not Applicable
Perform and applies Port Extensions.

SENSe:CORRection:EXTension:

AUTO
- CONFig
- DCOFfset
- LOSS
- MEASure
- PORT
- RESet
- STARt
- STOP

PORT
- DISTance
- FREQuency
- INCLude
  - [STATe]
- LDC
- LOSS
- MEDium
- SYSMedia
- SYSVelocity
- [TIME]
- UNIT
<table>
<thead>
<tr>
<th>VELFactor</th>
<th>WGCutoff</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECeiver</td>
<td></td>
</tr>
<tr>
<td>[TIME]</td>
<td></td>
</tr>
<tr>
<td>[STATe]</td>
<td></td>
</tr>
</tbody>
</table>

Click on a keyword to view the command details.

See Also

- Example Programs
- Learn about Port Extensions
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

### SENSE<cnw>:CORRrection:EXTension:AUTO:CONFig <char>

**Applicable Models:** All

*(Read-Write)* Sets the frequencies used to calculate Automatic Port Extension. [Learn more about calculating Automatic Port Extension.](#)

#### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1</td>
</tr>
<tr>
<td>&lt;char&gt;</td>
<td>Frequencies to be used:</td>
</tr>
</tbody>
</table>

- **CSPN** - Use current frequency span.
- **AMKR** - Use active marker frequency.

#### Examples

```
SENS:CORR:EXT:AUTO:CONF CSPN
sense2:correction:extension:auto:config amkr
```

#### Query Syntax

`SENSe<cnw>:CORRection:EXTension:AUTO:CONFig ?`

#### Return Type

Character

#### Default

CSPN
SENSe<cnum>:CORRection:EXTension:AUTO:DCOFfset <bool>

Applicable Models: All

(Read-Write) Specifies whether or not to include DC Offset as part of automatic port extension. Learn more about Automatic DC Offset. Only allowed when SENSe:CORRection:EXTension:AUTO:LOSS is set to ON.

Parameters

$cnum$ Any existing channel number. If unspecified, value is set to 1
$bool$ ON (or 1) - Includes DC Offset correction.
OFF (or 0) - Does NOT include DC Offset correction.

Examples

```
SENS:CORR:EXT:AUTO:DCOF 1
sense2:correction:extension:auto:dcoffset off
```

Query Syntax

SENSe<cnum>:CORRection:EXTension:AUTO:DCOFfset?

Return Type

Boolean

Default

OFF (0)

SENSe<cnum>:CORRection:EXTension:AUTO:LOSS <bool>

Applicable Models: All

(Read-Write) Specifies whether or not to include loss correction as part of automatic port extension. Learn more about Loss Compensation in port extension.

Parameters

$cnum$ Any existing channel number. If unspecified, value is set to 1
$bool$ ON (or 1) - Includes Loss correction.
OFF (or 0) - Does NOT include Loss correction.

Examples

```
SENS:CORR:EXT:AUTO:LOSS 1
sense2:correction:extension:auto:loss off
```

Query Syntax

SENSe<cnum>:CORRection:EXTension:AUTO:LOSS?

Return Type

Boolean

Default

OFF (0)

SENSe<cnum>:CORRection:EXTension:AUTO:MEASure <char>
Applicable Models: All

(Write-only) Measures either an OPEN or SHORT standard. When this command is sent, the VNA acquires the measurement with which to set automatic port extensions. This command should be preceded by the CALCulate:PARameter:MNUMber <num> where num is the trace number of a measurement on the specified channel. Learn more about which standard to measure.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Standard to be measured. Choose from:

  - OPEN Measure OPEN standard
  - SHORT Measure SHORT standard

**Examples**

```
SENS:CORR:EXT:AUTO:MEAS OPEN
sense2:correction:extension:auto:measure short
```

**Query Syntax** Not Applicable

**Default** Not Applicable

---

SENSe<cnum>:CORRection:EXTension:AUTO:PORT<n> <bool>

Applicable Models: All

(Read-Write) Enables and disables automatic port extensions on the specified port.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<n>` VNA Port number to enable or disable for automatic port extensions.
- `<bool>` ON (or 1) - Enable
  
  OFF (or 0) - Disable

**Examples**

```
SENS:CORR:EXT:AUTO:PORT2 0
sense2:correction:extension:auto:port4 on
```

**Query Syntax** SENSce<cnum>:CORRection:EXTension:AUTO:PORT<n>?

**Return Type** Boolean

**Default** All ports ON (enabled)

---

SENSe<cnum>:CORRection:EXTension:AUTO:RESet
Applicable Models: All

(Write-only) Clears old port extension delay and loss data in preparation for acquiring new data. Send this command prior to sending a new series of SENS:CORR:EXT:AUTO:MEAS. If acquiring both OPEN and SHORT standards, do not send this command between those acquisitions.

Parameters
<cnun> Any existing channel number. If unspecified, value is set to 1

Examples
SENS:CORR:EXT:AUTO:RES
sense2:correction:extension:auto:reset

Query Syntax
Not Applicable

Default
Not Applicable

SENSe<cnun>:CORRection:EXTension:AUTO:STARt <value>

Applicable Models: All

(Read-Write) Set the start frequency for custom user span. Learn more about User Span.

Parameters
<cnun> Any existing channel number. If unspecified, value is set to 1
<value> User span start value. Must be within the frequency range of the active channel and less than the value set by SENS:CORR:EXT:AUTO:STOP.

Examples
SENS:CORR:EXT:AUTO:STAR 1E9
sense2:correction:extension:auto:start 200e6

Query Syntax
SENSe<cnun>:CORRection:EXTension:AUTO:STARt <value>?

Return Type
Numeric

Default
Start frequency of the current active channel.
Applicable Models: All

(Read-Write) Set the stop frequency for custom user span. Learn more about User Span.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<value>`: User span stop value. Must be within the frequency range of the active channel and greater than the value set by SENS:CORR:EXT:AUTO:STARt.

Examples

```
SENS:CORR:EXT:AUTO:STOP 1E9
sense2:correction:extension:auto:stop 200e6
```

Query Syntax

```
SENSe<cnum>:CORRection:EXTension:AUTO:STOP <value>?
```

Return Type: Numeric

Default: Stop frequency of the current active channel.

SENSe<cnum>:CORRection:EXTension:PORT<pnum>:DISTance <value> Superseded

Applicable Models: All


(Read-Write) Sets and returns the port extension delay in physical length (distance).

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<pnum>`: Port Number that will receive the delay setting. If unspecified, value is set to 1.
- `<value>`: Physical length of fixture of added transmission line. First specify units with SENS:CORR:EXT:PORT:UNIT.

Examples

```
SENS:CORR:EXT:PORT1:DIST 12
sense2:correction:extension:port2:distance .003
```

Query Syntax

```
SENSe<cnum>:CORRection:EXTension:PORT<pnum>:DISTance?
```

Return Type: Numeric

Default: 0

SENSe<cnum>:CORRection:EXTension:PORT<pnum>:FREQuency<n> <value> Superseded
Applicable Models: All

The command set of `SENSe<cnum>:CORRection:EXTension:PORT:XXXX` is changed to `CALCulate:FSIMulator:DRAFt:EXTension:PORT:YYYY` and `CALCulate:FSIMulator::EXTension:PORT:YYYY`?

Learn about Using Fixture Simulator.

(Read-Write) Sets and returns the frequency for the Freq and Loss pair number and for the specified port number.

Learn about Loss Compensation values.

### Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<pnum>`: Port Number that will receive the freq/loss settings. If unspecified, value is set to 1.
- `<n>`: Freq and Loss pair number. Choose from 1 or 2. If unspecified, value is set to 1.
- `<value>`: Frequency value. Choose a frequency within the frequency span of the VNA.

### Examples

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SENS:CORR:EXT:PORT1:FREQ1 10E9</code></td>
</tr>
<tr>
<td><code>sense2:correction:extension:port2:freq2 2E10</code></td>
</tr>
</tbody>
</table>

### Query Syntax

`SENSe<cnum>:CORRection:EXTension:PORT<pnum>:FREQuency<n>`?

### Return Type

- **Numeric**

### Default

- 1 GHz

`SENSe<cnum>:CORRection:EXTension:PORT<pnum>:INCLUDen<n>[;STATe] <bool>`

**Superseded**
Applicable Models: All


(Read-Write) Sets and returns the ON/OFF state for the Freq and Loss pair number and for the specified port number.

Learn about Loss Compensation values.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number that will receive the Freq/Loss settings. If unspecified, value is set to 1.
- `<n>` Freq and Loss pair. Choose from 1 or 2. If unspecified, value is set to 1.
- `<value>` State of Freq and Loss values for port extension.

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 or OFF</td>
<td>Specified Freq and Loss values are OFF</td>
</tr>
<tr>
<td>1 or ON</td>
<td>Specified Freq and Loss values are ON</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:CORR:EXT:PORT:INCL 0
sense2:correction:extension:port2:include2:state on
```

**Query Syntax**

`SENSe<cnum>:CORRection:EXTension:PORT<pnum>:INCLude[:STATe]?`

**Return Type**

Boolean

**Default**

OFF

`SENSe<cnum>:CORRection:EXTension:PORT<pnum>:LDC <value>` Superseded
Applicable Models: All

The command set of `SENSe<cnum>:CORRection:EXTension:PORT:XXXX` is changed to `CALCulate:FSIMulator:DRAFt:EXTension:PORT:YYYY` and `CALCulate:FSIMulator::EXTension:PORT:YYYY?`
Learn about Using Fixture Simulator.

(Read-Write) Sets and returns the Port Loss at DC value for the specified port number.

Learn about Loss Compensation values.

Note: This command affects ALL measurements on the specified channel.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port number to receive Loss value. If unspecified, value is set to 1.
- `<value>` Loss in dB. Choose a value between -90 and 90

Examples

```
SENS:CORR:EXT:PORT:LDC 1.5
sense2:correction:extension:port2:ldc .1
```

Query Syntax

`SENSe<cnum>:CORRection:EXTension:PORT<pnum>:LDC?`

Return Type

Numeric

Default

0

---

`SENSe<cnum>:CORRection:EXTension:PORT<pnum>:LOSS<n> <value>` Superseded

Applicable Models: All

The command set of `SENSe<cnum>:CORRection:EXTension:PORT:XXXX` is changed to `CALCulate:FSIMulator:DRAFt:EXTension:PORT:YYYY` and `CALCulate:FSIMulator::EXTension:PORT:YYYY?`
Learn about Using Fixture Simulator.

(Read-Write) Sets and returns the Loss value for the specified port number.

Learn about Loss Compensation values.

Note: This command affects ALL measurements on the specified channel.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number that will receive the Freq/Loss settings. If unspecified, value is set to 1.
Loss "Use" number. Choose from 1 or 2. If unspecified, value is set to 1.

Loss in dB. Choose a value between -90 and 90

**Examples**

```
SENS:CORR:EXT:PORT:LOSS 1
sense2:correction:extension:port2:loss 1
```

**Query Syntax**

SENSe<cnum>:CORRection:EXTension:PORT<pnum>:LOSS<n>?

**Return Type**

Numeric

**Default**

0

---

**SENSe<cnum>:CORRection:EXTension:PORT<pnum>:MEDium <char> Superseded**

**Applicable Models:** All

The command set of SENSe<cnum>:CORRection:EXTension:PORT:XXXX is changed to CALCulate:FSIMulator:DRAFt:EXTension:PORT:YYYY and CALCulate:FSIMulator::EXTension:PORT:YYYY?

Learn about Using Fixture Simulator.

*(Read-Write)* Sets and returns the media type of the added fixture or transmission line.

See also SENSe:CORR:EXT:PORT:SYSMedia

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<pnum>** Port Number for which media type is being set. If unspecified, value is set to 1.
- **<char>** Medium type. Choose from:
  - COAX
  - WAVEguide

**Examples**

```
SENS:CORR:EXT:PORT:MED COAX
sense2:correction:extension:port2:medium waveguide
```

**Query Syntax**

SENSe<cnum>:CORRection:EXTension:PORT<pnum>:MEDium?

**Return Type**

Character

**Default**

COAX
SENSe<cnm>:CORRection:EXTension:PORT:SYSMedia <bool> Superseded

Applicable Models: All

The command set of SENSe<cnm>:CORRection:EXTension:PORT:XXXX is changed to CALCulate:FSIMulator:DRAFt:EXTension:PORT:YYYY and CALCulate:FSIMulator::EXTension:PORT:YYYY?

Learn about Using Fixture Simulator.

(Read-Write) Sets and returns the state of coupling with the system Media type. Learn more.

Note: This command potentially affects ALL measurements on the VNA.

Parameters

- <cnm> Any existing channel number. If unspecified, value is set to 1
- <bool> Coupling state. Choose from:
  - ON (or 1) - Media type is coupled with the system setting.
  - OFF (or 0) - Media type is NOT coupled with the system setting.

Examples

SENSe:CORR:EXT:PORT:SYSM 1
SENSe:CORR:EXT:PORT:SYSMedia off

Query Syntax

SENSe<cnm>:CORRection:EXTension:PORT:SYSMedia?

Return Type

Boolean

Default

1 or ON (Coupled)

---

SENSe<cnm>:CORRection:EXTension:PORT<pn>:SYSVelocity <bool> Superseded

Applicable Models: All

The command set of SENSe<cnm>:CORRection:EXTension:PORT:XXXX is changed to CALCulate:FSIMulator:DRAFt:EXTension:PORT:YYYY and CALCulate:FSIMulator::EXTension:PORT:YYYY?

Learn about Using Fixture Simulator.

(Read-Write) Sets and returns the state of coupling with the system Velocity Factor value. Learn more.

Note: This command potentially affects ALL measurements on the VNA.

Parameters

- <cnm> Any existing channel number. If unspecified, value is set to 1
- <pn> Port Number for which system Velocity Factor coupling is being set. If unspecified, value is set to 1.
Coupling state. Choose from:

- **ON** (or 1) - Velocity Factor is coupled with the system setting.
- **OFF** (or 0) - Velocity Factor is NOT coupled with the system setting.

**Examples**

```
SENS:CORR:EXT:PORT:SYSV 1
sense2:correction:extension:port2:sysvelocity off
```

**Query Syntax**

```
SENSe<cnum>:CORRection:EXTension:PORT<pnum>:SYSVelocity?
```

**Return Type**

Boolean

**Default**

1 or ON (Coupled)

---

**SENSe<cnum>:CORRection:EXTension:PORT<pnum>[:TIME] <num> Superseded**

**Applicable Models:** All


(Read-Write) Sets the extension delay value in time at the specified port. Must also set `SENS:CORR:EXT ON`.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number that will receive the extension. If unspecified, value is set to 1.
- `<num>` The port extension in seconds; may include suffix. Choose a number between: -1E18 and 1E18

**Examples**

```
SENS:CORR:EXT:PORT 2MS
sense2:correction:extension:port2 .00025
```

**Query Syntax**

```
SENSe<cnum>:CORRection:EXTension:PORT<pnum>[:TIME]?
```

**Return Type**

Numeric

**Default**

0

---

**SENSe<cnum>:CORRection:EXTension:PORT:UNIT <char> Superseded**
Applicable Models: All


(Read-Write) Sets and returns the units for specifying port extension delay in physical length (distance).

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<char>` Units for delay in distance. Choose from:
  - METer
  - FEET
  - INCH

Examples

```
SENS:CORR:EXT:PORT:UNIT MET
sense2:correction:extension:port:unit inch
```

Query Syntax

`SENSe<cnum>:CORRection:EXTension:PORT:UNIT?`

Return Type

Character

Default

METer

SENSe<cnum>:CORRection:EXTension:PORT<pnum>:VELFactor <value> Superseded

Applicable Models: All


(Read-Write) Sets and returns the velocity factor of the fixture or added transmission line.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number for which velocity factor is being set. If unspecified, value is set to 1.
- `<value>` Velocity Factor.

Set `SENS:CORR:EXT:PORT:SYSV` to use the system velocity factor.
SENSe<cnum>:CORRection:EXTension:PORT<pnum>:WGCutoff <value> Superseded

Applicable Models: All


Learn about Using Fixture Simulator.

(Read-Write) Sets and returns the cutoff (minimum) frequency of the added waveguide fixture or transmission line.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number for which media type is being set. If unspecified, value is set to 1.
- `<value>` Cutoff frequency in Hz.

This value is ignored when SENS:CORR:EXT:PORT:MED is set to COAX for the same port.

**Examples**

SENSe:CORR:EXT:PORT:WGC 1e8

SENSe2:CORR:EXT:PORT2:WGC 100MHz

**Query Syntax**

SENSe<cnum>:CORRection:EXTension:PORT<pnum>:WGCutoff?

**Return Type**

Numeric

**Default**

System Media Cutoff Frequency
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets the extension value at the specified receiver. Must also set SENS:CORR:EXT ON.

**Note:** Before using this command you must select a measurement using CALC:MEAS:DEF. You can select one measurement for each channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<Rnum>` Number of the receiver that will receive the extension. If unspecified, value is set to 1
  Choose from:
  - 1 for Receiver A
  - 2 for Receiver B
- `<num>` The electrical length in seconds; may include suffix. Choose a number between:
  - -10 and 10

**Examples**

- `SENS:CORR:EXT:REC 2MS`
- `sense2:correction:extension:receiver2:time .00025`

**Query Syntax**

`SENSe<cnum>:CORRection:EXTension:RECeiver<Rnum>[:TIME]?

**Return Type**

- Numeric

**Default**

- 0

**SENSe<cnum>:CORRection:EXTension[:STATe] <ON | OFF> Superseded**
Applicable Models: All


*(Read-Write)* Turns port extensions ON or OFF.

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Any existing channel number. If unspecified, value is set to 1</td>
</tr>
<tr>
<td>`&lt;ON</td>
<td>OFF&gt;`</td>
</tr>
<tr>
<td></td>
<td>OFF (or 0) - turns port extensions is OFF.</td>
</tr>
</tbody>
</table>

### Examples

```plaintext
SENS:CORR:EXT ON
sense2:correction:extension:state off
```

### Query Syntax

```
SENSe<cn>u>m>:CORRection:EXTension[:STATe]?
```

### Return Type

Boolean (1 = ON, 0 = OFF)

### Default

OFF
Sense:Correction:Collect:Guided Commands

Performs and applies a SmartCal (Guided) calibration and other error correction features.

Important Notes:

- To perform a **Guided Calibration**, use ONLY Sens:Corr:Coll:Guided commands. See the "Guided" example programs for clarification.

- ALWAYS send ALL measurement setup commands BEFORE initializing a remote calibration.

```
SENSe:CORRection:COLLect:GUIDed:
   ABORt
   ACQuire
   ADAPter
      | COUNt
      | ZERO
      | CREate?
      | DELay
      | DESCription
      | PATHs
   CHANnel:MODE
   CKIT
      | CATalog?
      | PORT
         | CATalog?
         | [SELect]
   CONNector
      | CATalog?
      | PORT
         | [SELect]
   DATA
      | CATalog?
   DESCription
   DMATch
      | APPLY
         | [IMMediate]
```
| PORTs? |
| [INITiate] |

ECal
| ACQuire |
| SELect |

ETERms
| COMPpute |
| LOAD[:CSET] |

INITiate
ISOLation
| AVERage |
| INCRement |
| PATHs |

ITERations
| COUNt? |
| MINimum? |
| RESet |

LIST
| COUNt? |
| STEP |

| COUNt? |
| DESCription? |
| LABel? |
| PORTs? |
| STANdard |
| LABel? |
| PORTs? |
| STYPe? |
| TPORts? |

METHod
PACQuire
PATH
  | CMETHod
  | TMETHod
PCAL
  | APPLy
PORTs?
PREFerence
  | SLIDingload
PSENsor - More commands
SAVE
  | CSET
SMC - More commands
STATes[:CATalog]?
STEPS?
THRU
UNCertainty
  | CHARacterize
    | CABLE[:INITiate]
    | NOISE[:INITiate]
  | [:ENABle]
VMC - More commands

Click on a keyword to view the command details.

Blue keywords are superseded commands.

See Also

- ECal Orientation commands
- Examples using these commands.
- Calibrating the VNA Using SCPI
- Learn about Measurement Calibration
- Synchronizing the Analyzer and Controller

SENSe<ch>:CORRection:COLLect:GUIDed:ABORt
Applicable Models: All

(Write-only) Aborts the acquiring of a guided calibration that has been INITialized but has not yet been concluded using the SAVE command. If at least one Cal standard has already been measured, and the Calibration Window is being displayed, this command also closes the Calibration Window and re-tiles the other measurement windows.

Parameters

<ch>  Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

Examples

SENS:CORR:COLL:GUID:ABOR
sense2:correction:collect:guided:abort

Query Syntax

Not Applicable

Default  Not Applicable

SENSe:<ch>CORRection:COLLect:GUIDed[:ACQuire] STAN<n>[,sync]

Applicable Models: All

(Write-only) Initiates the measurement of the specified calibration standard. Executing this command with an unnecessary standard has no affect.

The measured data is stored and used for subsequent calculations of error correction coefficients. All standards must be measured before a calibration can be completed. Any measurement can be repeated until the SENS:CORR:COLL:GUID:SAVE command is executed.

Query the user prompt description using SENS:CORR:COLL:GUID:DESC ?

Query the required calibration steps using SENS:CORR:COLL:GUID:STEP?

Parameters

<ch>  Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

STAN<n>  Choose from: STAN1, STAN2, etc. through STANn where n is the number of cal standard connection steps for the calibration.

Note: You do not necessarily have to invoke these connection steps in sequential order, but you must issue this command for all of the steps to be able to complete the calibration.

[sync]  Optional argument. Choose from:

SYNChronous - blocks SCPI commands during standard measurement
ASYNchronous - does NOT block SCPI commands during standard measurement.

Learn more about this argument

Examples
SENSe:CORRection:COLLect:GUIDed:ADAPter:CREate? <conn1>, <conn2>

Applicable Models: All

(Read-only) Specifies the use of a THRU adapter to be used during the Guided Cal Unknown THRU and Adapter Removal Cal. Returns an adapter index <n> which is used to refer to the adapter in several related commands. See Cal Thru Methods. While the choice of which end of the adapter is <conn1> and <conn2> is arbitrary, it is necessary to remember which will be used on each test port.

The settings for this command remain until Preset, or the command is sent using a different setting, or until the ZERO command is sent.

Parameters
<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
<conn1> Adapter port 1 connector type. Use SENS:CORR:COLL:GUID:CONN:CAT? to return a list of valid connector types.
<conn2> Adapter port 2 connector type.

Examples See example using this command.

Return Type Numeric
Default Not Applicable

SENSe<ch>:CORRection:COLLect:GUIDed:ADAPter:COUNt?
Applicable Models: All

(Read-Only) Returns the number of THRU adapters that have been created for this calibration using SENS:CORR:COLL:GUID:ADAP:CREate.

Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

Examples

See example using this command.

Return Type Numeric

Default Not Applicable

SENSe:<ch>CORRection:COLLect:GUIDed:ADAPter:COUNt:ZERO

Applicable Models: All

(Write-only) Removes all adapters that have been defined for calibrations on the specified channel using SENS:CORR:COLL:GUID:ADAP:CREate.

Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

Examples

SENSe:CORRection:COLLect:GUIDed:ADAPter:COUNt:ZERO

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:COLLect:GUIDed:ADAPter<n>:DELay[:VALue] <coax>, [<wphase>], [<wdelay>]
Applicable Models: All

(Write-only) Specifies the adapter delay and optionally waveguide delay and optional phase offset (degrees) of adapter \(<n>\).

The settings for this command remain until Preset, or the command is sent using a different setting, or until the ZERO command is sent.

**Parameters**

- **<ch>** Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- **<n>** Adapter index number that was returned from SENS:CORR:COLL:GUID:ADAP:CREate?
- **<coax>** Delay value of coax adapter \(<n>\) in seconds. If the adapter has no coax connector, enter 0.
- **<wphase>** Waveguide phase offset in degrees. If the adapter has no waveguide connector, do not enter a value.
- **<wdelay>** Waveguide delay in seconds. If the adapter has no waveguide connector, do not enter a value.

**Examples**

See example using this command.

**Default** Not Applicable

---

SENSe<ch>:CORRection:COLLect:GUIDed:ADAPter<n>:DESCription <string>

Applicable Models: All

(Write-only) Specifies the adapter description for use as the guided cal connection prompts.

The settings for this command remain until Preset, or the command is sent using a different setting, or until the ZERO command is sent.

**Parameters**

- **<ch>** Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- **<n>** Adapter index number that was returned from SENS:CORR:COLL:GUID:ADAP:CREate?
- **<string>** Adapter description.

**Examples**

See example using this command.
### SENSE:CORRection:COLLect:GUIDed:ADAPter<n>:PATHs <port pairs>

**Applicable Models:** All

*(Write-only)* Specifies the port pairs for which the adapter will be used for a THRU connection.

For example, for a 3-port cal on channel 1 using ports 1, 2, and 3, to use adapter 1 between the ports (1 to 2) and (1 to 3) the following command is used: `SENSe1:CORRection:COLLect:GUIDed:ADAPter1:PATH 1,2,1,3`.

The adapter must have the same DUT connectors as the ports that are already specified for these ports.

The settings for this command remain until Preset, or the command is sent using a different setting, or until the ZERO command is sent.

**Parameters**

- `<ch>` Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- `<n>` Adapter index number that was returned from `SENSe:CORRection:COLLect:GUIDed:ADAPter:CREate?`.
- `<port pair>` Ports for which the adapter will be used. The orientation is not critical, as the VNA will align the connector types as necessary. The minimum number of Thru connections required is the number of ports to calibrated -1.

**Examples**

See example using this command.

---

### SENSE:CORRection:COLLect:GUIDed:CHANnel:MODE <bool>

---
Applicable Models: All

(Read-Write) Determines whether or not to honor the channel <ch> argument in guided calibration SCPI commands.

Parameters

<bool>  **OFF (0)**  Honor all <ch> arguments. This means the <ch> channel is calibrated regardless of which channel is currently active.

**ON (1)**  Legacy behavior. Behavior is specified by the following table:

<table>
<thead>
<tr>
<th>&lt;ch&gt; channel type</th>
<th>Active channel type</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>Std or App</td>
<td>Std or App</td>
<td></td>
</tr>
<tr>
<td>Std</td>
<td>Std</td>
<td>Active chan cal'd</td>
</tr>
<tr>
<td>Std</td>
<td>App</td>
<td>&quot;Channel not found&quot; error</td>
</tr>
<tr>
<td>App</td>
<td>Std</td>
<td>&lt;ch&gt; chan cal'd</td>
</tr>
<tr>
<td>App</td>
<td>App</td>
<td>&lt;ch&gt; chan cal'd</td>
</tr>
</tbody>
</table>

Learn about Standard vs Application channels.

Examples

```
SENS:CORR:COLL:GUID:CHAN:MODE 0
sense:correction:collect:guided:channel:mode ON
```

Query Syntax  SENSE:CORRection:COLLect:GUIDed:CHANnel:MODE?

Return Type  Boolean

Default  OFF - This is the default beginning with A.09.50

ON - Default before A.09.50

```
```
Applicable Models: All

(Read-only) This command replaces SENS:CORR:COLL:GUID:CKIT:PORT:CAT?

Returns a comma-separated list of valid kits that use the specified connector type. This includes mechanical cal kits, applicable characterizations found within ECal modules currently connected to the VNA, and all user characterizations stored in VNA disk memory. For ECal modules, the returned list includes the serial numbers. See ECal User Characterization commands.

Use items in the list to select the kit to be used with the SENS:CORR:COLL:GUID:CKIT:PORT and SENSE<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:CKIT commands.

Parameters


Examples  SENS:CORR:COLL:GUID:CKIT:CAT? "Type N (50) male"

Return Type String

Default Not Applicable


Applicable Models: All

(Read-only) This command is replaced by SENS:CORR:COLL:GUID:CKIT:CAT?.

Returns a comma-separated list of valid kits for the specified VNA port. In addition to mechanical calibration kits, this will include applicable characterizations found within ECal modules currently connected to the VNA.

Use items in the list to select the kit to be used with the SENS:CORR:COLL:GUID:CKIT:PORT command.

Note: The serial number is returned for ALL ECal modules that are connected with the connector type of the specified port. Previously, the returned list would include the serial numbers to distinguish the ECal modules only when two or more identical ECal models were connected to the VNA.

Parameters

<pnum> Any existing port number. If unspecified, value is set to 1

Examples  SENS:CORR:COLL:GUID:CKIT:PORT1:CAT?

'When "Type N (50) male" is specified for connector type, returns:

"85054D, 85032F"
When two identical ECal modules are connected for the connector type, the return string includes serial numbers "85092-60010 ECal 10675, 85092-60010 ECal 00758"

**Return Type** String

**Default** Not Applicable


**Applicable Models:** All

(Read-Write) Specifies the calibration kit (mechanical or ECal) for each port to be used during a guided calibration. An unused port does NOT need to have a specified Cal Kit.

1. Specify the connector type for the port with SENS:CORR:COLL:GUID:CONN:PORT.

2. Query the valid available kits for the connector on each port with SENS:CORR:COLL:GUID:CKIT:PORT:CAT?

3. Specify the kit using this command.

4. Perform a query of this command. If the <kit> parameter was incorrectly entered, an error will be returned.

When using this command to specify the cal kit for the output of a VMC calibration mixer, specify port 3. If port 3 is already used for the output of the DUT mixer, then specify port 4. Learn more.

**Parameters**

- `<ch>` Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

- `<pnum>` Any existing port number. If unspecified, value is set to 1

- `<kit>` Calibration kit to be used for the specified port. **Case-sensitive.**

When using an ECal module, include the characterization name in the <kit> string. Use SENSe:CORR:COLL:GUID:CKIT:CAT? to read the list of characterizations available in the module and in VNA disk memory.

If two or more identical ECal modules are connected to the VNA, the serial number must be included to distinguish the ECal modules.

**Examples**

'Note: All of the following examples specify port 1 only'

'Mechanical Cal kit

SENS:CORR:COLL:GUID:CKIT:PORT1 '85055A'
Standard ECal modules

SENS:CORR:COLL:GUID:CKIT:PORT1 "N4691-60004 ECal"

Non-factory ECal characterizations are specified as follows:

SENS:CORR:COLL:GUID:CKIT:PORT1 "N4691-60004 User 1 ECal"

When two or more ECal modules with the same model number are connected, also specify the serial number as follows:

SENS:CORR:COLL:GUID:CKIT:PORT1 "N4691-60004 ECal 01234"

When Disk Memory ECal user characterizations are used, specify both the User char and the serial number as follows:

SENS:CORR:COLL:GUID:CKIT:PORT1 "N4691-60004 MyDskChar ECal 01234"

Query Syntax

SENSe:CORRection:COLLect:GUIDed:CKIT:PORT<pnum>[:SELect]?

Return Type

String - If the <kit> parameter was incorrectly entered while writing, an error will be returned.

Default

Not Applicable

SENSe:CORRection:COLLect:GUIDed:CONNector:CATalog?

Applicable Models: All

(Read only) Returns a list of valid connectors based on the connector descriptions of the available cal kits. Use an item from the returned list to specify a connector for SENS:CORR:COLL:GUID:CONN:PORT

Here are the more common connector types:
<table>
<thead>
<tr>
<th>Waveguide</th>
<th>Type</th>
<th>Gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>W-band</td>
<td>Type B</td>
<td>1.00 mm</td>
</tr>
<tr>
<td>V-band</td>
<td>Type A (50)</td>
<td>1.00 mm</td>
</tr>
<tr>
<td>U-band</td>
<td>Type A (50)</td>
<td>1.85 mm</td>
</tr>
<tr>
<td>R-band</td>
<td>Type F (75)</td>
<td>1.85 mm</td>
</tr>
<tr>
<td>Q-band</td>
<td>Type F (75)</td>
<td>2.92 mm</td>
</tr>
<tr>
<td>K-band</td>
<td>Type N (75)</td>
<td>2.92 mm</td>
</tr>
<tr>
<td>P-band</td>
<td>Type N (75)</td>
<td>APC 2.4</td>
</tr>
<tr>
<td>X-band</td>
<td>Type N (50)</td>
<td>APC 2.4</td>
</tr>
<tr>
<td>7-16</td>
<td>Type N (50)</td>
<td>APC 3.5</td>
</tr>
</tbody>
</table>

**Examples**

```
```

**Returns:**

Type N (50) female, Type N (50) male, APC 7 (50), 3.5 mm (50) male, 3.5 mm (50) female, User Connector A

**Return Type** Comma separated string values

**Default** Not Applicable


**Applicable Models:** All

*(Read-Write)* Specifies a DUT connector type for every port during the Guided Calibration procedure. Valid DUT connector names are stored within calibration kits. Some cal kits may include both male and female DUT connectors. Therefore, specifying the DUT connector gender may be required.

The VNA remembers previous Guided Cal settings. Therefore, for completeness, unused ports can either be defined as "Not used" or use the SENS:CORR:COLL:GUID:ABORt command to clear all ports. The ABORt command is a more thorough approach and more convenient. See Guided Cal examples.
A single port with a valid <conn> name indicates a 1-Port calibration will be performed.

Two ports with valid <conn> names indicate either a 2-Port SOLT or TRL calibration will be performed depending on the standards definition found within the cal kit and the capability of the VNA.

Three ports with valid <conn> names indicate a 3-Port calibration will be performed, and so forth.

Follow these steps to ensure port connectors are specified correctly:

2. Set a connector type for each port using this command. If the connector type was incorrectly entered, an error will be returned.
3. Specify the cal kit to use for each port with SENS:CORR:COLL:GUID:CKIT:PORT

Parameters

- `<ch>`: Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- `<pnum>`: Any existing port number. If unspecified, value is set to 1.
- `<conn>`: String - DUT connector type to connect with VNA port `<pnum>`. Case-sensitive.

Examples

- Specifying a 2-port cal (1 & 2) on a 4-port VNA
  - SENS:CORR:COLL:GUID:CONN:PORT1 'Type N (50) female'
  - SENS:CORR:COLL:GUID:CONN:PORT2 'Type N (50) male'
  - SENS:CORR:COLL:GUID:CONN:PORT3 'Not used'
  - SENS:CORR:COLL:GUID:CONN:PORT4 'Not used'

Query Syntax

SENS:CORR:COLL:GUID:CONN:PORT<pnum>[:SELect]?

Return Type

String

Default

Not Applicable

SENSe<ch>:CORR:COLL:GUID:DATA STAN<n>, <meas parameter>, [<ECal state num>],[<data>]

Applicable Models: All

(Read-Write) Sets and returns the measurement data for a specified measurement parameter of a particular step of a guided cal (and for a specific state of an ECal if the step is an ECal step). The measurement data is complex real-and-imaginary pairs where the number of points is the current number of points on the channel, and is in ASCII or binary format as dictated by the current setting of
the FORMat:DATA command.

The query returns measurement data once it has been uploaded. The Cal Wizard will also perform the acquisition as in a normal calibration and then use the query to see the data per ECal state.

**Note:** This command only applies to cals of standard S-parameter channels.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1.

- `STAN<n>`: Choose from: STAN1, STAN2, etc. through STANn where n is the number of cal standard connection steps for the calibration.

  **Note:** You do not necessarily have to invoke these connection steps in sequential order.

- `<meas parameter>`: Measurement parameters for standard S-parameter channels.

- `<data>`: Measurement data to upload.

- ` [<ECal state num>]`: ECal state number.

**Examples**

```
1.40827238560E-002,+2.33575403690E-001,-8.6207535315E-
002,+1.97563484311E-001,-3.09373617172E-001,-3.1908370050E-
002,-2.91508108377E-001,-1.58155187964E-001,-2.40630045533E-
001,-2.9587635479E-001,-1.10904276371E-001,-3.7042664816E-
001,+6.89683631063E-002,-3.54430228472E-002,+9.91653800011E-
002,-3.55183511972E-001,+4.09932315350E-001,-2.48026609421E-
001,+3.73078525066E-001,-5.87668754160E-002,+5.06927728653E-
001,+9.91396754980E-002,+3.50351631641E-001,+2.56966352463E-
001,+9.62104797363E-002,+4.53149110079E-001,-3.36435511708E-
002,+3.78106325865E-001,-2.88381099701E-001,+3.05592954159E-
001,-2.96310603619E-001,+1.54096364975E-001,-4.68389958143E-
001,+4.40414100885E-002,-1.57085657120E-001,-6.15487955511E-
002,-1.97938442230E-001,-1.97066932917E-001,-2.88984715939E-
001,-1.64536476135E-001,+1.0456377799E-001,-3.01358193159E-
001,-1.15329414606E-001,-3.01511257887E-001,+1.76867023110E-
001,-3.1372898171E-001,+2.52688616514E-001,-3.10592085123E-
001,+3.16383123398E-001,-2.51693593212E-002,+5.1517367363E-
001,-3.37228141725E-002,-5.16818091273E-002,+2.94606477022E-
001,+4.42426577210E-002,+6.23966455460E-001,-2.10833102465E-
001,+8.56095775962E-002,-3.56811732054E-001,+3.87589573860E-
001,-5.81349851564E-003,-4.00106996298E-001,-4.85193997622E-
001,-2.21789374948E-001,+5.68915042888E-001,-2.02025130391E-
001,+1.32879754528E-002,-1.02828477554E-001,+3.49758925856E-
001,+4.49306488037E-001,+5.84089420736E-002,+9.62211636899E-
002,-4.81228590012E-002,+3.52717667818E-001,-2.72483140230E-
001,-1.59653365612E-001,-2.63482809067E-001,-1.27529203892E-
001,+7.02303647995E-002,-6.81156396866E-001,-1.32705345750E-
001,-2.89749443531E-001,+2.88871675730E-001,-4.64907467365E-
```
Query Syntax: SENSE:CORRelction:COLLection:GUIDed::DATA? STAN<n>, <meas parameter>, [<ECal state num>]

Return Type: Depends on FORMat:DATA

Default: Not Applicable


Applicable Models: All

(Read-only) Returns a comma-delimited CSV string of measurement parameters (with parameter names) for STANn at State x. The measurement parameters have to be measured in the specified step number of a guided calibration. If the parameter name is a not an S-parameter, an underscore is inserted (for example, b1/a1,1 becomes b1/a1_1).

Note: This command only applies to cals of standard S-parameter channels.

Parameters:

- <ch>: Any existing channel number. If unspecified, value is set to 1.
- STAN<n>: Choose from: STAN1, STAN2, etc. through STANn where n is the number of cal standard connection steps for the calibration.

Note: You do not necessarily have to invoke these connection steps in sequential order.

Examples:

- SENS:CORR:COLL:GUID:DATA:CAT? STAN1,1
  "S11"
  See an example that uses this command.

Return Type: String

Default: Not Applicable
**SENSe<ch>:CORRection:COLLect:GUIDed:DESCription? <step>**

**Applicable Models:** All

*(Read-only)* Returns the connection description for the specified calibration step.

**Parameters**

- `<ch>` Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- `<step>` A number from 1 to the number of steps required to complete the calibration (Use SENS:CORR:COLL:GUID:STEP? to query the number of steps)

**Examples**

```
SENS:CORR:COLL:GUID:DESC? 10

'Returns:
Connect APC 7 Open to port3
```

**Return Type** String

**Default** Not Applicable

**SENSe<ch>:CORRection:COLLect:GUIDed:DMATch:APPLy[:IMMediate] [string]**

**Applicable Models:** N522xB, N523xB, N524xB

*(Write-only)* Specifies a Delta Match Cal Set to be used for delta match correction.

If the User-performed Cal Set GUID or Name is not specified, then the Global Delta Match Cal Set is applied.

An error is returned if the specified Cal Set does not meet the following Delta Match criteria:

- Must have been performed using ECal or as a guided mechanical cal (not Unguided).
- Must have the same start freq, stop freq, and number of points as the channel being calibrated.
- Must calibrate the ports that are required by the TRL or Unknown Thru cal as indicated by SENS:CORR:COLL:GUID:DMATch:APPLy:PORTs?

The Global Delta Match Cal can ALWAYS be applied.

Learn more about Delta match calibration.

See example of a complete Global Delta Match calibration.

See example where Delta Match is applied to a calibration.

**Parameters**
<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

[string] Optional argument. GUID or Name of the User-performed Delta Match Cal Set to apply.

If unspecified, the Global Delta Match Cal Set is applied.

Examples
- `SENS:CORR:COLL:GUID:DMAT:APPL`
- `SENSe:correction:collect:guided:dmatch:apply:immediate
  "{2B893E7A-971A-11d5-8D6C-00108334AE96}"`
- `SENS:CORR:COLL:GUID:DMAT:APPL "MyDMatchCalset"`

Query Syntax Not Applicable

Default Not Applicable

---

`SENSe<ch>:CORRection:COLLect:GUIDed:DMATch:APPLy:PORTs?`

Applicable Models: N522xB, N523xB, N524xB

(Read-only) Returns the port numbers for which delta match correction is required. 0 (zero) is returned if the Cal does NOT require Delta Match correction for one of the following reasons:

- The Cal does NOT involve Unknown THRU or TRL. You specify this using SENS:CORR:COLL:GUID:METH <UNKN | TRL>.

- The Cal DOES involve Unknown THRU or TRL, but the delta match data can be calculated by the Unknown Thru or TRL Cal. Learn how this is possible. However, you can force the Cal to use the Delta Match data from a Cal Set.

Learn more about Delta match calibration.

See example of a complete Delta Match calibration.

Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

Examples
- `SENS:CORR:COLL:GUID:APPL:PORT?`
  - Returns: 1,2,3

Return Type Numeric

Default Not Applicable
**SENSe<ch>:CORRection:COLLect:GUIDed:DMATch[:INITiate] <conn>,<cKit>**

**Applicable Models:** N522xB, N523xB, N524xB

*(Write-only)* Initiates a global delta match calibration.

Learn more about Global Delta Match calibration.

See example of a complete Delta Match calibration.

**Parameters**

- `<ch>` Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- `<conn>` **String**. Connector type for port 1. All other ports are set automatically.
- `<cKit>` **String** Cal Kit for all ports. If incorrectly entered while writing, an error is returned.

**Examples**

```
SENS:CORR:COLL:GUID:DMAT APC 3.5 female,"85052B"
```

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**SENSe<ch>:CORRection:COLLect:GUIDed:ECAL:ACQ <cal method>. <port list>, [calset]**

**Applicable Models:** All

*(Write only)* Execute the Ecal calibration with specified Ecal using SENS:CORR:COLL:GUID:ECAL:SEL . If ECal module is not specified, the first Ecal in the list is used. One item that we discussed is that this command needs to be overlapped. That's because it can be quite slow to finish an ecal, and slow operations need to be implemented using overlapped SCPI. Otherwise, the client will time out and will be unable to poll for completion. So, the command must be followed with either a *OPC? or a *OPC and serial poll.

**Parameters**

- `<ch>` Channel to be calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- `<cal method>` **String** Calibration Method.
  
  SOLT: using defined through
  
  SOLR: using undefined through
ERESponse: Enhanced Response

<port list> Array Port number to be calibrated. If enhanced response (ERESponse) is selected, the first port number is the stimulus port and the second port number is the response port.

[calset] Optional argument. Cal Set name

If NOT specified, behavior depends on the SENS:CORR:PREFerence:CSET:SAVE setting.

If specified, choose an existing Cal Set, either by name or by GUID.

- By Cal Set name: include quotes.
- Query all Cal Set GUIDs with SENS:CORR:CSET:CAT?

An error is reported if the Cal Set is not found.

The Cal Set is either supplemented or overwritten depending on the method, connectors, and ports selected. Learn more about Cal Sets.

Examples

' Full 2 port cal with defined through for ports 2 and 3
SENS:CORR:COLL:GUID:ECAL:ACQ SOLT,2,3

' Enhance Response Cal for ports 4 (Stimulus) and 1 (Response)
SENS:CORR:COLL:GUID:ECAL:ACQ ERES,4,1

' Full 2 port cal with calset and undefined through for ports 1 to 4
SENS:CORR:COLL:GUID:ECAL:ACQ SOLR,1,2,3,4,"MyCalSet"

Query Syntax Not Applicable

Default Not Applicable


Applicable Models: All
Specifies the Ecal Kit for Ecal Calibration. This is a new command that specifies the ECal kit to be used in the new 1-shot ECal execution command. If not specified, the 1-shot command will internally select the top one from the connected ECal kits (like the basic cal and start cal dialogs are doing.)

The top ecal kit in the list can be determined in two ways: by an internal module number that is dependent on the order in which the ecals are enumerated on the USB. Or it could be an alphabetical sorting by name. This default selection will allow the user to use generic test code but only if there is only one ECal connected.

Parameters

- `<ch>` Channel to be calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- `<ecal kit>` Ecal kit to be used for the specified port. **Case-sensitive.**

Include the characterization name in the `<kit>` string. Use SENSe:CORR:COLL:GUID:CKIT:CAT? to read the list of characterizations available in the module and in VNA disk memory.

If two or more identical ECal modules are connected to the VNA, the serial number must be included to distinguish the ECal modules.

Examples

- **Standard ECal modules**
  
  SENSe:CORR:COLL:GUID:ECAL "N4691-60004 ECal"

- **Non-factory ECal characterizations are specified as follows:**
  
  SENS:CORR:COLL:GUID:ECAL "N4691-60004 User 1 ECal"

- **When two or more ECal modules with the same model number are connected, also specify the serial number as follows:**
  
  SENS:CORR:COLL:GUID:ECAL "N4691-60004 ECal 01234"

- **When Disk Memory ECal user characterizations are used,** specify both the User char and the serial number as follows:
  
  SENS:CORR:COLL:GUID:ECAL "N4691-60004 MyDskChar ECal 01234"

Query Syntax

SENSe:CORRection:COLLect:GUIDed:ECAL[:SELect]?

Return Type

String - If the `<kit>` parameter was incorrectly entered while writing, an error will be returned.

Default

Not Applicable
SENSe<ch>:CORRection:COLLect:GUIDed:ETERms:COMPute [cal set name]

**Applicable Models:** All

_(Write-only)_ Computes the error correction terms, turns Correction ON, and saves the calibration to an existing, specified Cal Set.

The cal acquisition process does not conclude as with the SAVE command. This command leaves the cal acquisition in memory to allow re-measuring/re-computing. To conclude the cal acquisition process, use the SENS:CORR:COLL:GUID:ABOR . command.

Learn all about Cal Sets.

**Note:** This command is NOT supported for application channels (Gain Compression, SMC/VMC, Noise Figure, IMD and so forth). Use SENS:CORR:COLL:GUID:SAVE and save to a cal register. You can then use SENS:CORR:CSET:COPY to copy the cal register to a named Cal Set.

- Use this command instead of specifying the optional name or GUID argument in SENS:CORR:COLL:GUID:INIT .
- Use SENS:CORRection:CSET commands to get names of existing Cal Sets.
- The cal data is also saved to the channel Cal Register.
- If all of the required standards have not been measured, the calibration will not complete properly.

**For Calibrate All Channels**

When this command is used during a Cal All session, the <cal set name> argument sets the User Cal Set prefix. All generated Cal Sets will be preceded with this string name.

- Cal Set prefix can also be set using SYST:CAL:ALL:CSET:PREFix . When the Cal Set prefix has already been set with SYST:CAL:ALL:CSET:PREFix , this command overwrites it.
- When <cal set name> is an empty string, a User Cal Set will not be saved. Only Cal Registers will be saved.

**Parameters**

- **<ch>** Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- **<cal set name>** _String_ - Name of an existing Cal Set to be overwritten.

See Calibrate All Channels note (above).
### SENSe<ch>:CORRection:COLLect:GUIDed:ETERms:LOAD[:CSET] <cset>,<calPort> [,csPort]

**Applicable Models:** All

(Write-only) Loads 1-port error terms from a Cal Set into the current Guided Cal sequence. When the Cal steps are recomputed, connection steps are removed due to the loading of the error terms.

This command must be sent after the INIT command. This command was implemented to facilitate calibrating a large matrix of external ports and most users will not need to use this command.

See example of how to use this command.

#### Parameters

- `<ch>` Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- `<cset>` **String** Name of User Cal Set in which the error terms reside.
- `<pnum>` **Integer** Port number of the current cal to receive error terms.
- `[csPort]` **Integer** Optional argument. Port number associated with the error terms in the Cal Set. If unspecified, the same port number as `<calPort>` is used.

### Examples

See example

### Query Syntax

Not Applicable

### Default

Not Applicable
steps, issue the acquisition commands, query the connection description strings, and subsequently complete a guided calibration. See example calibration programs.

**Parameters**

<ch> Channel to be calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

[string] Optional argument. Cal Set name or GUID enclosed in quotes.

If NOT specified, behavior depends on the SENS:CORR:PREFERENCE:CSET:SAVE setting.

If specified, choose an *existing* Cal Set, either by name or by GUID.

- By Cal Set name: include quotes.
- Query all Cal Set GUIDs with SENS:CORR:CSET:CAT?

An error is reported if the Cal Set is not found.

The Cal Set is either supplemented or overwritten depending on the method, connectors, and ports selected. Learn more about Cal Sets.

[bool] Optional argument. To set this argument, also set the first optional argument. See example below.

**OFF (0)** If Cal Set stimulus settings differ from the existing channel, do not change channel stimulus settings. The Cal Set is saved to the current setting of the SENS:CORR:PREF:CSET:SAVE command. This is the default setting if not specified.

**ON (1)** If Cal Set stimulus settings differ from the existing channel, change the channel stimulus settings to match the Cal Set settings.

[char] Optional argument. To set this argument, also set the first two optional arguments. See example below.

**SYNChronous** - blocks further SCPI commands while processing this command. (default setting).

**ASYNchronous** - does NOT block further SCPI commands while processing this command.

Learn more about this argument

**Examples**

`SENS:CORR:COLL:GUID:INIT`

' *set first optional argument*

Applicable Models: All

(Read-Write) Specifies amount to increment (increase) the channels averaging factor during measurement of isolation standards in a guided calibration.

Note: If the channel has averaging turned OFF and the value of <num> is greater than 1, averaging will be turned ON only during the isolation measurements and with the averaging factor equal to <num>.

Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<num> Amount to increment the averaging factor for the isolation measurement. The maximum averaging factor for the channel is 65536 (2^16).

Examples

'Measure isolation on all paths for the cal
SENSe:CORR:COLL:GUID:ISOL ALL

'Remove the port pairs 1-to-2 and 1-to-3 from the list of paths on which to measure isolation
SENSe:CORR:COLL:GUID:ISOL:PATHs REMove,1,2,1,3

Query Syntax


Return Type

Numeric

Default

8 - If this command is NOT sent, but isolation is measured, then averaging will be turned ON with factor set to 8 during the isolation measurements.
Applicable Models: All

(Read-Write) Specifies the paths (port pairs) to make isolation measurements on during a guided calibration.

Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<char> ALL Measure isolation on all pairings of the ports that are to be calibrated.

NONE Do not measure isolation on any pairing of the ports to be calibrated. (Default behavior).

ADD Add one or more specific pairings of ports to the list of port pairings for which isolation will be measured.

REMove Remove one or more specific pairings of ports from the list of port pairings for which isolation will be measured. If many paths are to be measured, it may be easier to first send ALL, then REMove and specify the paths to remove.

<p1a, p2a...> For use when <char> is ADD or REMove.

Specify Port numbers in pairs:

- For 3-port cals, specify up to 3 pairs.
- For 4-port cals, specify up to 6 pairs.

p1a, p1b (Path1 - port A and port B)
p2a, p2b (Path2 - port A and port B)
p3a, p3b (Path3 - port A and port B)

Examples

'Measure isolation on all paths for the cal
SENS:CORR:COLL:GUID:ISOL ALL

'Remove the port pairs 1-to-2 and 1-to-3 from the list of paths on which to measure isolation
sense:correction:collect:guided:isolation:paths REMove,1,2,1,3

Query Syntax
SENSe<ch>:CORRrection:COLLect:GUIDed:ISOLation:PATHs?

Note: if isolation is not be measured on any of the paths, the query returns 0.

**Applicable Models:** All

*(Read-only)* Designed to be used for an iterative cal standard such as a sliding load, this command returns the number of iterative measurement acquisitions that has been made for the specified step. Zero (0) is returned if the step has not yet been measured.

For most cal steps that have already been measured, this command returns 1.


**Parameters**

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<step> Guided Cal step number for which the acquisition number will be returned.

Use SENS:CORR:COLL:GUID:STEP? to query the number of steps in the calibration.

**Examples**

SENSe:CORR:COLL:GUID:ITER:COUNt? 4

'Example return:

5

See example program

**Return Type** Numeric

**Default** Not Applicable

Applicable Models: All

(Read-only) Designed to be used for an iterative cal standard such as a sliding load, this command returns the minimum number of required iterative measurement acquisitions for the specified step.

For most connection steps this will return 1, but for an iterative cal standard such as a sliding load, it will return a number such as 5.


Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<step> Guided Cal step number for which to return the number of iterative measurement acquisitions that have been made. Use SENS:CORR:COLL:GUID:STEP? to query the number of steps in the calibration.

Examples


'Example return:

5

See example program

Return Type Numeric

Default Not Applicable
Applicable Models: All

(Write-only) Resets the specified guided cal connection step as unmeasured. This clears all previous measurements made for that step.

**Parameters**

- `<ch>`: Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- `<step>`: Guided Cal step number to reset. Use SENS:CORR:COLL:GUID:STEP? to query the number of steps in the calibration.

**Examples**

```
```

See example program

**Return Type** Not Applicable

**Default** Not Applicable

---

**SENSe<ch>:CORRection:COLLect:GUIDed:LIST:COUNt?**

Applicable Models: All

(Read-only) Returns the number of measurement steps required to complete the current guided calibration. This command is the same as the SENS:CORR:COLL:GUID:STEP command.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1.

**Examples**

```
SENS:CORR:COLL:GUID:LIST:COUN?
sense2:correction:collect:guided:list:count?
```

**Return Type** Numeric

**Default** 0

---

**SENSe<ch>:CORRection:COLLect:GUIDed:LIST:STEP<ListCount>:COUNt?**
**Applicable Models:** All

(Read-only) Returns the number of standards for step[n]. This is generally 1 unless the standard is an isolation standard or a composite standard.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<ListCount>` Step number from 1 to 1000.

**Examples**

```
sense2:correction:collect:guided:list:step1:count?
```

**Return Type** Numeric

**Default** 1

---


**Applicable Models:** All

(Read-only) Returns the step description.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<ListCount>` Step number from 1 to 1000.

**Examples**

```
sense2:correction:collect:guided:list:step1:description?
```

**Return Type** String

**Default** Not Applicable

---

**SENS<ch>:CORRection:COLLect:GUIDed:LIST:STEP<ListCount>:LABel?**
(Read-only) Returns the label for the complete standard used in the step. If the standard is a composite standard, the label is for the composite device.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<ListCount>` Step number from 1 to 1000.

**Examples**

```
SENS:CORR:COLL:GUID:LIST:STEP:LAB?
sense2:correction:collect:guided:list:step1:label?
```

**Return Type** String

**Default** Not Applicable

---

**SENS<ch>:CORRection:COLLect:GUIDed:LIST:STEP<ListCount>:PORTs?**

(Read-only) Returns the number of ports on the standard used in the step. If the standard is a composite standard, the number of ports applies to the composite. So if the composite standard is an offset line connected to a load, the composite device is a 1 port device.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<ListCount>` Step number from 1 to 1000.

**Examples**

```
sense2:correction:collect:guided:list:step1:ports?
```

**Return Type** Numeric

**Default** Not Applicable

---

**SENS<ch>:CORRection:COLLect:GUIDed:LIST:STEP<ListCount>:STANdard<StandardCount>:**
Applicable Models: All

(Read-only) Returns the label for the one of the standards used in the step. If the step contains only a single standard, the response to this query is identical to SENS:CORR:COLL:GUID:LIST:STEP:LAB?

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<ListCount>` Step number from 1 to 1000.
- `<StandardCount>` Standard number from 1 to 3.

Examples

sense2:correction:collect:guided:list:step1:standard2:label?

Return Type String

Default Not Applicable


Applicable Models: All

(Read-only) Returns the number of ports on one of the standard used in the step. If the step contains only a single standard, the response to this query is identical to SENS:CORR:COLL:GUID:LIST:STEP:PORT?

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<ListCount>` Step number from 1 to 1000.
- `<StandardCount>` Standard number from 1 to 3.

Examples


Return Type Numeric

Default Not Applicable

Applicable Models: All

(Read-only) Returns the enumeration for the type of standard that describes one of the standard devices used in the step. If the step contains only a single standard, the response to this query is identical to SENS:CORR:COLL:GUID:LIST:STEP:STYP?

The following list of enumerations is currently defined:

OPEN | SHORt | LOAD | REFLection | THRU | LINE | ECAL | ISOLation | COMPosite | SENSor | PHASeref | MIXer

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<ListCount>` Step number from 1 to 1000.
- `<StandardCount>` Standard number from 1 to 3.

Examples

```
sense2:correction:collect:guided:list:step1:standard2:stype?
```

Return Type: Enumeration

Default: OPEN

---


Applicable Models: All

(Read-only) Returns the list of VNA test ports to which one of the standards is attached. If the step contains only a single standard, the response to this query is identical to SENS:CORR:COLL:GUID:LIST:STEP:TPORts?

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<ListCount>` Step number from 1 to 1000.
- `<StandardCount>` Standard number from 1 to 3.

Examples

```
sense2:correction:collect:guided:list:step1:standard2:tports?
```

Return Type: Numeric

Default: Not Applicable

---

**Applicable Models:** All

*(Read-only)* Returns the enumeration for the type of standard device used in the step.

The following list of enumerations is currently defined:

OPEN | SHORt | LOAD | REFLection | THRU | LINE | ECAL | ISOLation | COMPosite | SENSor | PHASeref | MIXer

**Parameters**

- `<ch>`  Any existing channel number. If unspecified, value is set to 1.
- `<ListCount>`  Step number from 1 to 1000.

**Examples**

```
sense2:correction:collect:guided:list:step1:stype?
```

**Return Type**  Enumeration

**Default**  OPEN

---


**Applicable Models:** All

*(Read-only)* Returns the list of VNA test ports to which the standard(s) in this step is attached.

**Parameters**

- `<ch>`  Any existing channel number. If unspecified, value is set to 1.
- `<ListCount>`  Step number from 1 to 1000.

**Examples**

```
sense2:correction:collect:guided:list:step1:tports?
```

**Return Type**  Numeric

**Default**  Not Applicable

---

**SENSe:CORRRection:COLLect:GUIDed:METHod <char>**  Superseded
Applicable Models: All


(Read-Write) Selects from one of several algorithms available for performing the THRU portion of a guided calibration. Learn more about THRU methods.

Parameters

<char>  
DEFAULT - Informs guided calibrations to use the default algorithm when computing the number of needed standards acquisition steps. (default selection if omitted.)

ADAP - Use the adapter removal algorithm

FLUSH - Use with insertable devices.

UNKN - Use the Unknown THRU algorithm with calibrations for non-insertable devices.

DEFINED - Use the THRU definition that you stored in the cal kit file, or ECal module.

TRL - Select TRL Cal Type for guided cals. Valid for "TRL ready" Cal Kits with properly assigned TRL cal classes.

SOLT - Select SOLT Cal Type for guided cals. Valid for any kit with properly assigned SOLT cal classes.

Examples

SENS:CORR:COLL:GUID:METH ADAP
sense:correction:collect:guided:method unkn

Query Syntax

SENSe:CORRection:COLLect:GUIDed:METHod?

Return Type

Character

Default

DEFAULT

SENS<ch>:CORRection:COLLect:GUIDed:PACQuire STAN<n>
(Write-only) Show the Cal Window, and optionally one or more other specific windows before acquiring a Cal standard. This command will cause the Cal Window to display the specific measurements that are to be made for that particular Cal standard to facilitate the connection of standards.

**Parameters**

- `<ch>`: Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- `STAN<n>`: Choose from: STAN1, STAN2, etc. through STANn where n is the number of cal standard connection steps for the calibration.

**Note:** You do not necessarily have to invoke these connection steps in sequential order.

**Examples**

- `SENS:CORR:COLL:GUID:PACquire STAN2`
- `sense:correction:collect:guided:pacquire STAN5`

See an example that uses this command.

**Query Syntax**

- Not Applicable

**Default**

- Not Applicable

---

```
SENS<ch>:CORRection:COLLect:GUIDed:PATH:CMEThod
<port1>,<port2>,<caltype1[,caltype2]>
```

**Applicable Models:** All

**Note:** This command replaces SENS:CORR:COLL:GUID:METH.

(Read-Write) Specifies the calibration method for each port pair.

**Note:** Sending this command will overwrite the VNAs SmartCal determinations for the most accurate cal method for your connector settings and Cal Kits. Send this command ONLY if you have a deliberate reason for overwriting the SmartCal logic. You can send the query form of this command to learn the cal method determined by SmartCal.

See Thru Pairs Sequence to learn how to send this and other Thru commands.

After sending this command, send the query form to be sure that the command was accepted. If not, then the chosen Cal method is not compatible with the specified Thru method. For example, if the specified Thru method is Unknown Thru, an attempt to set Enhanced Response Cal should be
Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<port1> First port of the pair to be calibrated.

<port2> Second port of the pair to be calibrated.

<caltype1[caltype2]> (String) Cal type for the port pair, enclosed in a single pair of quotes. NOT case-sensitive.

**caltype1** Choose from:

- TRL
- SOLT
- QSOLTN
- EnhRespN
- TransRespN

For the last two arguments, replace N with the port to be used as the source port, which MUST be one of the port pair.

**caltype2** Optional argument. Use only when performing an adapter removal cal on the pair. This argument specifies the Cal type on the second port. Caltype1 then specifies the Cal type of the first port.

Choose from the same arguments as caltype1.

Examples

```
SENS:CORR:COLL:GUID:PATH:CMEThod 2,3,"QSOLTN2"
sense:correction:collect:guided:path:cmethod 2,3,"solt,trl"
```

Query Syntax  
SENS<ch>:CORRection:COLLect:GUIDed:PATH:CMEThod? <port1>,<port2>

If only one caltype is returned then its NOT adapter removal.

**Return Type** String

**Default** The most accurate Cal method for the current cal.
SENSe<ch>:CORR:COLL:GUID:PATH:TMETHod
<port1>,<port2>,<thruType1[,thruType2]>

Applicable Models: All

Note: This command replaces SENS:CORR:COLL:GUID:METH.

(Read-Write) Specifies the calibration THRU method for each port pair.

Note: Sending this command will overwrite the VNAs SmartCal determination for the thru method. Send this command ONLY if you have a deliberate reason for overwriting the SmartCal logic. You can send the query form of this command to learn the THRU method determined by SmartCal.

See Thru Pairs Sequence to learn how to send this and other Thru commands.

Learn more about Thru methods.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;port1&gt;</td>
<td>First port of the port pair to be calibrated.</td>
</tr>
<tr>
<td>&lt;port2&gt;</td>
<td>Second port of the port pair to be calibrated.</td>
</tr>
<tr>
<td>&lt;thruType1[,thruType2]&gt;</td>
<td>(String) Thru methods for port pair, enclosed in a single pair of quotes. NOT case-sensitive.</td>
</tr>
</tbody>
</table>

thruType1  Calibration Thru method. Choose from:

- **Defined Thru**  Measures a Thru for which there is a stored definition in the Cal kit of the lowest-numbered port of the pair. For example, if the port pair is 1,2, then the cal kit for port 1 MUST contain a Defined Thru.

- **Zero Thru**  Measures a Zero length Thru, also known as Flush-Thru.

- **Undefined Thru** (Also known as Unknown Thru)  A Thru type for which there is NOT a stored definition in the Cal Kit. Valid ONLY for SOLT cal type.

- **Undefined Thru using a Defined Thru** (ECal modules ONLY) Measures the internal Thru as an Unknown Thru.

thruType2  Optional argument. Use ONLY when Adapter Removal Cal is specified for the pair using SENS:CORR:COLL:GUID:PATH:CMETHod. When specifying ThruType2, this is the only valid argument: "Defined Thru, Defined Thru"
Examples

```
SENS:CORR:COLL:GUID:PATH:TMETHod 2,3,"Zero Thru"
```

```
sense:correction:collect:guided:path:tmethod 2,3,"Defined Thru,Defined Thru"
```

Query Syntax

```
SENSe<ch>:CORRection:COLLect:GUIDed:PATH:TMETHod?
<port1>,<port2>
```

Always returns two parts:

- If the second part of the string is empty, adapter removal is NOT being performed.
- If the string is "Defined Thru, Defined Thru", adapter removal IS being performed.

**Return Type** String

**Default** The most accurate Thru method for the current cal.

**SENSe<ch>:CORRection:COLLect:GUIDed:PCAL:APPLy <bool>**

**Applicable Models:** All

(Read-Write) When enabled, and the following conditions are met, the user's existing source power calibration array will be used when acquiring calibration standard data:

- The user is performing a vector calibration only (no power).
- A valid source cal is present in the channel being calibrated.
- The user has elected to enable this feature (via GUI or SCPI).

**Note:** This command **MUST** be sent **AFTER** setting up the connectors and kits for the calibration session. Otherwise, the firmware can not determine which ports are going to be used in the calibration session. Therefore, it can not check if there is an applicable power calibration, because power calibrations are per port.

If this command is sent before setting connectors and kits, the following error is displayed:

```
+3068,"The request to use existing source cal array during calibration was refused."
```

**Parameters**

- `<ch>` Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
Choose from:

**OFF (0)** Do not use existing source power calibration array when acquiring calibration standard data:

**ON (1)** Use existing source power calibration array when acquiring calibration standard data:

---

**Examples**

```plaintext
SENS:CORR:COLL:GUID:PCAL:APPL 1
sense2:correction:collect:guided:pcal:apply 0
```

**Query Syntax**

SENS<ch>:CORR:COLL:GUID:PCAL:APPL?

**Return Type**

Boolean

**Default**

OFF

---

**SENS<ch>:CORR:COLL:GUID:PORTS?**

**Applicable Models:** All

*(Read-only)* Returns the list of ports being calibrated by an active calibration session.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.

**Examples**

```plaintext
SENS:CORR:COLL:GUID:PORT?
sense2:correction:collect:guided:ports?
```

**Return Type**

List of numbers

**Default**

Not Applicable

---

Applicable Models: All

(Read-Write) Specifies the behavior for guided cal steps that involve a sliding load in a cal that is about to be performed. Send this command BEFORE sending the Guided INIT command.

Although the term 'Preference' is used in the command, this is NOT a VNA preference. This setting does NOT survive instrument preset or reboot. It remains ONLY for the duration of the Guided Cal.

Parameters

<char> Behavior when measurements of sliding load are acquired. Choose from:

**DIALog** - The Sliding load dialog box appears when the acquire command is received for a sliding load step. All slide positions are measured (with a user-interface prompt) from a single invocation of the acquire command.

**ITERate** - Each invocation of the acquire command for a sliding load step measures a single slide position and increments the slide position counter. No Move Sliding Load prompt is presented on the VNA screen.

Examples

```
SENS:CORR:COLL:GUID:PREF:SLID ITER
```

See example program

Query Syntax

```
SENSe<ch>:CORRection:COLLect:GUIDed:PREFerence:SLIDingload?
```

Return Type

Character

Default

DIALog

SENS<ch>:CORRection:COLLect:GUIDed:SAVE[:IMMediate] [bool]

Applicable Models: All

(Write-only) Completes the guided cal by computing the error correction terms, turning Correction ON, and saving the calibration to a Cal Set. If all of the required standards have not been measured, the calibration will not complete properly.

Learn all about Cal Sets.

Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

[bool] Optional argument. If unspecified, the default behavior is the current VNA preference setting of SENS<ch>:CORRection:PREFerence:CSET:SAVE.

**OFF (0)**  Save cal data ONLY to a Cal Register.
ON (1) Save cal data to a Cal Register and a User Cal Set. The filename is automatically generated.

- For application channels (Gain Compression, SMC/VMC, Noise Figure, IMD and so forth), this command saves ONLY to a Cal Register. Use SENS:CORR:CSET:COPY to copy the cal register to a named calset.

- For a Calibrate All Channels session, this argument is ignored. Instead, use SYST:CAL:ALL:CSET:PREFix .

Examples

```
SENS:CORR:COLL:GUID:SAVE
sense2:correction:collect:guided:save:immediate 0
```

Query Syntax

Not Applicable

Default

Not Applicable

SENSe<ch>:CORRection:COLLect:GUIDed:SAVE:CSET <cal set name or guid>

Applicable Models: All

(Write-only) Completes the guided cal by computing the error correction terms, turning Correction ON, and saving the calibration to an existing, specified Cal Set. This command performs the same function as SENSe:CORRection:COLLect:GUIDed:SAVE , except this command allows the name or GUID of the Cal Set to be specified.

Learn all about Cal Sets.

Note: This command is NOT supported for application channels (Gain Compression, SMC/VMC, Noise Figure, IMD and so forth). Use SENS:CORR:COLL:GUID:SAVE and save to a cal register. You can then use SENS:CORR:CSET:COPY to copy the cal register to a named Cal Set.

- Use this command instead of specifying the optional name or GUID argument in SENS:CORR:COLL:GUID:INIT .
- Use SENS:CORRection:CSET commands to get names or GUIDs of existing Cal Sets.
- The cal data is also saved to the channel Cal Register.
- If all of the required standards have not been measured, the calibration will not complete properly.

For Calibrate All Channels

When this command is used during a Cal All session, the <cal set name> argument sets the User Cal Set prefix. All generated Cal Sets will be preceded with this string name.
- Cal Set prefix can also be set using SYST:CAL:ALL:CSET:PREFix. When the Cal Set prefix has already been set with SYST:CAL:ALL:CSET:PREFix, this command overwrites it.

- When <cal set name> is an empty string, a User Cal Set will not be saved. Only Cal Registers will be saved.

**Parameters**

- `<ch>`  Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- `<cal set name or guid>`  String - Name or GUID of an existing Cal Set to be overwritten. If specifying a GUID, curly brackets must be included.

**Examples**

```plaintext
sense:correction:collect:guided:save:cset "MyCalSet"
```

**Query Syntax**  Not Applicable

**Default**  Not Applicable

---

**SENSe<ch>:CORRection:COLLect:GUIDed:STATes[:CATalog]?: STAN<n>**

**Applicable Models:** All

(Read-only) Returns an array of ecal states used for the standard number.

**Parameters**

- `<ch>`  Any existing channel number. If unspecified, value is set to 1.
- `STAN<n>`  Choose from:STAN1, STAN2, etc. through STANn where n is the number of cal standard connection steps for the calibration.

**Note:** You do not necessarily have to invoke these connection steps in sequential order.

**Examples**

```plaintext
SENS:CORR:COLL:GUID:STAT? STAN1
```

**Return Type**  String

**Default**  Not Applicable

---

**SENSe<ch>:CORRection:COLLect:GUIDed:STEPs?**
Applicable Models: All

(Read-only) Returns the number of measurement steps required to complete the current guided calibration. This command is sent after the `SENS:CORR:COLL:GUID:INIT`, `SENS:CORR:COLL:GUID:CONN:PORT` and `SENS:CORR:COLL:GUID:CKIT:PORT` commands.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.

**Examples**

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SENS:CORR:COLL:GUID:STEP?</code></td>
<td></td>
</tr>
<tr>
<td><code>sense2:correction:collect:guided:steps?</code></td>
<td></td>
</tr>
</tbody>
</table>

**Return Type** Numeric

**Default** Not Applicable

---

SENSe<ch>:CORRection:COLLect:GUIDed:THRU:PORTs <t1a, t1b, t2a, t2b, t3a, t3b...>

Applicable Models: All

(Read-Write) For calibrating more than 2-ports ONLY. Specifies the port pairs for the Thru connections of the calibration. Send the query form of this command to learn the Thru pairs determined by SmartCal.

**Note:** Sending this command will overwrite the VNAs SmartCal determinations for the thru ports. Send this command ONLY if you have a deliberate reason for overwriting the SmartCal logic.

See Thru Pairs Sequence to learn how to send this and other Thru commands.

**Parameters**

- `<ch>` Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- `<t1a,t1b...>` Always specify port numbers in pairs: For example: 1,2 or 1,2,1,3
  - For 3-port cals, specify two or three pairs.
  - For 4-port cals, specify from three up to six pairs.

**Examples**

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SENS:CORR:COLL:GUID:THRU:PORT 1,2,1,3,1,4</code></td>
<td>'4-port measurement'</td>
</tr>
<tr>
<td><code>sense:correction:collect:guided:thru:ports 1,2,2,3</code></td>
<td>'3-port measurement'</td>
</tr>
</tbody>
</table>

**Query Syntax** `SENSe<ch>:CORRection:COLLect:GUIDed:THRU:PORTs?`

**Return Type** Numeric
**Default**  Port pairings that were used in the previous cal.

### THRU Pairs sequence

The SmartCal logic always determines the best calibration based on your specified connectors and ports.

The following three commands overwrite the SmartCal logic. Send these commands ONLY if you have a deliberate reason for overwriting the SmartCal logic.

- `sens:corr:coll:guid:THRU:PORTS <p1, p2>`
- `sens:corr:coll:guid:path:tmet <p1,p2, thrutype>`
- `sens:corr:coll:guid:path:cmet <p1,p2, calmethod>`

When sending one or more of these commands, they must be sent in the following sequence with the other commands listed here.

**Note:** The **GUID:INIT** command is sent before and after these commands.

1. `SENS:CORR:COLL:GUID:CONN:PORT(N)`
2. `SENS:CORR:COLL:GUID:CKIT:PORT (N)`
3. `SENS:CORR:COLL:GUID:INIT`
4. `SENS:CORR:COLL:GUID:THRU:PORTS <P1, P2>`
5. `SENS:CORR:COLL:GUID:PATH:TMET <P1,P2, THRUTYPE>`
6. `SENS:CORR:COLL:GUID:PATH:CMET <P1,P2, CALMETHOD>`
8. `SENS:CORR:COLL:GUID:INIT`

```plaintext
```
**Applicable Models:** N522xB, N523xB, N524xB

*(Write-only)* Initializes a cable repeatability characterization for the specified channel and port. Learn more about Dynamic Uncertainty.

The following existing commands are then used to perform the initialized repeatability or noise characterization:

SENSe<ch>:CORRection:COLLect:GUIDed:STEPs?


SENSe<ch>:CORRection:COLLect:GUIDed:ACQuire <stepNum>


SENSe<ch>:CORRection:COLLect:GUIDed:ITERations:RESet <stepNum>

SENSe<ch>:CORRection:COLLect:GUIDed:SAVE[:IMMediate]

SENSe<ch>:CORRection:COLLect:GUIDed:ABORt

**Parameters**

- **<ch>** Channel number of the characterization, depending on the CHAN:MODE setting. If unspecified, value is set to 1.
- **<pNum>** VNA port number on which the cable repeatability is to be performed.
- **<iterations>** Number of Iterative connections of the standards to be measured for the characterization.

**Examples**


See example program

**Query Syntax**

Not Applicable

**Default**

1

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Initializes a noise characterization for the specified channel and ports. Learn more about Dynamic Uncertainty.

The following existing commands are used to perform the initialized noise characterization:

SENSe<ch>:CORRection:COLLect:GUIDed:STEpS?


SENSe<ch>:CORRection:COLLect:GUIDed:ACQuire <stepNum>


SENSe<ch>:CORRection:COLLect:GUIDed:ITERations:RESet <stepNum>

SENSe<ch>:CORRection:COLLect:GUIDed:SAVE[:IMMediate]

SENSe<ch>:CORRection:COLLect:GUIDed:ABORt

Parameters

<ch> Channel number of the characterization, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<firstPort> First VNA port number on which the noise characterization is to be performed.

<secondPort> Second VNA port number on which the noise characterization is to be performed.

<iterations> Number of Iterative measurements to be made for each connected standard.

Examples

SENSe1:CORR:COLL:GUID:UNC:CHAR:NOIS 1,2,5

See example program

Query Syntax Not Applicable

Default 1

SENSe<ch>:CORRection:COLLect:GUIDed:UNCertainty[:ENABle] <bool>

Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the ON/OFF state which determines if the calibration that is about to be performed will support Dynamic Uncertainty for S-Parameters (Opt. S93015A/B).
Dynamic Uncertainty for S-Parameters is supported ONLY for calibrations on standard S-Parameter channels. Calibrations performed with that feature enabled do NOT support the use of ALL traditional GUIDed calibration commands.

Then these existing commands are used for performing calibration:

```
SENSe<ch>:CORRection:COLLect:GUIDed:INITiate[:IMMediate] [string][, bool][,char]
SENSe<ch>:CORRection:COLLect:GUIDed:STEPs?
SENSe<ch>:CORRection:COLLect:GUIDed:ACQuire <stepNum>
SENSe<ch>:CORRection:COLLect:GUIDed:SAVE[:IMMediate]
```

where these commands might also optionally be used in performing the calibration:

```
SENSe<ch>:CORRection:COLLect:GUIDed:ABORt
SENSe:CORRection:PREFerence:ECAL:ORIentation[:STATe] <ON|OFF>
SENSe:CORRection:PREFerence:ECAL:PMAP <module>, <string>
```

Dynamic Uncertainty must be enabled using this command before starting the calibration procedure because this command controls the way connectors and calkits are assigned to ports during calibration. Therefore, this command must be enabled before any of the following commands to ensure that the connector and calkit settings will be set/queried correctly:

```
SENSe<ch>:CORRection:COLLect:GUIDed:CKIT:CATalog?
SENSe<ch>:CORRection:COLLect:GUIDed:CONNector:CATalog?
```

**Parameters**
<bool> Enable ON/OFF state. Choose from:

**ON** or **1** – The next calibration INITialized for the channel will support Dynamic Uncertainties for S-Parameters.

**OFF** or **0** - - The next calibration INITialized for the channel will NOT support Dynamic Uncertainties for S-Parameters.

**Examples**

```plaintext
SENS1:CORR:COLL:GUID:UNC:ON

See example program
```

**Query Syntax**

`SENSe<ch>:CORRection:COLLect:GUIDed:UNCertainty[:ENABle]?`

**Return Type**

Boolean

**Default**

OFF
Controls an IMD and IMDx calibration. These commands supplement the Guided Cal commands.

<table>
<thead>
<tr>
<th>SENSE:CORRection:IMD</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALibration</td>
</tr>
<tr>
<td>FREQuencies</td>
</tr>
<tr>
<td>METHOD</td>
</tr>
<tr>
<td>LO</td>
</tr>
<tr>
<td>PCAL</td>
</tr>
<tr>
<td>MPRoduct</td>
</tr>
<tr>
<td>POWer</td>
</tr>
<tr>
<td>SENSor</td>
</tr>
<tr>
<td>CKIT</td>
</tr>
<tr>
<td>CONNector</td>
</tr>
<tr>
<td>SORDer:INCLude</td>
</tr>
</tbody>
</table>

Click on a keyword to view the command details.

Other IMD (Opt S93087A/B) commands

- CALCulate:MEASure:DEFine - creates an IMD measurement.
- Swept IMD
- IMSpectrum

See Also

- **Example** - Create and Cal an IMD measurement
- Learn about IMD Calibration
- Learn about Measurement Class
- Synchronizing the Analyzer and Controller
- SCPI Command Tree
SENSe<cnum>:CORRection:IMD:CALibration:FREQuencies <char>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the frequencies at which an IMD source power cal is performed.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <char> Choose from:

  - CENTer - Perform source power calibration at only the center frequencies, midway between the main tones.
  - ALL - Perform source power calibration at all main tone frequencies.

Examples

SENSe<channel>:CORRection:IMD:CALibration:FREQuencies ALL

Query Syntax

SENSe<cnum>:CORRection:IMD:CALibration:FREQuencies?

Return Type

Character

Default

CENTer

SENSe<cnum>:CORRection:IMD:CALibration:METHod <char>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the method by which the match-correction portion of an IMD calibration is performed. Learn more.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <char> Choose from:

  - MATCH - Performs a full 2-port cal for full match-correction.
  - RESPONSE - Performs only a response (normalization) cal instead of a full 2-port cal.

Examples

SENSe<channel>:CORRection:IMD:CALibration:METHod MATCH

Query Syntax

SENSe<cnum>:CORRection:IMD:CALibration:METHod?

Return Type

Character
**SENSe<cnum>[:CORRection]: IMD:LO<n>:PCAL[:STATe] <bool>**

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and returns whether or not the LO power cal step is included in the cal steps when an IMDX cal is performed.  Learn more.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` LO Stage. Choose 1.
- `<char>` LO Power Cal state. Choose from:
  - **O** or **OFF** - Skips over the LO Power Cal when calibrating.
  - **1** or **ON** - Includes a step for LO Power Cal when calibrating

**Examples**

```plaintext
SENSe:CORR:IMD:LO1:PCAL 0
sense2:correction:imd:lo1:pcal:state on
```

**Query Syntax**

SENSe<cnum>:CORRection:IMD:LO<n>:PCAL[:STATe]?

**Return Type**

Boolean

**Default** 0 or OFF

**SENSe<cnum>[:CORRection]: IMD:MPRoduct <num>**

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and returns the maximum intermod product frequencies to be calibrated. All lower product frequencies are also calibrated.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<char>` Maximum IM products to calibrate. Choose from:
  - **2** - second order products
  - **3** - third order products
  - **5** - fifth order products
7 - seventh order products
9 - ninth order products

**Examples**

```
SENS:CORR:IMD:MPR 5
sense2:correction:imd:mproduct 9
```

**Query Syntax**

`SENSe<cnum>::CORRection:IMD:MPRoduct?`

**Return Type**

Numeric

**Default**

3

---

**SENSe<cnum>::CORRection:IMD:POWer <num>**

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and returns the power level at which to perform the source power calibration using a power sensor.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<num>`: Power level. Choose a value between the min and max power level of the VNA.

**Examples**

```
SENS:CORR:IMD:POW -5
sense2:correction:imd:power 5
```

**Query Syntax**

`SENSe<cnum>::CORRection:IMD:POWer?`

**Return Type**

Numeric

**Default**

0

---

**SENSe<cnum>::CORRection:IMD:SENSor:CKIT <string>**
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the cal kit to be used for calibrating at the port 1 reference plane when the power sensor connector is different from the DUT port 1. This effectively removes the effects of an adapter that is used to connect the power sensor.

**Parameters**

- **<cnum>** Any existing channel number; if unspecified, value is set to 1.
- **<string>** Cal Kit enclosed in quotes. First set the DUT connector for port 1 and the connector of the power sensor. Then use `SENS:CORR:COLL:GUID:CKIT:PORT1:CAT?` to return a list of valid cal kits.

**Examples**

```
SENS:CORR:IMD:SENS:CKIT "85052B"
```

**Query Syntax**

`SENSe<cnum>::CORRection:IMD:SENSor:CKIT?`

**Return Type** String

**Default** Depends on the specified connectors

---

SENSe<cnum>::CORRection:IMD:SENSor:CONNector <string>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the power sensor connector type which is used to perform the Source Power Cal portion of an IMD calibration.

**Parameters**

- **<cnum>** Any existing channel number; if unspecified, value is set to 1.

**Examples**

```
SENS:CORR:IMD:SENS:CONN "APC 3.5 male"
sense2:correction:imd:sensor:connector "Ignored"
```

**Query Syntax**

`SENSe<cnum>::CORRection:IMD:SENSor:CONNector?`

**Return Type** String

**Default** "Ignored"

---

SENSe<cnum>::CORRection:IMD:SORDer:INCLude <bool>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns whether to include the second order products in the calibration. These frequencies of these products can be far from the main tones.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<bool>` Choose from:
  - **ON** (or **1**) - Include 2nd order products
  - **OFF** (or **0**) - Do NOT include 2nd order products

Examples

```
SENS:CORR:IMD:SORD:INCL ON
```

Query Syntax

```
SENSe<cnum>::CORRection:IMD:SORDer:INCLUDe?
```

Return Type

Boolean

Default

OFF or 0
Note: These commands are replaced with Sense:Corr:Guided commands.

The commands in this topic are common to perform both SMC and VMC calibrations.

A **calibration session** is a term used to describe an instance of a SMC or VMC calibration. The session number is chosen in the SENSE:CORR:COLL:SESS:INITiate command. All other commands refer to that session number. For more commands, see SESS:SMC and SESS:VMC.

Commands to read (**STEP?**) and describe (**DESC?**) each step are provided to facilitate a remote user interface.

![Command Tree](image)

Click on a red keyword to view the command details.

**See Also**

- Learn about SMC and VMC calibrations
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

SENSe<ch>:CORRection:COLLect:SESSion<n>:ACQuire <step>[,sync]
Applicable Models: N522xB, N524xB

(Write only) Acquire a calibration measurement.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` Session number of the calibration.
  
  Learn about Cal sessions.

- `<step>` Step number to acquire. Use `SENS:CORR:COLL:SESS:STEPS?` to find the number of steps required for the calibration.

  [sync] Optional argument. Choose from:

  - **SYNChronous** - blocks SCPI commands during standard measurement (default behavior).
  - **ASYNchronous** - does NOT block SCPI commands during standard measurement.

  Learn more about this argument

Examples

```
SENSe2:CORR:COLL:SESS6:ACQ 5,ASYN;*OPC?
```

Query Syntax

Not Applicable

Default

Not Applicable

---

SENSe<ch>:CORRection:COLLect:SESSION<n>:CKIT:PORT<p>:CATalog?

Applicable Models: N522xB, N524xB

(Read only) Returns a list of cal kits that are compatible with the connector on port `<p>`. The port connector type is set with `SENS:CORR:COLL:SESS:CONN:PORT`.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` Session number of the calibration.
  
  Learn about Cal sessions.

- `<p>` VNA port number connector to query for compatible cal kits.
  
  For VMC, output port of the calibration mixer, specify 3 unless already used for the output of the mixer. Otherwise, specify 4.
Examples


Return Type
Comma separated string values

Default
Not Applicable


Applicable Models: N522xB, N524xB

(Read-Write) Set or return the Cal Kit for the specified port. Use SENS:CORR:COLL:SESS:CKIT:PORT:CAT? to list compatible Cal Kits.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

Learn about Cal sessions.

<p> VNA port number connector for which a cal kit is to be specified.

For VMC, output port of the calibration mixer, specify 3 unless already used for the output of the mixer. Otherwise, specify 4.

<calkit> Cal Kit Name

Examples

SENS:CORR:COLL:SESS:CKIT:PORT:SEL 85091A

Query Syntax

Return Type
String

Default
Not Applicable

Applicable Models: N522xB, N524xB

(Read-Write) Set the connector type and sex for the specified port number. Catalog valid connector types using `SENS:CORR:COLL:GUID:CONN:CAT?`.

Parameters
<ch> Any existing channel number. If unspecified, value is set to 1
<n> Session number of the calibration. Learn about Cal sessions.
<p> VNA port number connector for which to specify a connector type.
$conn$ Name of the connector type

Examples `SENS2:CORR:COLL:SESS6:CONN:PORT1:SEL "N Type"`

Query Syntax `SENSe<ch>:CORR:COLL:SESS<n>:CONN:PORT<p>[:SELect]?`

Return Type String
Default Not Applicable

SENSe<ch>:CORR:COLL:SESS<n>:DESC? <step>

Applicable Models: N522xB, N524xB

(Read-only) Returns the connection prompt for the step. List the number of steps in the calibration using `SENS:CORR:COLL:SESS:STEPS?`.

Parameters
<ch> Any existing channel number. If unspecified, value is set to 1
<n> Session number of the calibration. Learn about Cal sessions.
<step> Step number


Return Type Numeric
Default Not Applicable

SENSe<ch>:CORR:COLL:SESS<n>:DONE
Applicable Models: N522xB, N524xB

(Write only) Ends the calibration session. First use SAVE? to calculate error terms and save the CalSet.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1
<n> Session number of the calibration.

Examples

SENS1:CORR:COLL:SESS6:SAVE?
SENS1:CORR:COLL:SESS6:DONE

Query Syntax

Not Applicable

Default

Not Applicable

SENSe<ch>:CORRection:COLLect:SESSion<n>:INITiate <string>

Applicable Models: N522xB, N524xB

(Write only) Initiates an SMC or VMC calibration session. Use the session number for subsequent SMC or VMC commands.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1
<n> Session number. Choose from 1 to the max number of channels. If the session number already exists it will be terminated and a new session initiated.
<string> Name of the calibration. Choose from:

"VMC" or "VectorMixerCal.VMCType"
"SMC" or "ScalarMixerCal.SMCType"

Examples

SENS1:CORR:COLL:SESS6:INITiate "VectorMixerCal.VMCType"

Query Syntax

Not Applicable

Default

Not Applicable

SENSe<ch>:CORRection:COLLect:SESSion<n>:SAVE?
Applicable Models: N522xB, N524xB

(Read only) Finish the SMC or VMC calibration, compute error terms, populate and save the Cal Set, and return the GUID of the Cal Set.

Note: The destination (Cal Register or User Cal Set) is determined by the setting of the SENS:CORR:PREFerence:CSET:SAVE command.

Parameters
- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.

Examples
SENSe1:CORRection:COLLect:SESS6:SAVE?

Return Type
String specifying the GUID of the Cal Set produced by this session.

Default
Not Applicable

SENSe[ch>:CORRection:COLLect:SESSion<n>:STEPs?

Applicable Models: N522xB, N524xB

(Write-Read) Returns the number of steps required by the Calibration.

To ensure this query always completes successfully, first send the write command: SENS:CORR:COLL:SESS:STEP without the `<n>` argument, then send the query.

Parameters
- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.

Examples
SENSe1:CORR:COLL:SESS6:STEP?

Return Type
Numeric

Default
Not Applicable
Note: These commands (commonly known as "Session" commands) were replaced with Sens:Corr:Coll:Guid:SMC commands.

Performs scalar (SMC) calibration on a frequency converting device.

<table>
<thead>
<tr>
<th>SENSE:CORRectionCOLLect:SESSion:SMC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECAL</td>
</tr>
<tr>
<td>CHARacterza</td>
</tr>
<tr>
<td>FSIMulator</td>
</tr>
<tr>
<td>NETWork</td>
</tr>
<tr>
<td>FILeName</td>
</tr>
<tr>
<td>MODE</td>
</tr>
<tr>
<td>IMPort</td>
</tr>
<tr>
<td>PHASE</td>
</tr>
<tr>
<td>DELay</td>
</tr>
<tr>
<td>METHod</td>
</tr>
<tr>
<td>MIXer</td>
</tr>
<tr>
<td>PWRCal</td>
</tr>
<tr>
<td>SEParate</td>
</tr>
<tr>
<td>SRCPort</td>
</tr>
<tr>
<td>TWOPort</td>
</tr>
<tr>
<td>ECAL</td>
</tr>
<tr>
<td>ORIentation</td>
</tr>
<tr>
<td>[STATE]</td>
</tr>
<tr>
<td>PORTmap</td>
</tr>
<tr>
<td>METHod</td>
</tr>
<tr>
<td>OMITisolat</td>
</tr>
<tr>
<td>OPTion</td>
</tr>
</tbody>
</table>
Click on a red keyword to view the command details.

Red keywords are obsolete.

See Also

- Example Programs
- Learn about SMC Calibrations
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**NOTE:** To configure a power meter and sensor see `SOURce:POWer:` commands.

```
SENSe<ch>:CORRection:COLLect:SESSion<n >:SMC:ECAL:CHARacteriza <mod> ,<char>
```

**Applicable Models:** N522xB, N523xB, N524xB

(Read-Write) Specifies the ECal module and characterization to be used for the SMC calibration.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration. Learn about Cal sessions.
- `<mod>` 1 - Electronic Calibration Module
- `<char>` Specifies which characterization within the ECal module from which to read the confidence data.

- 0 Factory characterization (data that was stored in the ECal module by Keysight). Default if not specified.
- 1 User characterization #1
- 2 User characterization #2
- 3 User characterization #3
- 4 User characterization #4
- 5 User characterization #5
### Examples

SENS:CORR:COLL:SESS:SMC:ECAL:CHAR 1,2

### Query Syntax


### Return Type

*Default:* Numeric

### Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration. Learn about Cal sessions.
- `<p>` Apply network to input or output of mixer. Choose from:
  - 1 - Input of mixer
  - 2 - Output of mixer
- `<string>` Filename of the S2P used for embedding or de-embedding. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

### Applicable Models:

N522xB, N523xB, N524xB

### (Read-Write) Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement. Learn more.

### Examples

SENS:CORR:COLL:SESS:SMC:FSIM:NETW1:FIL "c:\users\public\network analyzer\documents/WaveguideAdapt.S2P"

### Query Syntax

SENS<ch>:CORRection:COLLect:SESSion<n>:SMC:FSIMulator:NETWork<x>:FILename?

### Return Type

*Default:* String

**Applicable Models:** N522xB, N523xB, N524xB

*(Read-Write)* Allows you to embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement. [Learn more.]

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration. [Learn about Cal sessions.]
- `<p>` Apply network to input or output of mixer. Choose from:
  - 1 - Input of mixer
  - 2 - Output of mixer
- `<char>` Choose from:
  - **NONE** - Do nothing with effects of S2P file.
  - **EMBed** - Add effects of S2P file from the measurement results.
  - **DEEMbed** - Remove effects of S2P file from the measurement results.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
</table>

**Query Syntax**


**Examples**

SENSe<ch>:CORRection:COLLect:SESSION<n>:SMC:IMPort <calName>, <dataset>
Applicable Models: N522xB, N523xB, N524xB

(Write-only) Imports existing Source Power Cal data into the SMC calibration.

**Parameters**

- `<ch>`  Any existing channel number. If unspecified, value is set to 1
- `<n>`  Session number of the calibration.

  Learn about Cal sessions.

- `<calName>`  (String) Name of existing Cal Set from which power meter data is imported.
- `<dataset>`  (String) Name of the data set. Use POWER_STEP

**Examples**

```
```

**Query Syntax** Not Applicable

**Default** NONE

---

SENSe<ch>:CORRection:COLLect:SESSion<n>:SMC:PHASe:DELay <num>

**Applicable Models:** N522xB, N523xB, N524xB

(Read-Write) Set and return the known delay through the calibration mixer. Learn more.

**Parameters**

- `<ch>`  Any existing channel number. If unspecified, value is set to 1
- `<n>`  Session number of the calibration.

  Learn about Cal sessions.

- `<char>`  Known delay through the calibration mixer in seconds.

**Examples**

```
```

**Query Syntax** SENSe<ch>:CORRection:COLLect:SESSion<n>:SMC:PHASe:DELay?

**Return Type** Numeric

**Default** 0 seconds

---

SENSe<ch>:CORRection:COLLect:SESSion<n>:SMC:PHASe:METHod <char>
Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Set and return the method of setting the delay through the calibration mixer. Learn more.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.
  
  Learn about Cal sessions.

- `<char>` Choose from:
  
  
  - **MIXer** - use the S2P file set with `SENS:CORR:COLL:SESS:SMC:PHAS:MIX`

Examples


Query Syntax

`SENSe<ch>:CORRection:COLLect:SESSion<n>:SMC:PHASe:METHod?`

Return Type

- **Character**
  
  Default: **FIXed**

---

`SENSe<ch>:CORRection:COLLect:SESSion<n>:SMC:PHASe:MIXer <string>`

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Set the filename of the S2P file used to characterize the calibration mixer. Learn more.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.
  
  Learn about Cal sessions.

- `<string>` Calibration mixer filename. Use the following rules to specify path names:
  
  - The default folder is "c:\users\public\network analyzer\documents"
  
  - You can change the active directory using `MMEMory:CDIRectory`.
  
  - Specify only the file name if using the active directory.
  
  - You can also use an absolute path name to specify the folder and file.
SENSe<ch>:CORRection:COLLect:SESSion<n>:SMC:PWRCal:SEParate <bool>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Specifies whether to use a Thru standard or to use two power sensor connections during the power cal of an SMC calibration.

This command must be sent BEFORE the INITiate command and all the other calibration commands.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1
<n> Session number of the calibration.

Examples

SENS:CORR:COLL:SESS:INIT "SMC"
SENS:CORR:COLL:SESS:SMC:PWRC:SEP 1

Query Syntax


Return Type

Boolean

Default

0 or OFF

SENSe<ch>:CORRection:COLLect:SESSion<n>:SMC:PWRCal:SRCPort <string> Obsolete
Specifies which port to calibrate.

**Note:** Beginning with Rev 6.0, this command is no longer necessary. Because of improved calibration techniques, **Both** is always selected although a power meter measurement is performed only on port 1.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.

Learn about Cal sessions.

- `<char>`
  - '1' Source port 1 (SMC forward direction)
  - '2' Source port 2 (SMC reverse direction)
  - 'BOTH' (both forward and reverse directions)

**Examples**

```
SENS:CORR:COLL:SESS:SMC:PWRC:SRCP 'both'
```

**Query Syntax**


**Return Type** String

**Default** 1

---


**Applicable Models:** N522xB, N523xB, N524xB

Sets ECAL Auto-Orientation ON or OFF. If setting auto-orientation OFF, you must manually specify the orientation of the ECAL module with `SENS:CORR:COLL:SESS:SMC:TWOP:ECAL:PORTmap`.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.

Learn about Cal sessions.

- `<bool>`
  - OFF or 0 = Orientation OFF
  - ON or 1 = Orientation ON

**Examples**

```
```

**Query Syntax**

**SENSe<ch>:CORRection:COLLect:SESSION<n>:SMC:TWOPort:ECAL:PORTmap <mod>, <string>**

**Applicable Models:** N522xB, N523xB, N524xB

*(Read-Write)* Specifies the manual orientation (which ports of the module are connected to which ports of the VNA) when auto-orientation is OFF.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.
  
  Learn about Cal sessions.
- `<mod>` 1 - Electronic Calibration Module
- `<string>` Format in the following manner:
  
  Aw,Bx,Cy,Dz

  where

  - A, B, C, and D are literal ports on the ECAL module
  - w,x,y, and z are substituted for VNA port numbers to which the ECAL module port is connected.

  Ports of the module which are not used are omitted from the string.

**For example**, on a 4-port ECAL module with

- port A connected to VNA port 2
- port B connected to VNA port 3
- port C not connected
- port D connected to VNA port 1

the string would be: A2,B3,D1
If either the receive port or source port (or load port for 2-port cal) of the CALC:PAR:SELected measurement is not in this string and orientation is OFF, an attempt to perform an ECal calibration will fail.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;A1,B2&quot;</td>
<td>Default</td>
</tr>
</tbody>
</table>

Example Syntax

<table>
<thead>
<tr>
<th>Examples</th>
<th>Query Syntax</th>
</tr>
</thead>
</table>

**SENSe<ch>:CORRection:COLLect:SESSion<n >:SMC:TWOPort:METHod <string>**

**Applicable Models:** N522xB, N523xB, N524xB

*(Read-Write)* Specifies the guided ECal method for performing the thru portion of the calibration.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.
  
  Learn about Cal sessions.
- `<string>` ECAL Method: Choose from:
  
  - 'DEFAULT' - Default
  - 'ADAP' - Adapter removal
  - 'FLUSH' - Flush Through
  - 'UNKN' - Unknown Thru

**Examples**

<table>
<thead>
<tr>
<th>Examples</th>
<th>Query Syntax</th>
</tr>
</thead>
</table>

**Query Syntax**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;A1,B2&quot;</td>
<td>Default</td>
</tr>
</tbody>
</table>

**SENSe<ch>:CORRection:COLLect:SESSion<n >:SMC:TWOPort:OMITisolat <bool>**
Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Select to omit or perform the isolation portion of the ECAL.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

<bool> ON or 1 - Omit isolation

OFF or 0 - Perform isolation

Examples

SENS:CORR:COLL:SESS:SMC:TWOPort:OMIT 1

Query Syntax


Return Type

Boolean

Default

1

SENSe<ch>:CORRection:COLLect:SESSion<n >:SMC:TWOPort:OPTion <string>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets the SMC calibration to ECAL or MECHanical

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

Learn about Cal sessions.

<char> Choose from:

'ECAL' Electronic Calibration Module -

Note: This selection assumes there is only one ECal module on the USB and so it selects the first enumerated module on the bus, and the factory characterization on that ECal module, to be used for the cal.

'MECH' Mechanical Calibration Kit

Examples

SENS:CORR:COLL:SESS:SMC:TWOPort:OPTion 'ECAL'

Query Syntax

SENS:CORR:COLL:SESS:SMC:TWOPort:OPTion?
Return Type  String
Default     ECAL
Note: These commands (commonly known as "Session" commands) were replaced with Sens:Corr:Coll:Guid:VMC commands.

Performs a vector (VMC) calibration on a frequency converting device.

SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:FSIMulator:NETWork<x>:MODE <char>

See Also

- Example Programs
- Learn about VMC Calibration
- Synchronizing the Analyzer and Controller
Applicable Models: N522xB, N524xB

(Read-Write) Allows you to embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement. Learn more.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

Learn about Cal sessions.

<x> Apply network to input or output of mixer. Choose from:

1 - Input of mixer
2 - Output of mixer

<char> Choose from:

NONE - Do nothing with effects of S2P file.

EMBed - Add effects of S2P file from the measurement results.

DEEMbed - Remove effects of S2P file from the measurement results.

Examples


Query Syntax

SENS<ch>:CORRection:COLLect:SESSion<n>:VMC:FSIMulator:NETWork<x>:MODE?

Return Type Character

Default NONE

SENS<ch>:CORRection:COLLect:SESSion<n>:VMC:FSIMulator:NETWork<x>:Filename <string>
Applicable Models: N522xB, N524xB

(Read-Write) Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement. Learn more.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration. Learn about Cal sessions.
- `<x>` Apply network to input or output of mixer. Choose from:
  - 1 - Input of mixer
  - 2 - Output of mixer
- `<string>` Filename of the S2P used for embedding or de-embedding. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

Examples

```
```

Query Syntax

```
SENS<ch>:CORRection:COLLect:SESSion<n>
:VMC:FSIMulator:NETWork<x>:FILename?
```

Return Type

String

Default

Not Applicable

---


Applicable Models: N522xB, N524xB

(Read-Write) Specifies the .S2P filename used for mixer characterization. Use the VMC:MIXer:CHARacterize:CAL: OPTion command to load the file for mixer characterization. Once loaded, use this command to query the current filename or set a new filename.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration. Learn about Cal sessions.
- `<string>` Filename of the S2P used for mixer characterization. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets the mixer characterization method to ECal, Mechanical, or read from a file.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.

*Learn about Cal sessions.*

- `<char>`
  - **ECAL** - Electronic Calibration Module
  - **MECH** - Mechanical Calibration Kit
  - **FILE, <filename>** - Retrieve a mixer characterization file. Also specify the filename of the S2P used for mixer characterization. Use the full path name, file name, and .S2P suffix. Use the `VMC:CHARacterize:CAL:FILename` command to query the filename..

**Examples**

```

'or

FILE,'c:\users\public\network analyzer\documents/Mixer001.s2p'
```

```
the filename
```

**Query Syntax**


**Return Type**

String

**Default**

MECH
Applicable Models: N522xB, N524xB

(Read-Write) Specifies the direction in which to characterize the calibration mixer. Learn more about the calibration mixer.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration. Learn about Cal sessions.
- `<bool>`
  - OFF (0) - Characterize the calibration mixer in the SAME direction as that specified in the mixer setup.
  - ON (1) - Characterize the calibration mixer in the REVERSE direction as that specified in the mixer setup.

Examples

```
```

Query Syntax

```
```

Return Type

- Boolean

Default

- OFF

SENS<ch>:CORRection:COLLect:SESSion<n> :VMC:MIXer:ECAL:CHARacteriza <mod>,<char>

Applicable Models: N522xB, N524xB

(Read-Write) Specifies the ECal module and characterization to be used for the mixer characterization portion of the calibration.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration. Learn about Cal sessions.
- `<mod>` 1 - Electronic Calibration Module
- `<char>` Characterization number in the specified ECAL module. Choose from:
  - 0 Factory characterization (data that was stored in the ECAL module by Keysight). Default if not specified.
  - 1 User characterization #1
2  User characterization #2
3  User characterization #3
4  User characterization #4
5  User characterization #5

Examples
SENS:CORR:COLL:SESS:VMC:MIX:ECAL:CHAR 1,0

Query Syntax

Return Type
Numeric
Default
1,0

SENSe<ch>:CORRection:COLLect:SESSion<n>:VMC:OPERation <string>

Applicable Models: N522xB, N524xB

(Read-Write) Sets the port mapping for the mixer characterization with ECAL. This command is required if SENS:CORR:COLL:SESS:VMC:MIX:CHAR:CAL:OPT ECAL is specified.

Parameters
<ch>  Any existing channel number. If unspecified, value is set to 1
<n>  Session number of the calibration.

Learn about Cal sessions.

<mod>  1 - Electronic Calibration Module
<string>  Choose from:

"A1" - ECAL module port A is connected to VNA port 1
"B1" - ECAL module port B is connected to VNA port 1

Examples
SENS:CORR:COLL:SESS:VMC:PORT?

Query Syntax
SENS:CORR:COLL:SESS:VMC:MIX:PORTmap?

Return Type
String
Default
"A1"
Applicable Models: N522xB, N524xB

(Read-Write) Perform either full VMC calibration or mixer characterization only.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.
- `<char>` 'CAL' - full calibration and mixer characterization
  'CHAR' - mixer characterization only (no reference mixer required) - Saves an .S2P file with the filename specified in `SENS<ch>:CORR:COLL:SESSION<n>:VMC:CHARacterize:CAL:FILENAME <filename>`. If none is specified, a filename is automatically generated and can be queried using the filename command.

**Examples**

```
SENS:CORR:COLL:SESSION<n>:VMC:OPER 'CAL'
```

**Query Syntax**

`SENS:CORR:COLL:SESSION<n>:VMC:OPER?`

**Return Type**

String

**Default**

CAL

---


Applicable Models: N522xB, N524xB

(Read-Write) Specifies the ECal module and characterization to be used for the VMC calibration.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.
- `<mod>` 1 - Electronic Calibration Module
- `<char>` Characterization number in the specified ECAL module. Choose from:
  - 0 Factory characterization (data that was stored in the ECal module by Keysight). Default if not specified.
  - 1 User characterization #1

Applicable Models: N522xB, N524xB

(Read-Write) Sets ECAL orientation for the VMC ECal.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.
  
  Learn about Cal sessions.

- <bool> ON or 1 - Perform orientation
- OFF or 0 - Do NOT perform orientation

Examples

Usage:

```
SENSe:CORR:COLL:SESS:VMC:TWOP:ECAL:CHAR 1,1
```

Query Syntax

```
```

Return Type

Integer

Default

```
1,0
```
**Applicable Models:** N522xB, N524xB

*(Read-Write)* Specifies the manual orientation (which ports of the module are connected to which ports of the VNA) when orientation is turned off.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration. Learn about Cal sessions.
- `<mod>` 1 - Electronic Calibration Module
- `<string>` Port Map, formatted in the following manner:

  \[Aw,Bx,Cy,Dz\]

  where:

  - A, B, C, and D are literal ports on the ECAL module.
  - w, x, y, z are substituted for VNA port numbers to which the ECAL module port is connected.
  - Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECal module with:

- port A connected to VNA port 2
- port B connected to VNA port 3
- port C not connected
- port D connected to VNA port 1

the string would be: A2,B3,D1

If either the receive port or source port (or load port for 2-port cal) of the measurement is not in this string and orientation is OFF, an attempt to perform an ECal will fail.

**Examples**

```
```

**Query Syntax**


**Return Type**

string

**Default**

"A1,B2"
SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:TWOPort:METHod <string>

Applicable Models: N522xB, N524xB

(Read-Write) Specifies the guided ECAL method for performing the thru portion of the calibration.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.

Learn about Cal sessions.

- `<char>`
  - 'DEFAULT' - Default
  - 'ADAP' - Adapter removal
  - 'FLUSH' - Flush Through
  - 'UNKN' - Unknown Thru

**Examples**

SENSe:CORR:COLL:SESS:VMC:TWOP:METH 'ADAP'
SENSe2:CORR:COLL:SESSion6:VMC:TWOPort:METHod 'FLUSH'

**Query Syntax**

SENSe:CORR:COLL:SESS:VMC:TWOP:METH?

**Return Type**

- String
- Default: DEFAULT

---

SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:TWOPort:OMITisolat <bool>

Applicable Models: N522xB, N524xB

(Read-Write) Select to omit or perform the isolation portion of the ECAL.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number of the calibration.

Learn about Cal sessions.

- `<bool>`
  - ON or 1 - omit isolation
  - OFF or 0 - perform isolation

**Examples**

SENSe:CORR:COLL:SESS:VMC:TWOP:OMIT 1
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>ON</td>
</tr>
</tbody>
</table>

SENSe<ch>:CORRection:COLLect:SESSion<n>:VMC:TWOPort:OPTion <string>

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets the 2-port calibration option to ECAL or MEChanical

**Parameters**

<table>
<thead>
<tr>
<th>&lt;ch&gt;</th>
<th>Any existing channel number. If unspecified, value is set to 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;n&gt;</td>
<td>Session number of the calibration.</td>
</tr>
</tbody>
</table>

Learn about Cal sessions.

<table>
<thead>
<tr>
<th>&lt;char&gt;</th>
<th>Choose from:</th>
</tr>
</thead>
<tbody>
<tr>
<td>'ECAL'</td>
<td>Electronic Calibration Module</td>
</tr>
<tr>
<td>'MECH'</td>
<td>Mechanical Calibration Kit</td>
</tr>
</tbody>
</table>

**Examples**

```
SENSe:CORR:COLL:SESS:VMC:TWOP:OPT 'MECH'
SENSe2:CORR:COLL:SESSION6:VMC:TWOPort:OPTion 'ECAL'
```

**Query Syntax** SENS:CORR:COLL:SESS:VMC:TWOP:OPT?

**Return Type** String

**Default** "MECH"
Sense: Couple Commands

**SENSe:COUPle**

| PARAmeter | [STATe] |

Click on a keyword to view the command details.

See Also

- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

---

**SENSe<cnum>:COUPle <ALL | NONE>**

**Applicable Models:** All

*(Read-Write)* Sets the sweep mode as Chopped or Alternate.

**Learn about Alternate Sweep**

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<ALL | NONE>`
  - **ALL** - Sweep mode set to Chopped - reflection and transmission measured on the same sweep.
  - **NONE** - Sweep mode set to Alternate - reflection and transmission measured on separate sweeps. Improves Mixer bounce and Isolation measurements. Increases sweep time.

**Examples**

```
SENS:COUP ALL
sense2:couple none
```

**Query Syntax**

SENSe<cnum>:COUPle?

**Return Type**

Character

**Default**

ALL

---

**SENSe<cnum>:COUPle:PARameter[:STATe] <bool>**
Applicable Models: All

(Read-Write) Turns ON and OFF Time Domain Trace Coupling. All of the measurements in the specified channel are coupled.

- To select Transform parameters to couple, use `CALC:MEAS:TRAN:COUP:PAR`
- To select Gating parameters to couple, use `CALC:MEAS:FILT:COUP:PAR`

Learn more about **Time Domain Trace Coupling**.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<bool>` **ON (or 1)** - Turns ON Time Domain Trace Coupling.
  
  **OFF (or 0)** - Turns OFF Time Domain Trace Coupling.

**Examples**

```
SENS:COUP:PAR 0
sense2:couple:parameter:state on
```

**Query Syntax**

`SENS<cnum>:COUPl:PARameter[:STATe]`?

**Return Type** Boolean

**Default** ON (or 1)
Controls the PXI DC measurement.

SENSe:DC:

| CURRent:RANGe
| SAMPles
| DPOint
| POINts
| TIME
| VOLTage:RANGe

Click on a keyword to view the command details.

SENSe<ch>:DC:CURRent:RANGe <name>, <num>

Applicable Models: M937xA, M9485A, P937xA

(Read-Write) Sets and reads the range (in Amps) used for sensing current, which must be higher than the maximum current you expect to measure. This command is available for PXI SMU only. This is the same function with “IKtM911xMeasurement Interface” section, “SenseCurrentRange” property of the M911x driver.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<name>` String. DC Meter Name.
  
  "SMU*C" or “SMU*V” represents the SMU DC Meter name. “*” is the SMU module number. “C” means current measurement, “V” means voltage measurement.
- `<num>` Range in Amps (3, 0.001 or 0.0001)

Examples

```
SOUR:DC:CURR:RANG "SMU1", 10
```

Query Syntax

```
SENSe<ch>:DC:CURRent:RANGe? <name>
```
SENSe<ch>:DC:SAMPles:DPOint <name>, <num>

**Applicable Models:** M937xA, M9485A, P937xA

*(Read-Write)* Sets and reads the trigger offset in the measurement sweep. A negative value specifies pre-trigger samples, and a positive value specifies post-trigger delay samples. This command is available for PXI SMU only. This is the same function with “IKtM911xMeasurementSweep Interface” section, “OffsetPoints” property of the M911x driver.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<name>` String. DC Meter Name.
  
  "SMU*C" or "SMU*V" represents the SMU DC Meter name. "*" is the SMU module number. “C” means current measurement, “V” means voltage measurement.
- `<num>` Offset points value. Value range: 1 to 100000, step 1,

**Examples**

```
SOUR:DC:SAMP:DPO "SMU1", 10
```

**Query Syntax**

SENSe<ch>:DC:SAMPles:DPOint? <name>

**Return Type** Numeric

**Default** 3

SENSe<ch>:DC:SAMPles:POINTs <name>, <num>
Applicable Models: M937xA, M9485A, P937xA

(Read-Write) Sets and reads the DC measurement number of sample counts per one point measurement. This command is available for both PXI SMU and Digital/Analog I/O M9341B. For PXI SMU, this is the same function with “IKtM911xMeasurementSweep Interface” section, “Points” property of the M911x driver.

Parameters

  <ch>  Any existing channel number. If unspecified, value is set to 1.
  <name>  String. DC Meter Name.

[SMU] "SMU*C” or “SMU*V” represents the SMU DC Meter name. “*” is the SMU module number. “C” means current measurement, “V” means voltage measurement.

[M9341B] "AI1", "AI2", "AI3", "AI4", "AOC1" or" AOC2"

  <num>  Sample count value. Value range: 1 to 100000, step 1 (SMU),

Examples

SOUR:DC:SAMP:POINT "SMU1", 1000

Query Syntax  SENSE<ch>:DC:SAMPLes:POINTs? <name>

Return Type  Numeric

Default  3255 (SMU), 500 (M9341B)

SENSe<ch>:DC:SAMPles:TIME <name>, <num>
"AI1", "AI2", "AI3", "AI4", "AOC1" or "AOC2",

<num> Measurement time in seconds

[SMU] M9111A sample time is 5.12 usec per point.

[SMU] M9111A sample time is 5.12 usec per point.

[M9341B] Only one value can be set for one M9341B module. Its sample time
is 20 nsec per point.

Examples

SOUR:DC:SAMP:TIME "SMU1C", 1000

Query Syntax
SENSe<ch>:DC:SAMPLes:TIME? <name>

Return Type
Numeric

Default 0.0166656 (SMU), 0.00001 (M9341B)

SENSe<ch>:DC:VOLTage:RANGe <name>, <num>

Applicable Models: M937xA, M9485A, E5080A, P937xA

(Read-Write) Sets and reads the range (in Volts) used for sensing voltage, which must be higher than
the maximum current you expect to measure. This command is available for M937xA, M9485A with
Digital/Analog I/O M9341B or E5080A.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<string> String. DC Meter Name.

"AI1", "AI2", "AI3", or "AI4"

<num> Range in Volts (M937xA, M9485A: 1, 5 or 10) (E5080A: 1 or 10)

Examples

SOUR:DC:CURR:RANG "AI1", 10

Query Syntax
SENSe<ch>:DC:VOLTage:RANGe? <name>

Return Type
Numeric

Default 10
Differential I/Q Commands

Controls the Differential I/Q configuration.

**Diff IQ Setup Dialog**

**Range Settings Dialog**

**SCPI**

- `SENSe:DIQ:FREQ:RANGe:ADD`
- `SENSe:DIQ:FREQ:RANGe:COUNt`
- `SENSe:DIQ:FREQ:RANGe:DELeTe`

- `SENSe:DIQ:FREQ:RANGe:STARt`
- `SENSe:DIQ:FREQ:RANGe:STOP`
- `SENSe:DIQ:FREQ:RANGe:IFBW`
- `SENSe:DIQ:FREQ:RANGe:COUPl:STATe`
- `SENSe:DIQ:FREQ:RANGe:COUPl:ID`
- `SENSe:DIQ:FREQ:RANGe:COUPl:OFFSet`
- `SENSe:DIQ:FREQ:RANGe:COUPl:UCONvert`
- `SENSe:DIQ:FREQ:RANGe:COUPl:MULTiplier`
- `SENSe:DIQ:FREQ:RANGe:COUPl:DIvisor`
### Source Configuration Dialog

<table>
<thead>
<tr>
<th>SCPI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:DIQ:PORT:STATe</td>
<td>Power State</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:RANGe</td>
<td>Power Range</td>
</tr>
<tr>
<td>SOURCe:PHASe:EXTernal:CATalog?</td>
<td>Power Catalog?</td>
</tr>
<tr>
<td>SOURCe:PHASe:EXTernal:PORT</td>
<td>Power Port</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:POWer:SWEep</td>
<td>Power Sweep</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:POWer:STARt</td>
<td>Power Start</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:POWer:STOP</td>
<td>Power Stop</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:POWer:ALC:MODE</td>
<td>Power ALC Mode</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:POWer:ATT</td>
<td>Power Attenuation</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:POWer:ATT:AUTO</td>
<td>Power Auto Attenuation</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:PHASe:STATe</td>
<td>Phase State</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:PHASe:SWEep</td>
<td>Phase Sweep</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:PHASe:STARt</td>
<td>Phase Start</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:PHASe:STOP</td>
<td>Phase Stop</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:PHASe:REF</td>
<td>Phase Reference</td>
</tr>
<tr>
<td>SOURCe:PHASe:CONTroll:TOlerance</td>
<td>Phase Control Tolerance</td>
</tr>
<tr>
<td>SOURCe:PHASe:CONTroll:ITERation</td>
<td>Phase Control Iteration</td>
</tr>
</tbody>
</table>

### Match Correction

<table>
<thead>
<tr>
<th>SCPI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:DIQ:PORT:MATCh:STATe</td>
<td>Match State</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:MATCh:TREC</td>
<td>Match Tolerance</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:MATCh:RREC</td>
<td>Match Reference</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:MATCh:RANG</td>
<td>Match Range</td>
</tr>
</tbody>
</table>

### Edit Parameters Dialog

<table>
<thead>
<tr>
<th>SCPI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:DIQ:PARameter:DEFine</td>
<td>Parameter Define</td>
</tr>
<tr>
<td>SENSE:DIQ:PARameter:DELeete</td>
<td>Parameter Delete</td>
</tr>
<tr>
<td>SENSE:DIQ:PARameter:CATalog?</td>
<td>Parameter Catalog?</td>
</tr>
<tr>
<td>SENSE:DIQ:SAVE</td>
<td>Parameter Save</td>
</tr>
<tr>
<td>SENSE:DIQ:LOAD</td>
<td>Parameter Load</td>
</tr>
</tbody>
</table>

Click on a keyword to view the command details.

**See Also**
- **CALC:MEAS:DEFine** - creates a Differential IQ measurement.
- **Select X-Axis commands**
- **Calibration** uses the Guided Calibration commands.

- **Example Program**
- **Learn about DIQ**
- **Synchronizing the Analyzer and Controller**
- **SCPI Command Tree**

### SENSE<ch>:DIQ:FREQuency:RANGe:ADD

**Applicable Models:** N522xB, N524xB

*(Write-only)* Adds a frequency range using the next available range name. For example, with the default F1 range name present, sending this command will create F2. On the Measurement Setup dialog this is the New setting.

**Parameters**

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.

**Examples**

```
SENSe:DIQ:FREQ:RANG:ADD
sense2:diq:frequency:range:add
```

**Query Syntax**

Not applicable

**Default** Not applicable

### SENSE<ch>:DIQ:FREQuency:RANGe:COUNt?

**Applicable Models:** N522xB, N524xB

*(Read-only)* Returns the number of frequency ranges in the DIQ channel.

**Parameters**

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.

**Examples**

```
SENSe:DIQ:FREQ:RANG:COUN?
```

**Return Type** Numeric

**Default** 1
**SENSe<ch>:DIQ:FREQuency:RANGe<rNum>:COUPle:DIVisor <value>**

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and reads the value by which the coupled range will be divided to achieve the frequency range specified by `<rNum>`.

On the **Frequency Range** dialog under **Coupling**, this is the **Divisor** setting.

**Parameters**

- `<ch>`: The Differential IQ channel number. If unspecified, value is set to 1.
- `<rNum>`: Frequency range number.
- `<value>`: Divisor value. Choose a positive or negative integer.

**Examples**

```
SENSe:DIQ:FREQ:RANGe:COUPle:DIV 1
sense2:diq:frequency:range2:couple:divisor 2
```

**Query Syntax**

```
SENSe<ch>:DIQ:FREQuency:RANGE<rNum>:COUPle:DIVisor?
```

**Return Type**

Integer

**Default**

1

**SENSe<ch>:DIQ:FREQuency:RANGe<rNum>:COUPle:ID <value>**

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and reads the frequency range to couple settings to.

On the **Frequency Range** dialog under **Coupling**, this is the **range to Couple To** setting.

**Parameters**

- `<ch>`: The Differential IQ channel number. If unspecified, value is set to 1.
- `<rNum>`: Frequency range number to be coupled. Range 1 can NOT be coupled to another range.
- `<value>`: Frequency range number to couple to. This range should be already created using SENS:DIQ:FREQ:RANGE:ADD

**Examples**

```
SENSe:DIQ:FREQ:RANG2:COUPle:ID 3
sense2:diq:frequency:range2:couple:id 1
```

**Query Syntax**

```
SENSe<ch>:DIQ:FREQuency:RANGE<num>:COUPle:ID?
```

**Return Type**

Integer
SENSe<ch>:DIQ:FREQuency:RANGe<rNum>:COUPle:MULTiplier <value>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the value by which the coupled range will be multiplied to achieve the frequency range specified by <rNum>.

On the Frequency Range dialog under Coupling, this is the Multiplier setting.

Parameters
- <ch> The Differential IQ channel number. If unspecified, value is set to 1.
- <rNum> Frequency range number.
- <value> Multiplier value. Choose a positive or negative integer.

Examples
- SENS:DIQ:FREQ:RANG1:COUP:MULT 2
- sense2:diq:frequency:range2:couple:multiplier 1

Query Syntax
SENSe<ch>:DIQ:FREQuency:RANGe<num>:COUPle:MULTiplier?

Return Type
- Integer
- Default 1

SENSe<ch>:DIQ:FREQuency:RANGe<rNum>:COUPle:OFFSet <value>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the frequency range number to be used as an offset. The frequencies of the range <rNum> will be offset from the 'coupled to' range by this frequency range. The SENS:DIQ:FREQ:RANG:COUP:UCON command determines whether the offset is positive or negative.

On the Frequency Range dialog under Coupling, this is the Offset setting.

Parameters
- <ch> The Differential IQ channel number. If unspecified, value is set to 1.
- <rNum> Frequency range number.
- <value> Offset range number. The resulting range must be within the frequency range of the analyzer.
<table>
<thead>
<tr>
<th>Examples</th>
<th>SENS:DIQ:FREQ:RANG1:COUP:OFFS 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sense2:diq:frequency:range3:couple:offset 2 'range 3 is offset from the 'coupled to' range by the frequencies defined by range 2</td>
</tr>
</tbody>
</table>

**Query Syntax**  
SENS<ch>:DIQ:FREQuency:RANGe<num>:COUPle:OFFSet?

**Return Type**  
Numeric

**Default**  
1

**SENSe<ch>:DIQ:FREQuency:RANGe<num>:COUPle:STATe <value>**

**Applicable Models:** N522xB, N524xB

(Read-Write) Sets and reads the ON / OFF state of frequency range coupling.

On the Frequency Range dialog under **Coupling**, this is the **Couple (On|Off)** setting.

**Parameters**

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.
- `<rNum>` Frequency range number.
- `<value>` (Boolean) Choose from:
  - ON or 1 - Range is coupled
  - OFF or 0 - Range is NOT coupled

<table>
<thead>
<tr>
<th>Examples</th>
<th>SENS:DIQ:FREQ:RANG1:COUP:STAT 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sense2:diq:frequency:range2:couple:state off</td>
</tr>
</tbody>
</table>

**Query Syntax**  
SENS<ch>:DIQ:FREQuency:RANGe<num>:COUPle:STATe?

**Return Type**  
Boolean

**Default**  
OFF (0)

**SENSe<ch>:DIQ:FREQuency:RANGe<num>:COUPle:UCONvert <value>**
**Applicable Models:** N522xB, N524xB

(Read-Write) Sets and reads the state of the Up / Down conversion setting.

On the **Frequency Range** dialog under **Coupling**, this is the Up (On|Off) setting.

**Parameters**

- `<ch>`: The Differential IQ channel number. If unspecified, value is set to 1.
- `<rNum>`: Frequency range number.
- `<value>`: (Boolean) Choose from:
  - ON or 1 - The offset range is ADDED to the 'coupled to' frequency range.
  - OFF or 0 - The offset range is SUBTRACTED from to the 'coupled to' frequency range.

**Examples**

```
SENS:DIQ:FREQ:RANG1:COUP:UCONvert 1
sense2:diq:frequency:range2:couple:uconvert off
```

**Query Syntax**

```
SENSe<ch>:DIQ:FREQuency:RANGe<num>:COUPle:UCONvert?
```

**Return Type**

Boolean

**Default**

OFF

---

**SENSe<ch>:DIQ:FREQuency:RANGe<rNum>:DELETE**

**Applicable Models:** N522xB, N524xB

(Write-only) Deletes the specified frequency range.

On the **Measurement Setup dialog** this is the **Remove** setting.

**Parameters**

- `<ch>`: The Differential IQ channel number. If unspecified, value is set to 1.
- `<rNum>`: Frequency range number to delete.

**Examples**

```
SENS:DIQ:FREQ:RANG1:DEL
sense2:diq:frequency:range2:delete
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable
SENSe<ch>:DIQ:FREQuency:RANGe<rNum>:IFBW <value>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the receiver IF bandwidth setting.

On the Frequency Range dialog under Frequency, this is the IFBW setting.

Parameters

<ch>  The Differential IQ channel number. If unspecified, value is set to 1.
<rNum> (Integer) Frequency range number.
<value> (Numeric) IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the analyzer model. (See the list.) If an invalid number is specified, the analyzer will round up to the closest valid number.

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:DIQ:FREQ:RANG1:IFBW 1e3</td>
</tr>
<tr>
<td>sense2:diq:frequency:range2:ifbw 100e3</td>
</tr>
</tbody>
</table>

Query Syntax  SENSe<ch>:DIQ:FREQuency:RANGe<num>:IFBW?

Return Type  Numeric

Default  1e5

SENSe<ch>:DIQ:FREQuency:RANGe<rNum>:STARt <value>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the start value for the specified frequency range.

On the Frequency Range dialog under Frequency, this is the Start setting.

Parameters

<ch>  The Differential IQ channel number. If unspecified, value is set to 1.
<rNum> (Integer) Frequency range number.
<value> (Numeric) Frequency range start value. Choose a value within the frequency range of the analyzer.

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:DIQ:FREQ:RANG1:STAR 1e9</td>
</tr>
<tr>
<td>sense2:diq:frequency:range2:start 20e6</td>
</tr>
</tbody>
</table>

Query Syntax  SENSe<ch>:DIQ:FREQuency:RANGe<num>:STARt?

Return Type  Numeric

Default  Start frequency of the analyzer
SENSe<ch>:DIQ:FREQuency:RANGe<rNum>:STOP <value>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the stop value for the specified frequency range.

On the Frequency Range dialog under Frequency, this is the Stop setting.

**Parameters**

- `<ch>`: The Differential IQ channel number. If unspecified, value is set to 1.
- `<rNum>`: (Integer) Frequency range number.
- `<value>`: (Numeric) Frequency range stop value. Choose a value within the frequency range of the analyzer.

**Examples**

- `SENS:DIQ:FREQ:RANG1:STOP 10e9`
- `sense2:diq:frequency:range2:stop 500e6`

**Query Syntax**

`SENSe<ch>:DIQ:FREQuency:RANGe<num>:STOP?`

**Return Type**

Numeric

**Default**

Maximum frequency of the analyzer.

---

SENSe<ch>:DIQ:LOAD <filename>[,<type>]

Applicable Models: N522xB, N524xB

(Write-only) Recalls the list of parameters and frequency ranges settings from a previously-saved *.xml file.

Use `SENSe:DIQ:SAVE` to store the files.

**Note:** The Frequency Range settings and the DIQ Parameters are saved and recalled from a single *.xml file.

**Parameters**

- `<ch>`: The Differential IQ channel number. If unspecified, value is set to 1.
- `<filename>`: (String) Full path (optional) and filename with or without the *.xml extension. If the full path is not provided, the file is saved to the D:\drive.
- `<type>`: (Character) Choose the type of settings to be recalled:

  **PLIS** - just the parameters.
FRANge - just the frequency settings.

ALL - both parameters and frequency settings.

Examples

```
SENS:DIQ:LOAD "myDIQfile",ALL
sense2:diq:load "D:\myDIQfile.xml",plist
```

Query Syntax Not Applicable

Default PLISt

SENSe<ch>:DIQ:SAVE <filename>

Applicable Models: N522xB, N524xB

(Write-only) Stores the list of parameters and frequency ranges settings to an*.xml file for recall at a later time.

Use SENSe:DIQ:LOAD to recall files.

Note: The Frequency Range settings and the DIQ Parameters are saved and recalled from a single *.xml file.

Parameters

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.
- `<filename>` (String) Full path (optional) and filename with or without the *.xml extension. If the full path is not provided, the file is saved to the D:\ drive.

Examples

```
SENS:DIQ:SAVE "myDIQfile"
sense2:diq:save "D:\myDIQfile.xml"
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:DIQ:PARameter:CATalog?
**Applicable Models:** N522xB, N524xB

(Read-only) Returns a list of all existing parameters.

**Parameters**

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.
- `<name>` (String) Comma-separated list of parameters, each parameter in the form “name:expression”.

**Examples**

```text
SENS:DIQ:PAR:CAT?
sense2:diq:parameter:catalog?
'Example of returned parameters
"IPwrF1:a1_F1","OPwrF1:b2_F1","GainF1:b2_F1/a1_F1"
```

**Default** Not Applicable

---

**SENSe<ch>:DIQ:PARameter:DEFine <name>,<expression>**

**Applicable Models:** N522xB, N524xB

(Write-only) Create a new parameter for Differential IQ channel. Use `CALCulate:MEASure:DEFine` to create a new trace with the new parameter.

Use `CALCulate:MEASure:PARameter` to change the parameter of one of the existing traces to the new parameter.

**Parameters**

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.
- `<name>` (String) Parameter name.
  
  **Note:** Do not use underscores in the parameter name. For example, b2_f1 cannot be used as a parameter name. However, b2f1 is a valid parameter name.

- `<expression>` (String) Parameter expression using receiver names and mathematical expressions.

**Examples**

```text
SENS:DIQ:PAR:DEF "myNewParam","(a1_F1+b1_F2)/c1"
sense2:diq:parameter:define "myNewParam","(a1_F1+b1_F2)/c1"
```

**Query Syntax** Not Applicable

**Default** Not Applicable
SENSe<ch>:DIQ:PARAMeter:DELete <name>

**Applicable Models:** N522xB, N524xB

*(Write-only)* Deletes the named parameter.

**Parameters**

- `<ch>`: The Differential IQ channel number. If unspecified, value is set to 1.
- `<name>`: *(String)* Parameter name that was used when the parameter was created.

**Examples**

```
SENSe:DIQ:PAR:DEL "myNewParam"
sense2:diq:parameter:delete "myNewParam"
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

SENSe<ch>:DIQ:PORT<port>:MATCH:RANGe <value>[,<src>]

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and reads the existing frequency ranges over which Match Correction is to be performed.

On the *Source Configuration* dialog under *Match Correction*, this is the *Match Frequency Range* setting.

**Parameters**

- `<ch>`: The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>`: *(Integer)* Source port number.
- `<value>`: *(String)* Frequency ranges, including the "F<n>" , where <n> is the range number. Separate each range with a comma.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
SENSe:DIQ:PORT1:MATC:RANG "F1,F2,F3"
sense2:diq:port3:match:range "F1", "Port 1 Src2"
```
### Query Syntax

```
SENSe<ch>:DIQ:PORT:RANGe? [src]
```

**Return Type**  
String

**Default**  
Depends on `<port>`

---

**SENSe<ch>:DIQ:PORT<port>:MATCH:RRECeiver <value>[,<src>]**

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and reads the reference receiver to be used to perform Match Correction.

On the **Source Configuration** dialog under **Match Correction**, this is the **Reference Receiver** setting.

**Parameters**

- `<ch>`  
The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>`  
(Integer) Source port number.
- `<value>`  
(String) Choose any of the reference receivers in the analyzer using logical receiver notation. [Learn more.](#)

These would be "a_" where _ is the test port number.

- `<src>`  

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

**Examples**

- `SENSe:DIQ:PORT2:MATC:RREC "a2" 'port 2 reference receiver`
- `SENSe:DIQ:PORT3:MATC:RREC "a1","Port 1 Src2"`

---

**SENSe<ch>:DIQ:PORT<port>:MATCH:STATe <value>[,<src>]**

**Query Syntax**  
```
SENSe<ch>:DIQ:PORT:RRECeiver? [src]
```

**Return Type**  
String

**Default**  
Depends on `<port>`

---

4242
Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the Match Correction ON/OFF state.

On the Source Configuration dialog under Match Correction, this is the Match Correction ON setting.

Parameters

- `<ch>`  The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>` (Integer) Source port number.
- `<value>` (Boolean) Choose from:
  - ON or 1 - Perform Match Correction
  - OFF or 0 - Do not perform Match Correction.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

Examples

```
SENS:DIQ:PORT1:MATC:STAT 1
sense2:diq:port3:match:state off,"Port 1 Src2"
```

Query Syntax

SENSe<ch>:DIQ:PORT:STATe? [src]

Return Type

Boolean

Default

OFF

SENSe<ch>:DIQ:PORT<port>:MATCh:TRCeiver <value>,<src>]
Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the test port receiver to be used to perform Match Correction.

On the Source Configuration dialog under Match Correction, this is the Test Receiver setting.

**Parameters**

- `<ch>`: The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>`: (Integer) Source port number.
- `<value>`: (String) Choose any of the test port receivers in the analyzer using logical receiver notation. Learn more.

These would be "b_" where _ is the test port number.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```plaintext
SENS:DIQ:PORT3:MATC:TREC "b3"

sense2:diq:port3:match:treceiver "b1","Port 1 Src2"
```

**Query Syntax**

SENSe<ch>:DIQ:PORT:TRECiever? [src]

**Return Type**

String

**Default**

Depends on `<port>`

SENSe<ch>:DIQ:PORT<port>:PHASE:PARameter <value>[,<src>]
Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the receivers to be used to measure the phase of the sources. The phase measurement will be the difference between these two receivers.

Select the receivers based on your application. You are responsible to make sure that your DUT configuration routes the signals of interest to the correct receivers. Otherwise, the phase will not be properly controlled.

On the Source Configuration dialog under Phase, this is the Control Receiver setting.

Parameters

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>` (Integer) Source port number.
- `<value>` (String) Phase parameter using the following syntax:

  \[ r\text{Cont}/r\text{Ref} \]

  Where:

  - `r\text{Cont}` is the receiver used to measure the controlled source.
  - `r\text{Ref}` is the receiver used to measure the reference source.

Only logical receiver notation (a1,b1) is available. Learn more.

Enclose the entire parameter in quotes.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:DIQ:PORT1:PHAS:PAR &quot;a1/a3&quot;</td>
<td>String</td>
<td>Not applicable</td>
</tr>
<tr>
<td>sense2:diq:port3:phase:parameter &quot;b2/a2&quot;,&quot;Port 1 Src2&quot;</td>
<td>String</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

SENSe<ch>:DIQ:PORT<port>:PHASe:REFerence <value>[,<src>]

Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the port to be used as a reference when controlling phase for the specified <port>.

On the Source Configuration dialog under Phase, this is the Refer To setting.

Parameters

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>` (Integer) Source port number.
- `<value>` (String) Reference port. Use SOUR:CAT? to return a list of valid port names.

The two internal VNA sources are available ONLY at specific ports. For example on a 4-port PNA-X, the possible port pairings are: 1/3, 1/4, 2/3, or 2/4. Port 1 can NOT be paired with Port 2, and Port 3 can NOT be paired with Port 4. Learn more about these limitations.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:DIQ:PORT1:PHAS:REF &quot;port 2&quot;</td>
<td>String</td>
<td>Depends on &lt;port&gt;</td>
</tr>
<tr>
<td>sense2:diq:port3:phase:reference &quot;Port 3&quot;,&quot;Port 1 Src2&quot;</td>
<td>String</td>
<td>Depends on &lt;port&gt;</td>
</tr>
</tbody>
</table>

Query Syntax

SENSe<ch>:DIQ:PORT<port>:PHASe:REFeReference? [src]

Return Type

String

Default

Depends on <port>
**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and reads the start value for a phase sweep.

On the *Source Configuration* dialog under *Phase*, this is the *Start Phase* setting.

**Parameters**

- `<ch>`: The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>`: (Integer) Source port number.
- `<value>`: (Numeric) Start phase sweep value in degrees. Choose any positive or negative value.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

**Examples**

```
SENS:DIQ:PORT1:PHAS:STAR -90
sense2:diq:port3:phase:start 0,"Port 1 Src2"
```

**Query Syntax**

`SENSe<ch>:DIQ:PORT<port>:PHASe:STARt? [src]`

**Return Type**

Numeric

**Default**

0

`SENSe<ch>:DIQ:PORT<port>:PHASe:STATe <value>[,<src>]`
**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and reads the ON/ OFF state of phase control.

On the **Source Configuration** dialog under **Phase**, this is the **Phase State** setting.

### Parameters

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>` (Integer) Source port number.
- `<value>` (Character) Phase control state. Choose from:
  - **OFF** - Phase is NOT set or controlled.
  - **CONTrolled** - Phase is measured and iterated to within the specified tolerance. Specify the receivers and iteration properties using the **Source:Phase** commands.
  - **OPENloop** - Phase is set, but receivers are NOT used to measure and iterate the phase of the source. Therefore, the setting of phase is not as accurate or stable. Open Loop mode can be used with phase sweep (for example, from 0 to 360 degrees). However, each sweep may not start at 0 degrees. NO settings on the Phase Control Setup dialog are used in Open Loop.

Note: After selecting Open Loop, set each source to ON (not Auto).


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

### Examples

```plaintext
SENS:DIQ:PORT1:PHAS:STAT OFF
sense2:diq:port3:phase:state controlled,"Port 1 Src2"
```

### Query Syntax

```
SENSe<ch>:DIQ:PORT<port>:PHASe:STATe? [src]
```

### Return Type

- **Character**

### Default

- **OFF**

```
SENSe<ch>:DIQ:PORT<port>:PHASe:STOP <value>[,<src>]
```
Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the stop value for a phase sweep.

On the Source Configuration dialog under Phase, this is the Stop Phase setting.

**Parameters**

- `<ch>`: The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>`: (Integer) Source port number.
- `<value>`: (Numeric) Stop phase sweep value in degrees. Choose any positive or negative value.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

- `SENS:DIQ:PORT1:PHASE:STOP 270`
- `sense2:diq:port3:phase:stop "Port 1 Src2"

**Query Syntax**

`SENSe<ch>:DIQ:PORT<port>:PHASE:STOP? [src]`

**Return Type**

Numeric

**Default**

0

`SENSe<ch>:DIQ:PORT<port>:PHASE:SWEep[:STATe] <value>[,<src>]`
**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and reads the ON / OFF state of phase sweep.

On the **Source Configuration** dialog under **Phase**, this is the **Sweep Phase (On|Off)** setting.

**Parameters**

- **<ch>** The Differential IQ channel number. If unspecified, value is set to 1.
- **<port>** (Integer) Source port number.
- **<value>** (Boolean) Choose from:
  - ON or 1 - Sweep phase.
  - OFF or 0 - Do not sweep phase.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

**Examples**

```
SENS:DIQ:PORT1:PHAS:SWE 1
sense2:diq:port3:phase:sweep:state off,"Port 1 Src2"
```

**Query Syntax**

```
SENSe<ch>:DIQ:PORT<port>:PHASe:SWEep[:STATe]? [src]
```

**Return Type** Boolean

**Default** OFF

---

**SENSe<ch>:DIQ:PORT<port>:POWer:ALC:MODE <value>[,<src>]**

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and reads the leveling mode to be used for the specified source port.

On the **Source Configuration** dialog under **Power**, this is the **Leveling Mode** setting.

**Parameters**

- **<ch>** The Differential IQ channel number. If unspecified, value is set to 1.
- **<port>** (Integer) Source port number.
<value> (String) Leveling mode. Choose from:

"Internal" - Standard internal analyzer leveling mode.

"Internal-<n>,<p>" - Receiver Leveling, where:

<n> is the receiver name.
<p> is the source port to be leveled.

"Open Loop" - Open loop leveling, used during pulse conditions with the internal source modulators. NOT available on all models. No leveling is used in setting the source power. The lowest settable power, without attenuation, is limited to -30dBm. The source power level accuracy is very compromised. Use a source power calibration to make the source power somewhat more accurate.

"Open Loop-<n>,<p>" - Open loop leveling, where:

<n> is the receiver name.
<p> is the source port to be leveled


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

SENS:DIQ:PORT1:POW:ALC:MODE "Internal"
sense2:diq:port3:power:alc:mode "Internal-a2,1","Port 1 Src2"

Query Syntax

SENSe<ch>:DIQ:PORT<port>:POWer:ALC:MODE? [src]

Return Type

String

Default

"Internal"

SENSe<ch>:DIQ:PORT<port>:POWer:ATTenuation <value>[,.<src>]
Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the amount of source attenuation. Sending this command will set SENS:DIQ:PORT:POW:ATT:AUTO OFF

On the Source Configuration dialog under Power, this is the Source Attenuation setting.

Parameters

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>` (Integer) Source port number.
- `<value>` (Numeric) Source Attenuation value. Choose from 0 to the maximum amount of source attenuation in the correct step size.

Rounding will occur when the selected value can not be achieved.

See attenuators for all VNA models.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SENS:DIQ:PORT1:POW:ATT 10
sense2:diq:port3:power:attenuation 0,"Port 1 Src2"
```

Query Syntax

`SENSe<ch>:DIQ:PORT<port>:POWer:ATTenuation? [src]`

Return Type

Numeric

Default

0

`SENSe<ch>:DIQ:PORT<port>:POWer:ATTenuation:AUTO <value>[,<src>]`
Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the ON/ OFF state of auto range source attenuation.

On the Source Configuration dialog under Power, this is the Auto range source attenuator setting.

Parameters

- **<ch>** The Differential IQ channel number. If unspecified, value is set to 1.
- **<port>** (Integer) Source port number.
- **<value>** (Boolean) Choose from:
  - ON or 1 - Auto range the source attenuation.
  - OFF or 0 - Do not Auto range the source attenuation.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SENSe:DIQ:PORT1:POW:ATT:AUTO 1
sense2:diq:port3:power:attenuation:auto off,"Port 1 Src2"
```

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
</table>

SENSe<ch>:DIQ:PORT<port>:POWer:STARt <value>[,<src>]
Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the start value of a power sweep.

On the **Source Configuration** dialog under **Power**, this is the **Start Power** setting.

**Parameters**

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>` (Integer) Source port number.
- `<value>` (Numeric) Power sweep start value in dBm. Choose start and stop value within a single attenuator range of the analyzer (generally about 30 dB).

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

**Examples**

```
SENS:DIQ:PORT1:POW:STAR -10
sense2:diq:port3:power:start 0,"Port 1 Src2"
```

**Query Syntax**

```
SENS<ch>:DIQ:PORT<port>:POWer:STARt? [src]
```

**Return Type** Numeric

**Default** -5 dBm

```
SENS<ch>:DIQ:PORT<port>:POWer:STOP <value>[,<src>]
```
(Read-Write) Sets and reads the stop value of a power sweep.

On the Source Configuration dialog under Power, this is the Stop Power setting.

**Parameters**

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>` (Integer) Source port number.
- `<value>` (Numeric) Power sweep stop value in dBm. Choose start and stop value within a single attenuator range of the analyzer (generally about 30 dB).

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

**Examples**

```
SENS:DIQ:PORT1:POW:STOP 0
sense2:diq:port3:power:stop 2,"Port 1 Src2"
```

**Query Syntax**

SENSe<ch>:DIQ:PORT<port>:POWer:STOP? [src]

**Return Type** Numeric

**Default** -5 dBm

---

SENSe<ch>:DIQ:PORT<port>:POWer:SWEep[:STATe] <value>[,<src>]
Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the state of sweeping power.

On the Source Configuration dialog under Power, this is the Sweep Power (On|Off) setting.

Parameters

- `<ch>` The Differential IQ channel number. If unspecified, value is set to 1.
- `<port>` (Integer) Source port number.
- `<value>` (Boolean) Sweep Power state. Choose from:
  - ON or 1 - Perform power sweep.
  - OFF or 0 - Do not perform power sweep.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

Examples

```
SENS:DIQ:PORT1:POW:SWE 1
sense2:diq:port3:power:sweep:state off, "Port 1 Src2"
```

Query Syntax

SENSe<ch>:DIQ:PORT<port>:POWer:SWEep[:STATe]? [src]

Return Type

Boolean

Default

OFF

SENSe<ch>:DIQ:PORT<port>:RANGe <value>[,<src>]
Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the frequency range number to which the specified source port will be set.

At the top of the Source Configuration dialog this is the Frequency Range setting.

Parameters

<ch> The Differential IQ channel number. If unspecified, value is set to 1.
<port> (Integer) Source port number.
<value> (Integer) Frequency range number.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

SENS:DIQ:PORT1:RANG 1
sense2:diq:port3:range 2,"Port 1 Src2"

Query Syntax

SENSe<ch>:DIQ:PORT:RANGe? [src]

Return Type

Integer

Default

1

SENSe<ch>:DIQ:PORT<port>:STATe <value>[,<src>]
Applicable Models: N522xB, N524xB

(Read-Write) Sets and reads the ON / OFF state of the source specified by <port>.

At the top of the Source Configuration dialog this is the Source State setting.

Parameters

- <ch> The Differential IQ channel number. If unspecified, value is set to 1.
- <port> (Integer) Source port name.
- <value> (Character) Choose from:

  AUTO - Source power is turned ON at the specified test port when required by the measurement. This is the most common (default) setting. Auto sources are turned OFF when other sources are performing Match Correction sweeps.

  ON - Source power is ALWAYS ON, regardless of measurements that are in process. Use this setting to supply source power to a DUT port that always requires power, such as an LO port. This could turn OFF power at another test port. Learn about internal second source restrictions.

  OFF - Source power is never ON, regardless of the measurement requirements. Use this setting to prevent damage to a sensitive DUT test port.


  While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SENS:DIQ:PORT1:STAT Auto
sense2:diq:port3:state On, "Port 1 Src2"
```

Query Syntax

```
SENSe<ch>:DIQ:PORT:STATe? [src]
```

Return Type

Character

Default

Auto
Modulation Distortion Measurement

Defines the modulation distortion settings including measurement, RF path, modulation source name, and sweep frequency and power.

<table>
<thead>
<tr>
<th>SENSE:</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISTortion</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>NF</td>
</tr>
<tr>
<td>OUTPut</td>
</tr>
<tr>
<td>SOURce</td>
</tr>
<tr>
<td>ATTenuation</td>
</tr>
<tr>
<td>INCLude</td>
</tr>
<tr>
<td>NOMinal</td>
</tr>
<tr>
<td>AMPLifier</td>
</tr>
<tr>
<td>PULSe</td>
</tr>
<tr>
<td>RECeiver</td>
</tr>
<tr>
<td>AUTO</td>
</tr>
<tr>
<td>SWEep</td>
</tr>
<tr>
<td>CARRier</td>
</tr>
<tr>
<td>FREQuency</td>
</tr>
<tr>
<td>DWELl</td>
</tr>
<tr>
<td>POWer</td>
</tr>
<tr>
<td>CARRier</td>
</tr>
<tr>
<td>LEVel</td>
</tr>
<tr>
<td>PORT</td>
</tr>
<tr>
<td>TYPE</td>
</tr>
<tr>
<td>LIST</td>
</tr>
<tr>
<td>ADD</td>
</tr>
<tr>
<td>DELe te</td>
</tr>
<tr>
<td>LEVel</td>
</tr>
<tr>
<td>LOAD</td>
</tr>
<tr>
<td>NBW</td>
</tr>
<tr>
<td>MODE</td>
</tr>
<tr>
<td>POINts</td>
</tr>
<tr>
<td>RECeiver</td>
</tr>
</tbody>
</table>
| ATTenuation
| MODE
| SAVE
| SOURce
| ATTenuation
| MODE
| RAMP
| LEVEL
| STARt
| STOP
| NBW
| AUTO
| PIONts
| SPARam
| LEVEL
| RETRace
| POWER
| SPARam
| BWIDTH
| REUSE
| [:STATE]
| STEP
| TYPE
| TYPE
| TABLe
| CATalog?
| DATA
Click on a keyword to view the command details.

**See Also**

- Learn about Modulation Distortion
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

```sql
SENSe<cnum>:DISTortion:ADC:FILTER:TYPE <enum>
```
Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Selects between a narrow and wide IF filter anti-aliasing path.

Parameters

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<enum>` Choose from:

  - **NARRow** - Selects the ADC 11 MHz IF filter path.
  - **WIDE** - Selects the ADC 38 MHz IF filter path.
  - **AUTO** - Automatically set the ADC Filter setting based on the ADC Sampling Frequency.

Examples

```
SENS:DIST:ADC:FILT:TYPE WIDE
```

Query Syntax

```
SENSe<cnum>:DISTortion:ADC:FILTer:TYPE?
```

Return Type

Enumeration

Default

AUTO

---

Applicable Models: All with Option S93070xB or S95070A/B

(Write-only) This command launches the Receiver IF Cal. The optional argument allows the user to run the command with asynchronous behavior.

Parameters

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `[SYNChronous | ASYNchronous]` Optional argument. Choose from:

  - **SYNChronous** - Blocks SCPI commands during standard measurement (default behavior).
  - **ASYNchronous** - Does NOT block SCPI commands during standard measurement.

Examples

```
SENS:DIST:CORR:COLL:IF:ACQuire [SYNChronous | ASYNchronous]
```

Learn more about this argument

Return Type

Not applicable
SENS<sup>cnum</sup>:DISTortion:EVM:NORMalize <num>

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-Write)* Sets and returns the scaling factor applied to the EVM measurements. Enter a value between 0.1 and 1.0.

**Parameters**
- `<cnum>`: Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>`: Scaling factor number between 0.1 and 1.0.

**Examples**
- `SENS:DIST:EVM:NORM 1`

**Query Syntax**
- `SENS<cnum>:DISTortion:EVM:NORMalize?`

**Return Type**
- Numeric

**Default**
- 1.0

--------

SENS<sup>cnum</sup>:DISTortion:FREQuency:TUNE:IMMediate

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Write-only)* Auto tunes all measurement bands for the modulation distortion channel.

**Parameters**
- `<cnum>`: Any existing Modulation Distortion channel. If unspecified, value is set to 1.

**Examples**
- `SENS:DIST:FREQ:TUNE:IMM`

**Return Type**
- Not applicable

**Default**
- Not applicable

--------

SENS<sup>cnum</sup>:DISTortion:MEASure:BAND<bnum>:ACP:LOWer:IBW <num>
Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the integration bandwidth (IBW) of the lower ACP measurement (ACPLo). The IBW is used to determine total signal power within a specified frequency span. For example, to calculate the total power of a signal composed of 100 tones spaced 1 MHz apart over a 100 MHz span, the signal power would be integrated over an IBW of 100 MHz.

![ACPLo Signal ACPUp](image)

**Parameters**

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.
- `<num>` Lower ACP integration bandwidth.

**Examples**

```
SENS:DIST:MEAS:BAND:ACP:LOW:IBW 100 MHz
```

**Query Syntax**

```
SENSe<cnum>:DISTortion:MEASure:BAND<bnum>:ACP:LOWer:IBW?
```

**Return Type** Numeric

**Default** 100 MHz

```
SENSe<cnum>:DISTortion:MEASure:BAND<bnum>:ACP:LOWer:OFFSet <num>
```
**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-Write)* Sets and returns the lower ACP measurement offset (ACPLo). Offsets the lower ACP integration bandwidth relative to the LO used to generate the modulated signal.

![Diagram of ACPLo Signal ACPUp with 0 Hz SA Display](image)

**Parameters**
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.
- `<num>` Lower ACP offset.

**Examples**
```
SENS:DIST:MEAS:BAND:ACP:LOW:OFFS -100 MHz
```

**Query Syntax**
```
SENS<cnum>:DISTortion:MEASure:BAND<bnum>:ACP:LOWer:OFFSet?
```

**Return Type** Numeric

**Default** -100 MHz

```
SENS<cnum>:DISTortion:MEASure:BAND<bnum>:ACP:UPPer:IBW <num>
```
Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the integration bandwidth (IBW) of the upper ACP measurement (ACPUp). The IBW is used to determine total signal power within a specified frequency span. For example, to calculate the total power of a signal composed of 100 tones spaced 1 MHz apart over a 100 MHz span, the signal power would be integrated over an IBW of 100 MHz.

Parameters
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.
- `<num>` Upper ACP integration bandwidth.

Examples
```
SENS:DIST:MEAS:BAND:ACP:UPP:IBW 100 MHz
```

Query Syntax
```
SENSe<cnum>:DISTortion:MEASure:BAND<bnum>:ACP:UPPer:IBW?
```

Return Type
Numeric

Default
100 MHz

```
SENSe<cnum>:DISTortion:MEASure:BAND<bnum>:ACP:UPPer:OFFSet <num>
```
**Applicable Models:** All with Option S93070xB or S95070A/B

**(Read-Write)** Sets and returns the upper ACP measurement offset (ACPUp). Offsets the upper ACP integration bandwidth relative to the LO used to generate the modulated signal.

![ACPLo Signal ACPUp](image)

**Parameters**
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.
- `<num>` Upper ACP offset.

**Examples**

```
SENS:DIST:MEAS:BAND:ACP:UPP:OFFS 100 MHz
```

**Query Syntax**

```
SENSe<cnum>:DISTortion:MEASure:BAND<bnum>:ACP:UPPer:OFFSet?
```

**Return Type** Numeric

**Default** 100 MHz

---

**SENSe<cnum>:DISTortion:MEASure:BAND<bnum>:ADD**

**Applicable Models:** All with Option S93070xB or S95070A/B

**(Write-only)** Adds a new distortion measurement band.

**Parameters**
- `<cnum>` Any existing modulation distortion channel. If unspecified, value is set to 1
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.

**Examples**

```
SENS:DIST:MEAS:BAND:ADD
```

**Query Syntax**

Not applicable

**Default** Not Applicable
SENSe<cnun>:DISTortion:MEASure:BAND<bnum>:AUTofill

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Write-only)* Automatically fills in the measurement settings for all bands from the currently active modulation file loaded in the source. Learn more.

**Parameters**
- `<cnun>` Any existing modulation distortion channel. If unspecified, value is set to 1.
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.

**Examples**
- SENSe:DIST:MEAS:BAND:AUT

**Query Syntax**
- Not applicable

**Default**
- Not Applicable

SENSe<cnun>:DISTortion:MEASure:BAND<bnum>:CARRier:IBW <num>

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-Write)* Sets and returns the carrier (Signal) integration bandwidth (IBW) for the distortion measurement. The IBW is used to determine total signal power within a specified frequency span. For example, to calculate the total power of a signal composed of 100 tones spaced 1 MHz apart over a 100 MHz span, the signal power would be integrated over an IBW of 100 MHz.

![ACPLo Signal ACPIp

- Hz

SA Display

**Parameters**
- `<cnun>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.
- `<num>` Carrier integration bandwidth.

**Examples**
- SENSe:DIST:MEAS:BAND:CARR:IBW 100 MHz

**Query Syntax**
- SENSe<cnun>:DISTortion:MEASure:BAND<bnum>:CARRier:IBW?
SENSe<cnum>:DISTortion:MEASure:_BAND<bnum>:CARRier:OFFSet <num>

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-Write)* Sets and returns the carrier offset (Signal). Offsets the Carrier integration bandwidth relative to the Carrier LO used to generate the modulation signal.

**Parameters**
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.
- `<num>` Carrier offset.

**Examples**
- `SENS:DIST:MEAS:BAND:CARR:OFFS 0`

**Query Syntax**
- `SENSe<cnum>:DISTortion:MEASure:_BAND<bnum>:CARRier:OFFSet?`

**Return Type** Numeric
- **Default** 0

SENSe<cnum>:DISTortion:MEASure:_BAND<bnum>:COUNt?
Applicable Models: All with Option S93070xB or S95070A/B

(Read-only) Queries the total number of distortion measurement bands.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.

Examples

```
SENS:DIST:MEAS:BAND:COUN?
```

Return Type Numeric

Default Not Applicable

---

SENS<cnm>:DISTortion:MEASure:BAND<bnum>:DELete

Applicable Models: All with Option S93070xB or S95070A/B

(Write-only) Deletes the specified distortion measurement band.

Parameters

- `<cnum>` Any existing modulation distortion channel. If unspecified, value is set to 1
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.

Examples

```
SENS:DIST:MEAS:BAND:DEL 2
```

Query Syntax Not applicable

Default Not Applicable

---

SENS<cnm>:DISTortion:MEASure:BAND<bnum>:INITialize
Applicable Models: All with Option S93070xB or S95070A/B

(Write-only) This command ignores any specified band number and deletes all existing bands and adds a single default band back into the Measurement Band Table.

Parameters

- `<cnum>` Any existing modulation distortion channel. If unspecified, value is set to 1
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.

Examples

```
SENS:DIST:MEAS:BAND:INIT 2
```

Query Syntax

Not applicable

Default

Not Applicable

---

SENSe<cnum>:DISTortion:MEASure:BAND<bnum>:NAME <string>

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Assigns a name to the specified distortion measurement band.

Parameters

- `<cnum>` Any existing modulation distortion channel. If unspecified, value is set to 1
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.
- `<string>` String. Distortion measurement band name.

Examples

```
SENS:DIST:MEAS:BAND:NAME "NewBand"
```

Query Syntax

SENSe<cnum>:DISTortion:MEASure:BAND<bnum>:NAME?

Return Type

String

Default

"New Band"

---

SENSe<cnum>:DISTortion:MEASure:BAND<bnum>:NOTCh:IBW <num>
Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the notch integration bandwidth (IBW) for an NPR Notch modulation measurement. The IBW is used to determine total signal power within a specified frequency span. For example, to calculate the total power of a notch composed of 1000 tones spaced 100 kHz apart over a 100 MHz span, the notch power would be integrated over an IBW of 100 MHz.

Parameters
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.
- `<num>` Notch integration bandwidth.

Examples
```
SENS:DIST:MEAS:BAND:NOTC:IBW 10 MHz
```

Query Syntax
```
SENS<cnum>:DISTortion:MEASure:BAND<bnum>:NOTCh:IBW?
```

Return Type
Numeric

Default
10 MHz

```
SENS<cnum>:DISTortion:MEASure:BAND<bnum>:NOTCh:OFFSet <num>
```
Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the notch offset for an NPR Notch modulation measurement. The offset settings are relative to the carrier center frequency (0 Hz).

**Parameters**
- `<cnum>`: Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<bnum>`: Band number from 1 - 100. If unspecified, value is set to 1.
- `<num>`: Notch offset.

**Examples**
```
SENS:DIST:MEAS:BAND:NOTC:OFFS 0
```

**Query Syntax**
```
SENS<cnum>:DISTortion:MEASure:BAND<bnum>:NOTCh:OFFSet?
```

**Return Type**
Numeric

**Default**
0

```
SENS<cnum>:DISTortion:MEASure:BAND<bnum>:TYPE <enum>
```
Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Set and read the modulation distortion type.

Parameters

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<bnum>` Band number from 1 - 100. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - ACPEVM
  - ACP
  - BPWR
  - EVM
  - NPR

Learn about these settings.

Examples

```
SENS:DIST:MEAS:BAND:TYPE ACPEVM
```

Query Syntax

```
SENSe<cnum>:DISTortion:MEASure:BAND<bnum>:TYPE?
```

Return Type

Enumeration

Default

ACPEVM

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the frequency span window used for modeling the DUT’s gain and distortion.

Parameters

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>` Distortion measurement DUT analysis bandwidth in Hz.

Examples

```
SENS:DIST:MEAS:CORR:APER 500000 Hz
```

Query Syntax

```
SENSe<cnum>:DISTortion:MEASure:CORRelation:APERture?
```

Return Type

Numeric
**SENSe<cnum>:DISTortion:MEASure:CORRelation:APERture:AUTO[:STATe] <bool>**

**Applicable Models:** All with Option S93070xB or S95070A/B

(Read-Write) Enable or disable automatic calculation of the frequency span window used for modeling the DUT's gain and distortion.

**Parameters**

- `<cnum>` Any existing modulation distortion channel. If unspecified, value is set to 1.
- `<bool>` Choose from:

  - 0 - **OFF** - Disable DUT analysis bandwidth settings.
  - 1 - **ON** - Enable DUT analysis bandwidth settings.

**Examples**

```
SENS:DIST:MEAS:CORR:APER :AUTO 1
```

**Query Syntax**

`SENSe<cnum>:DISTortion:MEASure:COR Relation:APERture:AUTO[:STATe]?

**Return Type**

Boolean

**Default** 0

---

**SENSe<cnum>:DISTortion:MEASure:FILTer <enum>**

**Applicable Models:** All with Option S93070xB or S95070A/B

(Read-Write) Sets the measurement filter to either None (default) or RRC (root-raised-cosine filter) for EVM and ACP measurements only.

**Parameters**

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<enum>` Choose from:

  - **NONE** - Do not use the measurement filter.
  - **RRC** - Select the RRC measurement filter.

**Examples**

```
SENS:DIST:MEAS:FILTER RRC
```

**Query Syntax**

`SENSe<cnum>:DISTortion:MEASure:FILT?

**Return Type**

Enumeration
**SENSe<cnum>:DISTortion:MEASure:FILTer:ALPHa <num>**

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-Write)* Sets and returns the Alpha factor of the measurement filter.

**Parameters**

- **<cnum>** Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- **<num>** Alpha factor of the measurement filter (0 to 1).

**Examples**

```
SENS:DIST:MEAS:FILT:ALPH 1
```

**Query Syntax**

`SENSe<cnum>:DISTortion:MEASure:FILTer:ALPHa?`

**Return Type**

Numeric

**Default**

0

---

**SENSe<cnum>:DISTortion:MEASure:FILTer:SRATe <num>**

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-Write)* Sets and returns the Symbol Rate of the filter. If `SENSe:DISTortion:MEASure:FILTer:SRATe:AUTO[:STATe]` is ON, the symbol rate from the file is used. If no Symbol Rate is indicated in the file, then the Symbol Rate will be approximated from the bandwidth of the signal.

**Parameters**

- **<cnum>** Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- **<num>** Symbol rate of the measurement filter (10 MHz to 50 GHz).

**Examples**

```
SENS:DIST:MEAS:FILT:SRAT 10E6
```

**Query Syntax**

`SENSe<cnum>:DISTortion:MEASure:FILTer:ALPHa?`

**Return Type**

Numeric

**Default**

0

---

**SENSe<cnum>:DISTortion:MEASure:FILTer:SRATe:AUTO[:STATe] <bool>**
**Applicable Models:** All with Option S93070xB or S95070A/B

**Parameters**

- `<cnum>`: Any existing modulation distortion channel. If unspecified, value is set to 1.
- `<bool>`: Choose from:
  - **0 - OFF**: Do not use Symbol Rate from file.
  - **1 - ON**: Use Symbol Rate from file.

**Examples**

```
SENS:DIST:MEAS:FILT:SRAT:AUTO 1
```

**Query Syntax**

`SENSe<cnum>:DISTortion:MEASure:FILTer:SRATe:AUTO[:STATe]?

**Return Type**

Boolean

**Default**

1

**SENSe<cnum>:DISTortion:MODulate:SOURce <string>**

**Applicable Models:** All with Option S93070xB or S95070A/B

**Parameters**

- `<cnum>`: Any existing modulation distortion channel. If unspecified, value is set to 1
- `<string>`: String. Distortion modulation source name.

**Examples**

```
SENS:DIST:MOD:SOUR "MyModSource"
```

**Query Syntax**

`SENSe<cnum>:DISTortion:MODulate:SOURce?

**Return Type**

String

**Default**

Not Applicable

**SENSe<cnum>:DISTortion:PATH:DUT:INPut <num>**
Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the VNA port number connected to the DUT input.

Parameters
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>` VNA port number connected to the DUT input.

Examples
- `SENS:DIST:PATH:DUT:INPut 1`

Query Syntax
- `SENSe<cnum>:DISTortion:PATH:DUT:INPut?`

Return Type
- Numeric

Default
- 1

---

SENSe<cnum>:DISTortion:PATH:DUT:NOMinal:GAIN <num>

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the nominal DUT gain.

Parameters
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>` Nominal DUT gain in dB.

Examples
- `SENS:DIST:PATH:DUT:NOM:GAIN 0 dB`

Query Syntax
- `SENSe<cnum>:DISTortion:PATH:DUT:NOMinal:GAIN?`

Return Type
- Numeric

Default
- 0

---

SENSe<cnum>:DISTortion:PATH:DUT:NOMinal:NF <num>
Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the nominal DUT noise figure. This value is used by the EVM measurement. This setting adds noise to the measurement for more realistic results. Setting the noise figure to a low value (-200 dB) will make EVM measurements without the effects of noise.

Parameters

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>` Nominal DUT noise figure in dB.

Examples

```
SENS:DIST:PATH:DUT:NOM:NF 0 dB
```

Query Syntax

```
SENSe<cnum>:DISTortion:PATH:DUT:NOMinal:NF?
```

Return Type

Numeric

Default

0

---

**SENSe<cnum>:DISTortion:PATH:DUT:OUTPut <num>**

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the VNA port number connected to the DUT output.

Parameters

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>` VNA port number connected to the DUT output.

Examples

```
SENS:DIST:PATH:DUT:OUTP 2
```

Query Syntax

```
SENSe<cnum>:DISTortion:PATH:DUT:OUTPut?
```

Return Type

Numeric

Default

2

---

**SENSe<cnum>:DISTortion:PATH:SOURce:ATTenuation:INCLude <bool>**
**Applicable Models:** All with Option S93070xB or S95070A/B

**Parameters**
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Do not include attenuator.
  - **1 - ON** - Include attenuator.

**Examples**

```
SENS:DIST:PATH:SOUR:ATT:INCL 1
```

**Query Syntax**

```
SENSe<cnum>:DISTortion:PATH:SOURce:ATTenuation:INCLude?
```

**Return Type**
Boolean

**Default**
1

---

**SENSe<cnum>:DISTortion:PATH:SOURce:NOMinal:AMPLifier <num>**

**Applicable Models:** All with Option S93070xB or S95070A/B

**Parameters**
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>` Nominal source amplifier gain in dB.

**Examples**

```
SENS:DIST:PATH:SOUR:NM:AMPL 0 dB
```

**Query Syntax**

```
SENSe<cnum>:DISTortion:PATH:SOURce:NOMinal:AMPLifier?
```

**Return Type**
Numeric

**Default**
0

---

**SENSe<cnum>:DISTortion:PULSe:RECeiver:AUTO <bool>**
### Applicable Models:
All with Option S93070xB or S95070A/B

**Read-Write** Enables or disables receiver timing auto generation from the source pulse timing.

**Parameters**

- `<cnum>` Any existing modulation distortion channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Do not auto generate receiver timing.
  - **1 - ON** - Auto generate receiver timing.

**Examples**

```
SENS:DIST:PULS:REC:AUTO 1
```

**Query Syntax**
`SENSe<cnum>:DISTortion:PULSe:RECeiver:AUTO?`

**Return Type** Boolean

**Default** 1

---

### SENSE<cnum>:DISTortion:SWEep:CARRier:FREQuency <num>

**Applicable Models:** All with Option S93070xB or S95070A/B

**Read-Write** Sets and returns the carrier center frequency.

**Parameters**

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>` Carrier center frequency.

**Examples**

```
SENS:DIST:SWE:CARR:FREQ 1.5 GHz
```

**Query Syntax**
`SENSe<cnum>:DISTortion:SWEep:CARRier:FREQuency?`

**Return Type** Numeric

**Default** 1.5 GHz

---

### SENSE<cnum>:DISTortion:SWEep:DWELl <num>

---
Applicable Models: All with Option S93070xB or S95070A/B

*(Read-Write)* Sets and returns the delay in seconds before the test signal to allow the RF source to settle.

**Parameters**

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>` Delay time in seconds.

**Examples**

```
SENS:DIST:SWE:DWEL .1
```

**Query Syntax**

`SENSe<cnum>:DISTortion:SWEep:LEVel?`

**Return Type**

Numeric

**Default**

0

---

**SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LEVel <num>**

Applicable Models: All with Option S93070xB or S95070A/B

*(Read-Write)* Sets and returns the carrier power level used for the distortion test at either the input or output of the DUT.

**Parameters**

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>` Power level.

**Examples**

```
SENS:DIST:SWE:POW:CARR:LEV -10 dBm
```

**Query Syntax**

`SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LEVel?`

**Return Type**

Numeric

**Default**

-10 dBm

---

**SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LEVel:PORT <enum>**
**Applicable Models:** All with Option S93070xB or S95070A/B

**Parameters**

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - **DIN1** - DUT input.
  - **DOUT2** - DUT output.

**Examples**

```plaintext
SENS:DIST:SWE:POW:CARR:LEV:PORT DIN1
```

**Query Syntax**

```
SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LEVel:PORT?
```

**Return Type**  
Enumeration

**Default**  
DIN1

---

**SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LIST<index>:TYPE <enum>**

**Applicable Models:** All with Option S93070xB or S95070A/B

**Parameters**

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<index>` Power sweep row number. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - **RAMP** - Selects a ramp type power sweep.
  - **LIST** - Selects a list of power values to define a power sweep.

**Examples**

```plaintext
SENS:DIST:SWE:POW:CARR:LEV1:TYPE RAMP
```

**Query Syntax**

```
SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LEVel<index>:TYPE?
```

**Return Type**  
Enumeration

**Default**  
RAMP

---

**SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LIST<index>:ADD**
**Applicable Models:** All with Option S93070xB or S95070A/B

*(Write-only)* Adds a new row to the power sweep list table.

**Parameters**

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1
- `<index>` Power sweep row number. If unspecified, value is set to 1.

**Examples**

```
SENS:DIST:SWE:POW:CARR:LIST1:ADD
```

**Query Syntax** Not applicable

**Default** Not Applicable

---

**SEnSe`<cnum>`:DIStortion:SWEep:POWer:CARRier:LIST`<index>`:DELeTe**

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Write-only)* Deletes the specified row from the power sweep list table.

**Parameters**

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1
- `<index>` Power sweep row number. If unspecified, value is set to 1.

**Examples**

```
SENS:DIST:SWE:POW:CARR:LIST1:DEL
```

**Query Syntax** Not applicable

**Default** Not Applicable

---

**SEnSe`<cnum>`:DIStortion:SWEep:POWer:CARRier:LIST`<index>`:LEVel `<num>`**
Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the power level used for the specified row in the power sweep list table.

Parameters
  <cnum> Any existing Modulation Distortion channel. If unspecified, value is set to 1.
  <index> Power sweep row number. If unspecified, value is set to 1.
  <num> Power level.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:DIST:SWE:POW:CARR:LIST1:LEV -20 dBm</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LIST<index>:LOAD <fileName>

Applicable Models: All with Option S93070xB or S95070A/B

(Write-only) Specifies the file path to recall a previous power sweep list file.

Parameters
  <cnum> Any existing Modulation Distortion channel. If unspecified, value is set to 1.
  <index> Power sweep row number. If unspecified, value is set to 1.
  <fileName> (String) Power sweep file name (*.csv).

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:DIST:SWE:POW:CARR:LIST1:LOAD &quot;C:/myPowerSweepList.csv&quot;</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LIST<index>:NBW <num>

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Not applicable</td>
</tr>
</tbody>
</table>
**Applicable Models:** All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the noise bandwidth setting used for the specified row in the power sweep list table.

**Parameters**
- `<cnum>`: Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<index>`: Power sweep row number. If unspecified, value is set to 1.
- `<num>`: Noise bandwidth value.

**Examples**
- `SENS:DIST:SWE:POW:CARR:LIST1:NBW 100 Hz`

**Query Syntax**
- `SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LIST<index>:NBW?`

**Return Type**
- Numeric

**Default**
- 100 Hz

---

**SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LIST<index>:NBW:MODE <enum>**

**Applicable Models:** All with Option S93070xB or S95070A/B

(Read-Write) Selects the noise bandwidth mode in the power sweep list table.

**Parameters**
- `<cnum>`: Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<index>`: Power sweep row number. If unspecified, value is set to 1.
- `<enum>`: Choose from:
  - **FIXed** - Sets all noise bandwidth values in the power sweep list table to the same value.
  - **CUSTom** - Each noise bandwidth setting in the power sweep list table may be different.
  - **AUTO** - The Noise BW setting will be used for the minimum power level. As the power level increases, the Noise BW will increase automatically. This results in faster measurements and ensures that the noise error is approximately the same for each power level.

**Examples**

**Query Syntax**
- `SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LIST<index>:NBW:MODE?`
Return Type  Enumeration
Default      FIXed

SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LIST<index>:POINts <num>

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the number of power points to measure for the power sweep list measurement.

Parameters
- <cnum>      Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- <index>     Power sweep row number. If unspecified, value is set to 1.
- <num>       Number of power points to measure.

Examples  SENS:DIST:SWE:POW:CARR:LIST1:POIN 10

Query Syntax  SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LIST<index>:POINts?

Return Type  Numeric
Default      11

SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LIST<index>:RECeiver:ATTenuation <num>

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the receiver attenuation for the specified row in the power sweep list table.

Parameters
- <cnum>      Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- <index>     Power sweep row number. If unspecified, value is set to 1.
- <num>       Receiver attenuation value.

Examples  SENS:DIST:SWE:POW:CARR:LIST1:REC:ATT 0

Query Syntax  SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:LIST<index>:RECeiver:ATTenuation?

Return Type  Numeric
Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Selects the receiver attenuation mode in the power sweep list table.

Parameters

- `<cnum>`: Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<index>`: Power sweep row number. If unspecified, value is set to 1.
- `<enum>`: Choose from:
  - **FIXed**: Sets all receiver attenuation values in the power sweep list table to the same value.
  - **CUSTOM**: Each receiver attenuation setting in the power sweep list table may be different.

Examples

```
```

Query Syntax

```
```

Return Type: Enumeration

Default: `FIXed`

---

Applicable Models: All with Option S93070xB or S95070A/B

(Write-only) Specifies the file path to save a power sweep list file.

Parameters

- `<cnum>`: Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<index>`: Power sweep row number. If unspecified, value is set to 1.
- `<fileName>`: (String) Power sweep file name (*.csv).

Examples

```
SENS:DIST:SWE:POW:CARR:LIST1:SAVE "C:/myPowerSweepList.csv"
```

Query Syntax: Not applicable

Default: Not applicable
SENSe<cnum>:DIStortion:SWeep:POWer:CARRier:LIST<index>:SOURce:ATTenuation <num>

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the source attenuation for the specified row in the power sweep list table.

Parameters
- <cnum>: Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- <index>: Power sweep row number. If unspecified, value is set to 1.
- <num>: Source attenuation value.

Examples
- SENSe<getNum>:DIStortion:SWeep:POWer:CARRier:LIST1:SOURce:ATTenuation 0 dB

Query Syntax
SENSe<cnum>:DIStortion:SWeep:POWer:CARRier:LIST<index>:SOURce:ATTenuation?

Return Type
Numeric

Default
0 dB


Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Selects the source attenuation mode in the power sweep list table.

Parameters
- <cnum>: Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- <index>: Power sweep row number. If unspecified, value is set to 1.
- <enum>: Choose from:
  - FIXed: Sets all source attenuation values in the power sweep list table to the same value.
  - CUSTOM: Each source attenuation setting in the power sweep list table may be different.

Examples

Query Syntax
SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:RAMP:LEVel<index>:STARt <num>

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the starting power level for the power sweep ramp measurement.

Parameters

- <cnum> Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- <index> Power sweep row number. If unspecified, value is set to 1.
- <num> Starting power level.

Examples

```
```

Query Syntax

```
SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:RAMP:LEVel<index>:STARt?
```

Return Type

Numeric

Default

-20 dBm

---

SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:RAMP:LEVel<index>:STOP <num>

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the stop power level for the power sweep ramp measurement.

Parameters

- <cnum> Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- <index> Power sweep row number. If unspecified, value is set to 1.
- <num> Stop power level.

Examples

```
```

Query Syntax

```
SENSe<cnum>:DISTortion:SWEep:POWer:CARRier:RAMP:LEVel<index>:STOP?
```

Return Type

Numeric
SENSe<cnump>:DISTortion:SWEep:POWER:CARRier:RAMP:NBW:AUTO <bool>

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-Write)* Enable or disable increasing noise bandwidth at high powers automatically for faster measurements.

**Parameters**
- `<cnump>` Any existing modulation distortion channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable auto-increase NBW at high powers.
  - **1 - ON** - Enable auto-increase NBW at high powers.

**Examples**
- `SENSe<cnump>:DISTortion:SWEep:POWER:CARRier:RAMP:NBW:AUTO 1`

**Query Syntax**

**Return Type** Boolean

**Default** 0

---


**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-Write)* Sets and returns the number of power points to measure for the power sweep ramp measurement.

**Parameters**
- `<cnump>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>` Number of power points to measure.

**Examples**

**Query Syntax**

**Return Type** Numeric

**Default** 11
SENSe<cnum>:DISTortion:SWEep:POWer:SPARam:LEVel <num>

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the power level used to measure the DUT S-parameters when operating in its linear region. These S-parameters are used to calculate distortion.

Parameters
- <cnum> Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- <num> Power level.

Examples

```
SENS:DIST:SWE:POW:SPAR:LEV -30 dBm
```

Query Syntax
SENSe<cnum>:DISTortion:SWEep:POWer:SPARam:LEVel?

Return Type
Numeric

Default
-30 dBm

SENSe<cnum>:DISTortion:SWEep:RETRace:POWer <enum>

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Keeps RF power ON at the End of Sweep.

Parameters
- <cnum> Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- <enum> Choose from:
  - AUTO- RF Power is not forced off at the end of a sweep.
  - OFF- RF Power will turn off at the end of a sweep.

Examples

```
SENS:DIST:SWE:RETR:POW
```

Query Syntax
SENSe<cnum>:DISTortion:SWEep:RETRace:POWer?

Return Type
Enumeration

Default
AUTO

SENSe<cnum>:DISTortion:SWEep:SPARam:BWIDth <num>
**Applicable Models:** All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the IF bandwidth for the linear S21 sweep.

**Parameters**

- `<cnum>`: Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>`: IF bandwidth.

**Examples**

```
SENS:DIS:SWEP:SPAR:BWID 1 kHz
```

**Query Syntax**

```
SENSe<cnum>:DISTortion:SWEep:SPARam:BWIDth?
```

**Return Type** Numeric

**Default** 1 kHz

---

**SENSe<cnum>:DISTortion:SWEep:SPARam:REUSE <bool>**

**Applicable Models:** All with Option S93070xB or S95070A/B

(Read-Write) Enable or disable the re-use of the previous linear S-parameter measurements if available and skip sweep.

**Parameters**

- `<cnum>`: Any existing modulation distortion channel. If unspecified, value is set to 1.
- `<bool>`: Choose from:
  - 0 - OFF - Disable linear power sweep.
  - 1 - ON - Enable linear power sweep.

**Examples**

```
SENS:DIS:SWEP:SPAR:REUS 1
```

**Query Syntax**

```
SENSe<cnum>:DISTortion:SWEep:SPARam:RESUe?
```

**Return Type** Boolean

**Default** 0

---

**SENSe<cnum>:DISTortion:SWEep:SPARam[:STATe] <bool>**

---
Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Enable or disable S-parameter sweep. If disabled, curve fitting (smoothing) is used instead of the S-parameter sweep. Curve fitting can be adjusted by varying the window aperture using the SENSe:DISTortion:MEASure:CORRelation:APERture command.

### Parameters

- `<cnum>` Any existing modulation distortion channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable S-parameter sweep and use curve fitting.
  - **1 - ON** - Enable S-parameter sweep.

### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe&lt;cnum&gt;:DISTortion:SWEep:SPARam[:STATe]?</td>
<td>Boolean</td>
<td>1</td>
</tr>
</tbody>
</table>

### Query Syntax

SENSe<cnum>:DISTortion:SWEep:SPARam:STEP <num>

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the step size for the linear S21 sweep.

### Parameters

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<num>` Linear S21 step size.

### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe&lt;cnum&gt;:DISTortion:SWEep:SPARam:STEP?</td>
<td>Numeric</td>
<td>1 MHz</td>
</tr>
</tbody>
</table>

### Query Syntax

SENSe<cnum>:DISTortion:SWEep:SPARam:TYPE <enum>

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-Write)* Selects a chirp signal from the external signal generator.

**Parameters**

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<enum>` Choose from:

  **ECHirp** - Selects a chirp signal.

**Examples**

```
SENS:DIST:SWE:SPAR:TYPE ECH
```

**Query Syntax**

```
SENSe<cnum>:DISTortion:SWEep:SPARam:TYPE?
```

**Return Type** Enumeration

**Default** ECHirp

---

**SENSe<cnum>:DISTortion:SWEep:TYPE <enum>**

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-Write)* Selects between a fixed or power sweep type of sweep.

**Parameters**

- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<enum>` Choose from:

  **FIXed** - Measures a modulated signal with a fixed carrier LO frequency and power level.

  **POWer** - Sweeps the total power of the modulated signal. This power may be defined at the input or output of the DUT.

**Examples**

```
SENS:DIST:SWE:TYPE POW
```

**Query Syntax**

```
SENSe<cnum>:DISTortion:SWEep:TYPE?
```

**Return Type** Enumeration

**Default** FIXed

---

**SENSe<cnum>:DISTortion:TABLE:CATalog?**
(Read-only) Returns a list of existing band names from the Measurement Band Table.

**Parameters**

- `<cnum>`: Any existing Modulation Distortion channel. If unspecified, value is set to 1.

**Examples**

```
SENS:DIST:TABL:CAT?
sense2:distortion:table:catalog?
```

**Return Type**: String

**Default**: Not applicable

---

**SENSe<cnum>:DISTortion:TABLE:DATA:CATalog?**

(Read-only) Returns a list of data parameter names corresponding to the currently specified Measurement Type using the SENSe:DISTortion:MEASure:BAND:TYPE command.

**Parameters**

- `<cnum>`: Any existing Modulation Distortion channel. If unspecified, value is set to 1.

**Examples**

```
SENS:DIST:MEAS:BAND:TYPE ACP
SENS:DIST:TABL:DATA:CAT?

"ACP LoOut2 dBc,ACP LoOut2 dBm,ACP LoOut2 dBM/Hz,ACP LoDist21 dBc,ACP LoIBW,ACP LoOffsFreq,ACP UpIn1 dBc,ACP UpIn1 dBm,ACP UpIn1 dBm/Hz,ACP UpOut2 dBc,ACP UpOut2 dBm,ACP UpOut2 dBM/Hz,ACP UpDist21 dBc,ACP UpIBW,ACP UpOffsFreq"
```

**Return Type**: String

**Default**: Not applicable

---

**SENSe<cnum>:DISTortion:TABLE:DATA:VALue? <bandid>,<paramName>**
**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-only)* Returns a list of data values from the Distortion Table for the specified band and parameter name.

**Parameters**
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<bandid>` Band number.
- `<paramName>` Parameter name. For example, "ACP LoOut2 dBc".

**Examples**
- `sense2:distortion:table:data:value? 1,"ACP LoOut2 dBc"

**Return Type** Numeric

**Default** Not applicable

---

**SENSe<cnum>:DISTortion:TABLE:DISPLAY:CATalog?**

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-only)* Returns the currently displayed Distortion Table parameters.

**Parameters**
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.

**Return Type** String of Character values, separated by commas

**Example**
- `SENSe:DIStortion:TABLE:DISPLAY:CATalog?: "Carrier Out2 dBm,Carrier IBW,EVM DistEq21 dBc,ACP LoOut2 dBc,ACP UpOut2 dBc"

**Default** Not applicable

---

**SENSe<cnum>:DISTortion:TABLE:DISPLAY:DELeete <paramName>**
Applicable Models: All with Option S93070xB or S95070A/B

(Write-only) Deletes the specified parameter from the Distortion Table.

Parameters
- <cnum> Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- <paramName> Distortion Table parameter name.

Examples
SENS:DIST:TABL:DISP:DEL "Carrier IBW"

Return Type: String
Default: Not Applicable

SENSe<cnum>:DISTortion:TABLE:DISPLAY:FEED <paramName>

Applicable Models: All with Option S93070xB or S95070A/B

(Write-only) Adds the specified parameter to the Distortion Table.

Parameters
- <cnum> Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- <paramName> Distortion Table parameter name.

Examples
SENS:DIST:TABL:DISP:FEED "Carrier IBW"

Return Type: String
Default: Not Applicable

SENSe<cnum>:DISTortion:TABLE:DISPLAY:FONT <enum>

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Selects between a small- and medium-sized font displayed below the trace area of the display.

Parameters
- <cnum> Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- <enum> Choose from:
  - SMALL - Sets the font size to small.
  - MEDIUM - Sets the font size to medium.

Examples
SENS:DIST:TABL:DISP:FONT SMAL

Query Syntax
SENSe<cnum>:DISTortion:TABLE:DISPLAY:FONT?

Return Type: Enumeration
SENSe<cnum>:DISTortion:TABLE:DISPLAY:SAVE <filepath>

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Write-only)* Specifies the file path to save a modulation distortion table file.

**Parameters**
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<filepath>` (String) Modulation distortion table path and file name (*.csv).

**Examples**
```
SENS:DIST:TABLE:DISPLAY:SAVE "C:/myModDistortionTable.csv"
```

**Query Syntax** Not applicable

**Default** Not applicable

SENSe<cnum>:DISTortion:TABLE:DISPLAY:SORT <enum>

**Applicable Models:** All with Option S93070xB or S95070A/B

*(Read-Write)* Selects between sorting the modulation distortion table by band or by power.

**Parameters**
- `<cnum>` Any existing Modulation Distortion channel. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - `BAND` - Sort modulation distortion table by band.
  - `POWER` - Sort modulation distortion table by power.

**Examples**
```
SENS:DIST:TABLE:DISPLAY:SORT POW
```

**Query Syntax** 
```
SENSe<cnum>:DISTortion:TABLE:DISPLAY:SORT?
```

**Return Type** Enumeration

**Default** `BAND`

SENSe<cnum>:SA:BW:NOISe <num>
Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Sets and returns the noise bandwidth. The Noise Bandwidth is equal to the Resolution BW divided by the Vector Average factor. You cannot directly set the Resolution BW or Vector Average in a Modulation Distortion channel. Resolution BW is always set to its maximum value for the given signal chosen. Vector Average is automatically set depending on the noise bandwidth setting.

Parameters
<cnun> Any existing Modulation Distortion channel. If unspecified, value is set to 1.
<num> Noise bandwidth value.

Examples SENS:SA:BAND:NOIS 100

Query Syntax SENSe<cnun>:SA:BANDwidth:NOISe?

Return Type Numeric

Default 100

SENSe<cnun>:SA:BANDwidth:NOISe:AUTO <bool>

Applicable Models: All with Option S93070xB or S95070A/B

(Read-Write) Enable or disable setting the noise bandwidth to its maximum possible value. This results in the fastest possible sweep but increases the noise floor. If Auto is enabled, then the Vector Average is set to 1, resulting in the widest noise bandwidth.

Parameters
<cnun> Any existing modulation distortion channel. If unspecified, value is set to 1.
<bool> Choose from:

0 - OFF - Disable auto noise bandwidth settings.
1 - ON - Enable auto noise bandwidth settings.

Examples SENS:SA:BAND:NOIS:AUTO 1

Query Syntax SENSe<cnun>:SA:BANDwidth:NOISe:AUTO?

Return Type Boolean

Default 0
DUT Control

When you use the PXI VNAs, you can control the M9341B 8 bit IO through the VNA firmware. The following commands are available when the launcher includes the M9341B.

```
SENSe:DUTControl:M9341:[MODule]
  | [:STATe]
  | :IOTYpe
  | :LEVEL
  | :PIO
    | :TYPE
    | :LEVEL
  | :RFFE
    | :CLOck
    | :CSEQUence
      | :SADDreess
      | :TYPE
      | :BCOunt
      | :ADDreess
  | [:WRITe]:DATA
  | :READ:DATA
  | :COUNt
```

Click on a keyword to view the command details.
**Applicable Models:** All PXIe VNAs

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

- `<mod>`: Module number of M9341B. The number starts from 1 for the leftmost module of M9341B.

- `<bool>`: Module control state. Choose from:
  - **O** or **OFF**: Skips to control the M9341A/B at the specified channel.
  - **1** or **ON**: Enables to control the M9341A/B at the specified channel.

**Examples**

```
SENS:DUTC:M9341
sense2:dutcontrol:m9341?
```

**Query Syntax**

```
:SENSe<cnum>:DUTControl:M9341[:MODule<mnum>][:STATe]?
```

**Return Type**

Boolean

**Default**

OFF or 0

---

:SENSe<cnum>:DUTControl:M9341[:,MODule<mod>]:IOTYpe<iogroup> <iofunc>

**Applicable Models:** All PXIe VNAs

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

- `<mod>`: Module number of M9341B. The number starts from 1 for the leftmost module of M9341B.

- `<iogroup>`: IO group number. Value range, 1 to 4.
  - 1: Group 1 (pins No. 1 and 2)
  - 2: Group 2 (pins No. 3 and 4)
  - 3: Group 3 (pins No. 5 and 6)
  - 4: Group 4 (pins No. 7 and 8)
<iofunc>  set the IO function for the io group. <PARallel | RFFE>.

Examples

SENS:DUTC:M9341:IOTY1 PAR
sense2:dutcontrol:m9341:module2:iotype RFFE

Return Type  Char

Default  PARallel

:SENSe<cnum>:DUTControl:M9341[:MODule<mod>]:LEVel <lvl>

Applicable Models:  All PXIe VNAs

(Read-Write) Sets and reads the output voltage level of the M9341B 8bit I/O

Parameters

<cnum>  Any existing channel number; if unspecified, value is set to 1.

<mod>  Module number of M9341B. The number starts from 1 for the leftmost module of M9341B.

<lvl>  IO level in volt. Value range: 0.9 to 3.5. Step 0.05.

Examples

SENS:DUTC:M9341:LEV 1.5
sense2:dutcontrol:m9341:module2:level?

Query Syntax  :SENSe<cnum>:DUTControl:M9341[:MODule<mnum>]:LEVel?

Return Type  Numeric

Default  1.2

:SENSe<cnum>:DUTControl:M9341[:MODule<mod>]:PIO<iopin>:TYPE <dir>
Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the signal direction type of Parallel IO, for each IO pin. This setting is valid when the io pin function is selected as parallel IO.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<mod>` Module number of M9341B. The number starts from 1 for the leftmost module of M9341B.
- `<iopin>` IO pin number
- `<dir>` IO direction. Choose from: **IN** or **OUT**

**Examples**

SENS:DUTC:M9341:PIO:TYPE IN  
sense2:dutcontrol:m9341:module2:pio2:type?

**Query Syntax**

:SENSe<cnum>:DUTControl:M9341[:MODule<mod>]:PIO<iopin>:TYPE?

**Return Type** Character

**Default** OUT

---

:SENSe<cnum>:DUTControl:M9341[:MODule<mod>]:PIO<iopin>:LEVel <lvl>

Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the signal level of IO pin, high or low. This setting is valid when the io pin function is selected as parallel IO.

If the IO type is IN, this command shall be a read-only command. Write command will cause error.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<mod>` Module number of M9341B. The number starts from 1 for the leftmost module of M9341B.
- `<iopin>` IO pin number
- `<lvlr>` Signal level. Choose from: **HIGH** or **LOW**

**Examples**

SENS:DUTC:M9341:PIO:LEV LOW  
sense2:dutcontrol:m9341:module2:pio2:level?

**Query Syntax**

:SENSe<cnum>:DUTControl:M9341[:MODule<mod>]:PIO<iopin>:LEVel?

**Return Type** Char
Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the RFFE clock rate.

Parameters

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9341B. The number starts from 1 for the leftmost module of M9341B.
- `<clk>`: Clock rate in Hz. Value range, 25kHz to 25000kHz. Possible values are 50000/n, with integer n, 2000 to 2. User can use suffix such as “kHz” and so on.

Examples

```
:SENSe<cnx>:DUTControl:M9341[:MODule<mod>]:RFFE:CLOCK 25KHZ
sense2:dutcontrol:m9341:module2:rffe:clock?
```

Query Syntax

`:SENSe<cnx>:DUTControl:M9341[:MODule<mn>]:RFFE:CLOCk?`

Return Type

numeric

Default 50000

```
:SENSe<cnx>:DUTControl:M9341[:MODule<mod>]:RFFE<rffech>:CSEQuence<csnum>:SADD<adrs>
```
Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the secondary address (“SA” in GUI) for the specified command sequence.

Parameters

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9341B. The number starts from 1 for the leftmost module of M9341B.
- `<rffech>`: RFFE channel number. 1 to 4
- `<csnum>`: RFFE command sequence number. 1 to 16.
- `<adrs>`: DUT RFFE secondary address. 0 to 15.

Examples:

- `sense2:dutcontrol:m9341:module2:rffe:csequence2:saddress?`

Query Syntax:

- `:SENS<cnum>:DUTControl:M9341[:MODule<mod>]:RFFE<rffech>:CSEQuence<csnum>:SADDress?`

Return Type:
- numeric

Default:
- 0

Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the command sequence type for the specified command sequence.

Parameters

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9341B. The number starts from 1 for the leftmost module of M9341B.
- `<rffech>`: RFFE channel number. 1 to 4.
- `<csnum>`: RFFE command sequence number. 1 to 16.
- `<adrs>`: RFFE command sequence type. Choose from:
  - **R0WRite**: Register 0 Write
  - **RREad**: Register Read
**RWRite** : Register Write

**ERRead** : Extended Register Read

**ERWRite** : Extended Register Write

**Examples**

```
:SENS:DUTC:M9341:RFFE:CSEQ:TYPE R0WR
sense2:dutcontrol:m9341:module2:rffe:csequence2:type?
```

**Query Syntax**

`:SENSe<cnum>:DUTControl:M9341[:MODule<mnum>]:RFFE<rffech>:CSEQuence<csnum>:TYPE?`

**Return Type**

Char

**Default**

RREad

---

**:SENSe<cnum>:DUTControl:M9341[:MODule<mod>]:RFFE<rffech>:CSEQuence<csnum>:BCOunt<byt>**

**Applicable Models:** All PXIe VNAs

*(Read-Write)* Sets and reads the byte count for the specified command sequence.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>` Module number of M9341B. The number starts from 1 for the leftmost module of M9341B
- `<rffech>` RFFE channel number. 1 to 4
- `<csnum>` RFFE command sequence number. 1 to 16.
- `<byt>` **Byte Count value. Integer value. The value range is coupled with command sequence type**

<table>
<thead>
<tr>
<th>Command sequence type</th>
<th>Byte count range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register 0 Write</td>
<td>1</td>
</tr>
<tr>
<td>Register Read</td>
<td></td>
</tr>
<tr>
<td>Register Write</td>
<td></td>
</tr>
<tr>
<td>Extended Register Write</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Extended Register Read</td>
<td></td>
</tr>
</tbody>
</table>
Examples
sense2:dutcontrol:m9341:module2:rffe:csequence2:bcount?

Query Syntax
:SENS<cnum>:DUTControl:M9341[:MODule<mnum>]:RFFE<rffech>:CSEQuence<csnum>:BCOUnt?

Return Type
Numeric

Default 1

SENSe<cnum>:DUTControl:M9341[:MODule<mod>]:RFFE<rffech>:CSEQuence<csnum>:ADDR<adrs>

Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the address value for the specified command sequence.

Parameters
- `<cnum>` Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>` Module number of M9341B. The number starts from 1 for the leftmost module of M9341B.
- `<rffech>` RFFE channel number. 1 to 4.
- `<csnum>` RFFE command sequence number. 1 to 16.
- `<adrs>` Address value. Integer value. The value range is coupled with command sequence type setting.

<table>
<thead>
<tr>
<th>Command sequence type</th>
<th>Byte count range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register 0 Write</td>
<td>0 (fixed)</td>
</tr>
<tr>
<td>Register Read</td>
<td>#h00 to #h1F (0-31)</td>
</tr>
<tr>
<td>Register Write</td>
<td></td>
</tr>
<tr>
<td>Extended Register Write</td>
<td>#h00 to #hFF (0-255)</td>
</tr>
<tr>
<td>Extended Register Read</td>
<td></td>
</tr>
</tbody>
</table>

Examples
sense2:dutcontrol:m9341:module2:rffe:csequence2:address?

Query Syntax
:SENS<cnum>:DUTControl:M9341[:MODule<mnum>]:RFFE<rffech>:CSEQuence<csnum>
**SENSe<cnum>:DUTControl:M9341[:MODule<mnum>]:RFFE<rffech>:CSEQuence<csnum>:WRITe:DATA**

**Applicable Models:** All PXIe VNAs

*(Read-Write)* Sets and reads the data values for the specified command sequence.

This command works if the command sequence type is “Register 0 Write” or “Register Write” or “Extended Register Write”. If the command sequence type is “Register Read” or “Extended Register Read”, this command will cause error.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>` Module number of M9341B. The number starts from 1 for the leftmost module of M9341B.
- `<rffech>` RFFE channel number. 1 to 4.
- `<csnum>` RFFE command sequence number. 1 to 16.
- `<adrs>` comma separated list of data values. The value length is coupled with byte count setting. If data list length does not match with byte count setting, write command will cause error.

**Examples**

```
:SENSe:DUTCONtrol:M9341:RFFE:CSEQ:WRITe:DATA 10
sense2:dutcontrol:m9341:module2:rffe:csequence2:write:data?
```

**Query Syntax**

```
:SENSe<cnum>:DUTControl:M9341[:MODule<mnum>]:RFFE<rffech>:CSEQuence<csnum>:READ:DATA?
```

**Return Type**

Comma separated numeric values

**Default** 0
**Applicable Models:** All PXIe VNAs

*(Read only)* Reads the data and parity value pairs from DUT for the specified command sequence.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9341B. The number starts from 1 for the leftmost module of M9341B.
- `<rffech>`: RFFE channel number. 1 to 4.
- `<csnum>`: RFFE command sequence number. 1 to 16.

**Examples**

```
sense2:dutcontrol:m9341:module2:rffe:csequence2:read:data?
```

**Query Syntax**

`:SENSe<cnum>:DUTControl:M9341[:MODule<mnum>]:RFFE<rffech>:CSEQuence<csnum>:READ:DATA?`

**Return Type**

Comma separated numeric values, list of data and parity pairs.

Ex. Byte count is 3 case, return values are below:

```
[data#1],[parity#1],[data#2],[parity#2],[data#3],[parity#3]
```

**Default**

Not applicable

---

**SENSe<cnum>:DUTControl:M9341[:MODule<mnum>]:RFFE<rffech>:CSEQuence<csnum>:COUNT <cnt>**

**Applicable Models:** All PXIe VNAs

*(Read-Write)* Sets and reads the the number of RFFE Command Sequence. If user set the larger value than previously set, new RFFE command sequences will be added with the default parameter value.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9341B. The number starts from 1 for the leftmost module of M9341B.
- `<rffech>`: RFFE channel number. 1 to 4.
- `<cnt>`: RFFE Command Sequence count. 1 to 16.

**Examples**

```
sense2:dutcontrol:m9341:module2:rffe:csequence2:count?
sense2:dutcontrol:m9341:module2:rffe:csequence2:count 10
```

```
Query Syntax: SENSe<cnum>:DUTControl:M9341[:MODule<mnum>:]:RFFE<rfech>:CSEQUence<csnum>

Return Type: Numeric

Default: 0
Controls the frequency offset settings which cause stimulus and response frequencies to be different.

**Note:** These commands replace the previous FOM commands. Although the old commands will continue to work, they can NOT be mixed with these new commands.

SENSe<cnum>:FOM[:STATe] <bool>
**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Turns Frequency Offset ON and OFF. Frequency offset settings are not enabled until this setting is ON.

Send this command (FOM ON) AFTER sending other FOM settings to avoid 'out-or-range' errors.

**Parameters**

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <bool> ON (or 1) - turns FOM ON.
- OFF (or 0) - turns FOM OFF.

**Examples**

```plaintext
SENS:FOM 1
sense2:fom:state on
```

**Query Syntax**

```plaintext
SENSe<cnum>:FOM:STATe?
```

**Return Type**

Boolean

**Default**

OFF

---

**SENSe<cnum>:FOM:CATalog?**

**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-only) Returns a comma-separated list of available range names in the VNA.

Use **SENS:FOM:CAT?** to see a list of available range names.

Use **SENS:FOM:COUNI?** to see a list of available range numbers.

Use **SENS:FOM:RNUM?** to see the range number for a specific name.

Use **SENS:FOM:RANG:NAME?** to see the range name for a specific number.

External devices can appear in the list of range names. Learn more.

**Parameters**

- <cnum> Any existing channel number; if unspecified, value is set to 1.

**Examples**

```plaintext
SENS:FOM:CAT?
returns
"Primary, Source, Receivers"
```

**Return Type**

String
SENSe<cnum>:FOM:COUNT?

**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Read-only)* Returns the number of valid range numbers in the VNA.

**Parameters**
- `<cnum>` Any existing channel number; if unspecified, value is set to 1.

**Examples**
- `SENSe:FOM:COUN?`
- `sense2:fom:count?`

**Query Syntax** `SENSe<cnum>:FOM:COUNT?`

**Return Type** Numeric

**Default** Not Applicable

SENSe<cnum>:FOM:DISPLAY:SELECT <string>

**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Read-Write)* Select the range to be displayed on the VNA x-axis. All traces in the channel have this same x-axis scaling.

**Parameters**
- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<string>` Range name. Case insensitive. Use `SENSe:FOM:CAT?` to see a list of available frequency range names.

**Examples**
- `SENSe:FOM:DISPLAY:SELECT "source2"`
- `sense2:fom:display:select "source"`

**Query Syntax** `SENSe<cnum>:FOM:DISPLAY:SELECT?`

**Return Type** String

**Default** Receivers
Applicable Models: All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-only) Returns the number of a specified range name.

The FOM range items are typically numbered as follows:

1. Primary
2. Source
3. Receivers
4. Source2 (if present)
5. Source3 (if present)

Use SENS:FOM:CAT? to see a list of available range names.

Use SENS:FOM:COUNt? to see a list of available range numbers.

Use SENS:FOM:RANG:NAME? to see the range name for a specific number.

External devices can appear in the list of range names. Learn more.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<string>` Range name for which a number is being queried. Case insensitive.

Examples

<table>
<thead>
<tr>
<th>SENS:FOM:RNUM? &quot;receivers&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>sense2:fom:rnum? &quot;Source2&quot;</td>
</tr>
</tbody>
</table>

Return Type

- Numeric
- Default: Not Applicable

SENSe<cnum>:FOM:RANGe<n>:COUPled <bool>
Applicable Models: All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Sets and returns the state of coupling (ON or OFF) of the specified range to the primary range.

Parameters

<cn> Any existing channel number; if unspecified, value is set to 1.
<n> Range number to couple to primary range. An error is returned when attempting to couple to the Primary range (1).

Use SENS:FOM:CAT? to see a list of available range names.

Use SENS:FOM:COUNt? to see a list of available range numbers.

Use SENS:FOM:RNUM? to see the range number for a specific name.

Use SENS:FOM:RANG:NAME? to see the range name for a specific number.

<bool> ON (or 1) - Couple range to primary range.
OFF (or 0) - Do NOT couple to primary range.

Examples
SENS:FOM:RANG:COUP 1
sense2:fom:range2:coupled 0

Query Syntax
SENSe<cn>:FOM:RANGE<n>:COUPled?

Return Type
Boolean

Default
ON (or 1) Coupled

SENSe<cn>:FOM:RANGE<n>:FREQuency:CW <num>
**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Read-Write)* Sets and returns the CW frequency.

This setting is valid for the primary range, or if the specified range is already *uncoupled* from the primary range and if the *sweep type* is CW.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.

  Use `SENS:FOM:CAT?` to see a list of available range names.

  Use `SENS:FOM:COUNT?` to see a list of available range numbers.

  Use `SENS:FOM:RNUM?` to see the range number for a specific name.

  Use `SENS:FOM:RANG:NAME?` to see the range name for a specific number.

- `<num>` CW frequency value in Hz. Choose any frequency within the range of the VNA.

**Examples**

```
SENS:FOM:RANG:FREQ:CW 1e9
sense2:fom:range2:frequency:cw 10000000
```

**Query Syntax**

`SENSe<cnum>:FOM:RANGE<n>:FREQuency:CW?`

**Return Type**

Numeric

**Default**

Center frequency of the VNA.

```
SENSe<cnum>:FOM:RANGE<n>:FREQuency:DIVisor <num>
```
Applicable Models: All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Sets and returns the divisor value.

This setting is valid only if the specified range is coupled to the primary range.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.

Use SENS:FOM:CAT? to see a list of available range names.

Use SENS:FOM:COUNt? to see a list of available range numbers.

Use SENS:FOM:RNUM? to see the range number for a specific name.

Use SENS:FOM:RANG:NAME? to see the range name for a specific number.

- `<num>` Divisor value (unitless).

Examples

```
SENS:FOM:RANG:FREQ:DIV 3
sense2:fom:range2:frequency:divisor 0
```

Query Syntax

SENS<cnum>:FOM:RANGe<n>:FREQuency:DIVisor?

Return Type Numeric

Default 1

SENSe<cnum>:FOM:RANGe<n>:FREQuency:MUlTiplier <num>

Applicable Models: All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Sets and returns the multiplier value to be used when coupling this range to the primary range.

This setting is valid only if the specified range is coupled to the primary range.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.

Use SENS:FOM:CAT? to see a list of available range names.

Use SENS:FOM:COUNt? to see a list of available range numbers.

Use SENS:FOM:RNUM? to see the range number for a specific name.
Use `SENS:FOM:RANG:NAME?` to see the range name for a specific number.

**<num>**
Multiplier value. (Unitless)

**Examples**
```
SENS:FOM:RANG:FREQ:MULT 1
sense2:fom:range2:frequency:multiplier 2
```

**Query Syntax**
`SENSc<num>:FOM:RANGe<n>:FREQuency:MUlTiplier?`

**Return Type**
Numeric

**Default**
1

---

**SENSe<cnum>:FOM:RANGe<n>:FREQuency:OFFSet <num>**

**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Read-Write)* Sets and returns the offset value to be used when coupling this range to the primary range. Learn more about offset value.

This setting is valid only if the specified range is coupled to the primary range.

**Parameters**

- **<cnum>**
  Any existing channel number; if unspecified, value is set to 1.

- **<n>**
  Range number. If unspecified, value is set to 1.

  Use `SENS:FOM:CAT?` to see a list of available range names.

  Use `SENS:FOM:COUNt?` to see a list of available range numbers.

  Use `SENS:FOM:RNUM?` to see the range number for a specific name.

  Use `SENS:FOM:RANG:NAME?` to see the range name for a specific number.

- **<num>**
  Offset value. (Unitless)

**Examples**
```
SENS:FOM:RANG:FREQ:OFFS 1E9
sense2:fom:range2:frequency:offset 10000000
```

**Query Syntax**
`SENSc<num>:FOM:RANGe<n>:FREQuency:OFFSet?`

**Return Type**
Numeric

**Default**
0

---

**SENSe<cnum>:FOM:RANGe<n>:FREQuency:STARt <num>**

---

4322
**Applicable Models**: All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Sets and returns the Start value of frequency range. Also specify Stop frequency.

This setting is valid for the primary range, or if the specified range is already uncoupled from the primary range and if the sweep type is LOG or LINear.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<n>`: Range number. If unspecified, value is set to 1.

Use `SENS:FOM:CAT?` to see a list of available range names.

Use `SENS:FOM:COUNT?` to see a list of available range numbers.

Use `SENS:FOM:RNUM?` to see the range number for a specific name.

Use `SENS:FOM:RANG:NAME?` to see the range name for a specific number.

- `<num>`: Start value in Hz. Choose any frequency within the range of the VNA.

**Examples**

```
SENSe<FOM<RANG:NAME? 1GHz
sense2:fom:range2:frequency:start 100000000
```

**Query Syntax**

`SENSe<cnum>:FOM:RANGe<n>:FREQuency:STARt?`

**Return Type**

Numeric

**Default**

Minimum frequency of the VNA.

---

**SENSe<cnum>:FOM:RANGe<n>:FREQuency:STOP <num>**

**Applicable Models**: All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Sets and returns the Stop value of frequency range. Also specify Start frequency.

This setting is valid for the primary range, or if the specified range is already uncoupled from the primary range and if the sweep type is LOG or LINear.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<n>`: Range number. If unspecified, value is set to 1.

Use `SENS:FOM:CAT?` to see a list of available range names.

Use `SENS:FOM:COUNT?` to see a list of available range numbers.
Use **SENS:FOM:RNUM?** to see the range number for a specific name.

Use **SENS:FOM:RANG:NAME?** to see the range name for a specific number.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>&lt;num&gt;</strong></td>
<td>Stop value in Hz. Choose any frequency within the range of the VNA.</td>
</tr>
</tbody>
</table>

**Examples**

```plaintext
SENS:FOM:RANG:FREQ:STOP 1e12  
sense2:fom:range2:frequency:stop 10000000000
```

**Query Syntax**

`SENSe<cnum>:FOM:RANGe<n>:FREQuency:STOP?`

**Return Type**

Numeric

**Default**

Maximum frequency of the VNA.

---

**SENSe<cnum>:FOM:RANGe<n>:NAME?**

**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Read-only)* Returns the name of range<n>.

The FOM range items are typically named as follows:

1. Primary
2. Source
3. Receivers
4. Source2 (if present)
5. Source3 (if present)

Use **SENS:FOM:CAT?** to see a list of available range names.

Use **SENS:FOM:COUNt?** to see a list of available range numbers.

Use **SENS:FOM:RNUM?** to see the range number for a specific name.

**Parameters**

- **<cnum>** Any existing channel number; if unspecified, value is set to 1.
- **<n>** Range number. If unspecified, value is set to 1.

**Examples**

```plaintext
SENSe:FOM:RANGe2:NAME?
```

**Return Type**

String

**Default**

Not Applicable
SENSe<cnum>:FOM:RANGe<n>:SWEep:TYPE <char>

Applicable Models: All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Sets and returns the sweep type to be used with the specified range.

This setting is valid only if the specified range is already uncoupled from the primary range.

Learn about Unsupported Sweep Type combinations.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.
<n> Range number. If unspecified, value is set to 1.

Use SENS:FOM:CAT? to see a list of available range names.

Use SENS:FOM:COUNt? to see a list of available range numbers.

Use SENS:FOM:RNUM? to see the range number for a specific name.

Use SENS:FOM:RANG:NAME? to see the range name for a specific number.

<char> Sweep type. Choose from:

CW - Also specify CW frequency.

LINear - Also specify frequency Start/Stop or Center/Span

LOG - Also specify frequency Start/Stop or Center/Span

PHASE - See all Phase sweep settings.

POWER - Also specify power Start/Stop or Center/Span

SEGMENT - Also specify segment sweep settings.

Examples

SENS:FOM:RANG:SWE:TYPE LOG
sense2:fom:range2:sweep:type linear:

Query Syntax

SENSe<cnum>:FOM:RANGe<n>:SWEep:TYPE?

Return Type

Character

Default

Linear
Sense:FOM:Range:Segment Commands

Constructs a segment table for a specified **UNCOPLED** FOM range.

**Note:** Do NOT use Sens:Segment commands for FOM segment sweep.

![Diagram of SENSEe:FOM:RANGEe:SEGMen command structure]

Click on a red keyword to view the command details.

**See Also**

- Other SENSEe:FOM Commands
- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**SENSe<cnum>:FOM:RANGe<n>:SEGMen<s>:ADD**
**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Write-only) Adds a segment.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<s>` Segment number to add. If unspecified, value is set to 1. Segment numbers must be sequential. If a new number is added where one currently exists, the existing segment and those following are incremented by one.

**Examples**

Two Segments exist (1 and 2). The following command will add a new segment (1). The existing (1 and 2) will become (2 and 3) respectively.

```
sense2:fom:range2:segment:add
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:BWIDth[:RESolution] <num>**

**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Sets the IF Bandwidth for the specified segment. First set SENS:FOM:RANGe:SEGM:BWIDth:CONTrol ON. All subsequent segments that are added assume the new IF Bandwidth value.

Valid either for Receiver range or for Primary range when coupled to Receiver.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<s>` Segment number for which to set independent IF Bandwidth.
- `<num>` IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the VNA model. See the list of valid IFBW values. If an invalid number is specified, the analyzer will round up to the closest valid number.

**Examples**

```
SENSe:FOM:RANG:SEGM:BWIDth 100
sense2:fom:range2:segment4:bwidth:resolution 1e3
```

**Query Syntax**

SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:BWIDth[:RESolution]?

**Return Type**

Numeric
SENSe<cnum>:FOM:RANGe<n>:SEGMent:BWIDth[:RESolution]:CONTrol <bool>

Applicable Models: All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Specifies whether the IF Bandwidth resolution can be set independently for each segment. When set, each segment added after this will be set to ON automatically.

Valid either for Receiver range or for Primary range. Primary range value is ignored unless Receiver is coupled to Primary.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <bool> ON (or 1) - turns Bandwidth control ON. Bandwidth can be set for each segment

  OFF (or 0) - turns Bandwidth control OFF. Use the channel IF bandwidth setting instead.

Examples

SENSe:FOM:RANGe:SEGMeNT:BWIDth:CONT 0
sense2:fom:range2:segment:bwidth:resolution:control 1

Query Syntax

SENSe<cnum>:FOM:RANGe<n>:SEGMent:BWIDth[:RESolution]:CONTrol?

Return Type

Boolean

Default

OFF

SENSe<cnum>:FOM:RANGe<n>:SEGMent:COUNt?
**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Read-only)* Returns the number of segments that exist for the specified range.

**Parameters**
- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.

**Examples**
- `SENS: FOM: RANG: SEGM: COUN?`
- `sense2: fom: range2: segment: count?`

**Return Type** Numeric

**Default** Not Applicable

**SENSe`<cnum>`: FOM: RANG`<n>`: SEGM`<s>`: DELete**

**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Write-only)* Deletes the specified sweep segment.

**Parameters**
- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<s>` Number of the segment to delete. If unspecified, value is set to 1.

**Examples**
- `SENS: FOM: RANG: SEGM3: DEL`
- `sense2: fom: range2: segment4: delete`

**Query Syntax** Not Applicable

**Default** Not Applicable

**SENSe`<cnum>`: FOM: RANG`<n>`: SEGMeNT: DELete: ALL**
**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Write-only)* Deletes all sweep segments in the specified range.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.

**Examples**

```
SENS:FOM:RANG:SEGM:DEL:ALL
sense2:fom:range2:segment:delete:all
```

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:FREQuency:CENTer <num>**

**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Read-Write)* Sets and returns the center frequency for the specified sweep segment. Also specify segment frequency span.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<s>` Segment number to modify. Choose any existing segment number.
- `<num>` Center Frequency in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

**Examples**

```
SENS:FOM:RANG:SEGM:FREQ:CENT 1GHz
sense2:fom:range2:segment4:frequency:center 1e9
```

**Query Syntax** `SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:FREQuency:CENTer?`

**Return Type** Numeric

**Default** Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.

---

**SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:FREQuency:SPAN <num>**
Applicable Models: All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Sets and returns the span frequency for the specified sweep segment. Also specify segment center frequency.

Parameters
- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<s>` Segment number to modify. Choose any existing segment number.
- `<num>` Frequency span in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

Examples
```
SENS:FOM:RANG:SEGM:FREQ:SPAN 1GHz
sense2:fom:range2:segment4:frequency:span 1e9
```

Query Syntax
```
SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:FREQuency:SPAN?
```

Return Type Numeric

Default If first segment, frequency span of the analyzer. Otherwise 0.

---

SENS<cnum>:FOM:RANGe<n>:SEGMent<s>:FREQuency:STARt <num>

Applicable Models: All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Sets and returns the start frequency for the specified sweep segment. Also specify segment stop frequency.

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

Parameters
- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<s>` Segment number to modify. Choose any existing segment number.
- `<num>` Start frequency in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

Examples
```
SENS:FOM:RANG:SEGM:FREQ:STAR 1GHz
sense2:fom:range2:segment4:frequency:start 1e9
```

Query Syntax
```
SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:FREQuency:STARt?
```

---

4331
SENSe<cnump>:FOM:RANGen>:SEGMenst<s>:FREQuency:STOP <nump>

Applicable Models: All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Sets and returns the stop frequency for the specified sweep segment. Also specify segment start frequency.

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

Parameters

- <cnump> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <s> Segment number to modify. Choose any existing segment number.
- <nump> Stop frequency in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

Examples

- sense2:fom:range2:segment4:frequency:stop 1e9

Query Syntax

SENSe<cnump>:FOM:RANGen>:SEGMenst<s>:FREQuency:STOP?

Return Type

- Numeric

Default

- Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.
Applicable Models: All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

(Read-Write) Sets the Port Power level for the specified sweep segment. First set SENS:FOM:RANG:SEGM:POW:CONTrol ON.

When port power is Coupled, setting port power for one port will apply port power for all source ports. All subsequent segments that are added assume the new Power Level value.

Valid either for Source ranges or for Primary range when coupled to the source.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<s>` Segment number to modify. Choose any existing segment number.
- `<p>` Port number of the source. If unspecified, value is set to 1.
- `<num>` Power level in dBm.

Note: The range of settable power values depends on the VNA model and if source attenuators are installed. To determine the range of values, send SOUR:POW? MAX and SOUR:POW? MIN. (SOUR:POW:ATT:AUTO must be set to ON).

Actual achievable leveled power depends on frequency.

Examples

- `sense2:fom:range2:segment4:power2:level 5`

Query Syntax

SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:POWer<p>[:LEVel]? Return Type

Numeric Default

0

SENSe<cnum>:FOM:RANGe<n>:SEGMent:POWer[:LEVel]:CONTrol <bool>
**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Read-Write)* Specifies whether Power Level is to be set independently for each segment.

Valid either for Source ranges or for Primary range. Primary range value is ignored unless Source is coupled to Primary.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<bool>` ON (or 1) - Power level will be set for each segment.
  
  OFF (or 0) - Use the channel power level setting.

**Examples**

```
SENS:FOM:RANG:SEGM:POW:CONT 0
sense2:fom:range2:segment:power:control on
```

**Query Syntax**

```
SENSe<cnum>:FOM:RANGe<n>:SEGMent:POWer[:LEVel]:CONTrol?
```

**Return Type** Boolean

**Default** OFF (or 0)

---

**SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>[:STATe] <bool>**

**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Read-Write)* Turns the specified sweep segment ON or OFF.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<s>` Segment number to be turned ON or OFF. Choose any existing segment number.
- `<bool>` ON (or 1) - turns segment ON.
  
  OFF (or 0) - turns segment OFF.

**Examples**

```
SENS:FOM:RANG:SEGM 0
sense2:fom:range2:segment:state on
```

**Query Syntax**

```
SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>[:STATe]?
```

**Return Type** Boolean
SENSe\textless\textit{cnum}\textgreater;:FOM:RANG\textless\textit{n}\textgreater;:SEG\textit{m}ent\textless\textit{s}\textgreater;:SWEep:POINts \textless\textit{num}\textgreater;

**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Read-Write)* Sets the number of data points for the specified sweep segment.

**Parameters**

\textless\textit{cnum}\textgreater; Any existing channel number; if unspecified, value is set to 1.

\textless\textit{n}\textgreater; Range number. If unspecified, value is set to 1.

\textless\textit{s}\textgreater; Segment number to modify. Choose any existing segment number.

\textless\textit{num}\textgreater; Number of points in the segment. The total number of points in all segments cannot exceed 20001. A segment can have as few as 1 point.

**Examples**

```
sense2:fom:range2:segment4:sweep:points 201
```

**Query Syntax**

SENSe\textless\textit{cnum}\textgreater;:FOM:RANG\textless\textit{n}\textgreater;:SEG\textit{m}ent\textless\textit{s}\textgreater;:SWEep:POINts?

**Return Type** Numeric

**Default** 21

SENSe\textless\textit{cnum}\textgreater;:FOM:RANG\textless\textit{n}\textgreater;:SEG\textit{m}ent\textless\textit{s}\textgreater;:SWEep:TIME \textless\textit{num}\textgreater;

**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Read-Write)* Sets the time the VNA takes to sweep the specified segment.

Valid ONLY for receiver ranges.

**Parameters**

\textless\textit{cnum}\textgreater; Any existing channel number; if unspecified, value is set to 1.

\textless\textit{n}\textgreater; Range number. If unspecified, value is set to 1.

\textless\textit{s}\textgreater; Segment number for which to set sweep time.

\textless\textit{num}\textgreater; Sweep time in seconds. Choose a number between 0 and 100

**Examples**

```
SENS:FOM:RANG:SEG:M:SWE:TIME 1
sense2:fom:range2:segment3:sweep:time .1
```

**Query Syntax**

SENSe\textless\textit{cnum}\textgreater;:FOM:RANG\textless\textit{n}\textgreater;:SEG\textit{m}ent\textless\textit{s}\textgreater;:SWEep:TIME?
SENSe<cnum>:FOM:RANGe<n>:SEGMenT:SWEep:TIME:CONTrol <bool>

**Applicable Models:** All with FOM Options (S9x080A/B, S9x082A/B, S9x083A/B)

*(Read-Write)* Specifies whether Sweep Time can be set independently for each sweep segment.

Valid either for Receiver ranges or for Primary range. Primary range value is ignored unless Receiver is *coupled* to Primary.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<bool>` **ON** (or 1) - Sweep time will be set for each segment.
  **OFF** (or 0) - Use the channel sweep time setting.

**Examples**

- `SENSe:FOM:RANG:SEG:SWEP:TIME:CONT 1`
- `sense2:fom:range2:segment:sweep:time:control off`

**Query Syntax**

`SENSe<cnum>:FOM:RANGe<n>:SEGMenT:SWEep:TIME:CONTrol?`

**Return Type** Boolean

**Default** OFF
Sets the sweep frequencies of the analyzer.

SENSe:FREQQuency

| CENTer
| STEP
| AUTO
| SIZE
| CW | FIXed
| SPAN
| FULL
| STARt
| STOP

Click on a keyword to view the command details.

see Also

- Example using some of these commands.
- Learn about Frequency Sweep
- Synchronizing the Analyzer and Controller
- SCPI Command Tree
- See CALCulate:MEASure:X:VALues for frequency point data.

SENSe<cnum>:FREQuency:CENTer <num>
### SENS:FREQ:CENT

**Applicable Models:** All

**(Read-Write)** Sets the center frequency of the analyzer.

**Note:** When the sweep type is "segment sweep", this command is not used.

#### Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Center frequency. Choose any number between the **minimum** and **maximum** frequency limits of the analyzer. Units are Hz.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

#### Examples

```plaintext
SENS:FREQ:CENT 1000000
sense2:frequency:center 1mhz
sense2:frequency:center 1e6
```

#### Query Syntax

SENS<cnum>:FREQuency:CENTer?

#### Return Type

**Numeric**

#### Default

Center of the analyzer's frequency span

---

### SENS<cnum>:FREQ:CENT:STEP:AUTO

**Applicable Models:** All

**(Read-Write)** Sets and reads how the center frequency step size is set. When TRUE, center steps by 5% of span. When FALSE, center steps by STEP:SIZE value.

#### Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<bool>`: Choose from:
  - **ON** (or 1) - Step size is set automatically.
  - **OFF** (or 0) - Step size is set manually using `SENS:FREQ:CENT:STEP:SIZE`.

#### Examples

```plaintext
SENS:FREQ:CENT:STEP:AUTO 1
```

#### Query Syntax

SENS<cnum>:FREQuency:CENTer:STEP:AUTO?

#### Return Type

**Boolean**

#### Default

OFF
SENSe<cnum>:FREQuency:CENTer:STEP:SIZE <num>

Applicable Models: All

(Read-Write) Sets the center frequency step size of the analyzer. This command sets the manual step size (only valid when STEP:AUTO is FALSE).

Parameters
- <cnum> Any existing channel number. If unspecified, value is set to 1
- <num> Step size in Hz. Choose a value below the stop frequency of the analyzer.

Examples
```
SENS:FREQ:CENT:STEP:SIZE 1e9
```

Query Syntax
```
SENSe<cnum>:FREQuency:CENTer:STEP:SIZE?
```

Return Type
Numeric

Default
Default is 40 MHz. When STEP:AUTO is TRUE, this value is ignored.

SENSe<cnum>:FREQuency[:CW | :FIXed] <num>

Applicable Models: All

(Read-Write) Sets the Continuous Wave (or Fixed) frequency. Must also send `SENS:SWEEP:TYPE CW` to put the analyzer into CW sweep mode.

Parameters
- <cnum> Any existing channel number. If unspecified, value is set to 1
- <num> CW frequency. Choose any number between the minimum and maximum frequency limits of the analyzer. Units are Hz.

This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples
```
SENS:FREQ 1000000
SENS:FREQ:CW MIN
sense2:frequency:fixed 1mhz
```

Query Syntax
```
SENSe<cnum>:FREQuency[:CW | :FIXed]?
```

Return Type
Numeric

Default
1 GHz

SENSe<cnum>:FREQuency:SPAN <num>
Applicable Models: All

(Read-Write) Sets the frequency span of the analyzer.

Note: When the sweep type is "segment sweep", this command is not used.

Parameters

<cnm> Any existing channel number. If unspecified, value is set to 1
<num> Frequency span in Hz. Choose any number from 70 (minimum) and the maximum frequency span of the analyzer.

This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:FREQ:SPAN 1000000</td>
</tr>
<tr>
<td>sense2:frequency:span max</td>
</tr>
</tbody>
</table>

Query Syntax

SENSe<cnm>:FREQuency:SPAN?

Return Type

Numeric

Default

Maximum frequency span of the analyzer

---

SENSe<cnm>:FREQuency:SPAN:FULL

Applicable Models: All

(Write-Only) Sets the frequency span to the entire frequency range of the analyzer.

Parameters

<cnm> Any existing channel number. If unspecified, value is set to 1

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:FREQ:SPAN:FULL</td>
</tr>
</tbody>
</table>

Query Syntax

Not Applicable

Default

Maximum frequency span of the analyzer

---

SENSe<cnm>:FREQuency:STARt <num>
**Applicable Models:** All

(Read-Write) Sets the start frequency of the analyzer.

**Note:** When the sweep type is "segment sweep", this command is not used.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Start frequency. Choose any number between the **MIN** and **MAX** frequency limits of the analyzer. Units are Hz.

If FREQ:START is set greater than FREQ:STOP, then the stop frequency is set to the start frequency + frequency span.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See **SCPI Syntax** for more information.

**Examples**

```plaintext
SENSe:FREQ:STAR 1000000
sense2:FREQ:START MIN
```

**Query Syntax**

`SENSe<cnum>:FREQuency:STARt?`

**Return Type**

Numeric

**Default**

Minimum frequency of the analyzer

---

**SENSe<cnum>:FREQuency:STOP <num>**

**Applicable Models:** All

(Read-Write) Sets the stop frequency of the analyzer.

**Note:** When the sweep type is "segment sweep", this command is not used.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Stop frequency. Choose any number between 70 (minimum) and **maximum** frequency limits of the analyzer. Units are Hz.

If FREQ:STOP is set less than FREQ:START, then the start frequency is set to the stop frequency - frequency span.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See **SCPI Syntax** for more information.
| Examples          | SENS:FREQ:STOP 1000000  
<table>
<thead>
<tr>
<th></th>
<th>sense2:frequency:stop max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Syntax</td>
<td>SENSE&lt;cnum&gt;:FREQuency:STOP?</td>
</tr>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>Maximum frequency of the analyzer</td>
</tr>
</tbody>
</table>
SENSe:GCSetup Commands

Controls the Gain Compression configuration.

```
SENSe:GCSetup:

AMOd <enum>

COMPression:
  | ALGorithm <enum>
  | BACKoff:LEVel <num>
  | DELTa:X <num>
  | DELTa:Y <num>
  | INTerpolate
  | [:STATe]
  | LEVel <num>
  | PHASe
  | LEVel
  | MODE
  | SATuration:LEVel

EOSoperation <string>

MIXer
  | REFerence

PMAP
  | INPut?
  | OUTPut?
  | SOURce
  | OVERride

POWer:
  | LINear:INPut:COMPute:APERture
```
Click on a keyword to view the command details.
See Also

Other Gain Compression commands

- CALCulate:MEASure:DEFine - creates a gain compression measurement.
- CALC:MEAS:GCMes:ANAL - Gain Compression Analysis settings
- GCA Calibration uses the Guided Calibration commands, except for the following:
  - Sens:Corr:GCS:Power - sets power level for Source Power Cal

GCX

- Setup Mixer using Sense:Mixer commands
- Calibrate using SMC commands and Guided commands

- Example Programs
  - Create and Cal a Gain Compression Measurement
  - Create and Cal a GCX Measurement
- Learn about Gain Compression Application
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

SENSe<ch>:GCSetup:AMODe <enum>

Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the method by which gain compression data is acquired.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.
<enum> Choose from:
  - PFREQuency - 2D Power Per Frequency
  - FPOWer - 2D Frequency Per Power
  - SMARtsweep - Smart Sweep
SENSe<ch>:GCSetup:COMPression:ALGorithm <enum>

Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the algorithm method used to compute gain compression.

Parameters

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<enum>` Algorithm method. Choose from:
  
  1. **CFLG** - compression from linear gain
  2. **CFMG** - compression from maximum gain
  3. **BACKoff** - compression from BackOff
  4. **XYCOM** - X/Y Compression
  5. **SAT** - compression from saturation

Examples

SENSe:GCS:COMP:ALG BACK

SENSe:gcsetup:compression:algorithm XYcom

Query Syntax

SENSe<ch>:GCSetup:COMPression:ALGorithm? 

Return Type

Enumeration

Default

CFLG

SENSe<ch>:GCSetup:COMPression:BACKoff:LEVel <num>
Applicable Models: All with Gain Compression Option

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Backoff value in dB. Choose a value between 1 and 99.

**Examples**

- SENS:GCS:COMP:BACK:LEV 10
- sense:gcsetup:compression:backoff:level 5

**Query Syntax**

SENS<ch>:GCSetup:COMPression:BACKoff:LEVel?

**Return Type** Numeric

**Default** 10

---

SENSe<ch>:GCSetup:COMPression:DELTa:X <num>

Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the 'X" value in the delta X/Y compression algorithm.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` X value in dB. Choose a value from .01 to 10.

**Examples**

- SENS:GCS:COMP:DEL:T:X 9
- sense:gcsetup:compression:delta:X 8

**Query Syntax**

SENSe<ch>:GCSetup:COMPression:DELTa:X?

**Return Type** Numeric

**Default** 10

---

SENSe<ch>:GCSetup:COMPression:DELTa:Y <num>
Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the 'Y' value in the delta X/Y compression algorithm.

Parameters
- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Y value in dB. Choose a value from .01 to 10.

Examples
- SENS:GCS:COMP:DELT:Y 9
- sense:gcsetup:compression:delta:y 8

Query Syntax
- SENSE<ch>:GCSsetup:COMPression:DELTa:Y?

Return Type
- Numeric

Default
- 9

SENSe<ch>:GCSetup:COMPression:INTerpolate[:STATe] <bool>

Applicable Models: All with Gain Compression Option

(Read-Write) Sets whether or not interpolation should be performed on 2D measured compression data. Applies ONLY to 2D acquisition modes.

Parameters
- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - ON or (1) Interpolate the results
  - OFF or (0) Do NOT interpolate the results but return the value closest to compression.

Examples
- SENS:GCS:COMP:INT 1
- sense:gcsetup:compression:interpolate off

Query Syntax
- SENSE<ch>:GCSsetup:COMPression:INTerpolation?

Return Type
- Boolean

Default
- OFF

SENSe<ch>:GCSetup:COMPression:LEVel <num>
**Applicable Models:** All with Gain Compression Option

(Read-Write) Set and read the desired gain reduction (from reference gain).

This value is used for Compression from Linear Gain and Compression from Maximum Gain.

Use `SENS:GCS:COMP:ALG CFLG` to set this compression algorithm.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Compression level in dB. Choose a value between .01 and 100.

**Examples**

```
SENS:GCS:COMP:LEV 1
sense:gcsetup:compression:level 3
```

**Query Syntax** `SENSe<ch>:GCSetup:COMPression:LEVel?`

**Return Type** Numeric

**Default** 1

---

**SENSe<ch>:GCSetup:COMPression:PHASe:LEVel <num>**

**Applicable Models:** All with Gain Compression Option

(Read-Write) Set and read the desired phase to measure compression. This is only used when `SENSe:GCSetup:COMPression:PHASe:MODE` is set to PHASe or BOTH.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Phase level in degrees. Choose a value between .01 and 360.

**Examples**

```
SENS:GCS:COMP:PHAS:LEV 0.01
sense:gcsetup:compression:phase:level 0.01
```

**Query Syntax** `SENSe<ch>:GCSetup:COMPression:PHASe:LEVel?`

**Return Type** Numeric

**Default** 2

---

**SENSe<ch>:GCSetup:COMPression:PHASe:MODE <enum>**
Applicable Models: All with Gain Compression Option

(Read-Write) Set and read compression format to be either magnitude, phase, or magnitude and phase. This option is only available with the 2D type of compression sweeps.

Parameters

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - **MAGNitude** - compression against magnitude
  - **PHASE** - compression against phase
  - **BOTH** - compression against magnitude and phase

Examples

```
SENS:GCS:COMP:PHAS:MODE PHAS
sense:gcsetup:compression:phase:mode both
```

Query Syntax

SENSe<ch>:GCSetup:COMPression:PHASe:MODE?

Return Type

Enumeration

Default

MAGNitude

SENSe<ch>:GCSetup:COMPression:SATuration:LEVel <num>

Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the deviation dB from the maximum Pout. This is the point of saturation.

Use **SENS:GCS:COMP:ALG CFLG** to set this compression algorithm.

Parameters

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Saturation level in dB. Choose a value between .01 and 10.

Examples

```
SENS:GCS:COMP:SAT:LEV 1
sense:gcsetup:compression:saturation:level 3
```

Query Syntax

SENSe<ch>:GCSetup:COMPressionSATuration:LEVel?

Return Type

Numeric

Default

.1 dB

SENSe<ch>:GCSetup:EOSoperation <enum>
**Applicable Models:** All with Gain Compression Option

*(Read-Write)* This setting is used to protect a sensitive device from too much power during the sweep retrace. Other instrument settings or channels may over-ride this setting. [Learn more.](#)

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<enum>` End Of Sweep operation. Choose from:
  - STANdard Use the default VNA method. [Learn more.](#)
  - POFF Always turn power OFF while waiting.
  - PSTArt Sweep Start power
  - PSTOp Sweep Stop power.

**Examples**

```
SENS:GCS:EOS PSTA
sense:gcsetup:eosoperation standard
```

**Query Syntax**

`SENSe<ch>:GCSetup:EOSoperation?`

**Return Type**

Enumeration

**Default**

STANdard

---

**SENSe<ch>:GCSetup:MIIXer:REFerence <bool>**

**Applicable Models:** All with Gain Compression Option

*(Read-Write)* Set and read the state of the mixer reference. This option is only available in Gain Compression for Converters (GCX), and is only used when `SENSe:GCSetup:COMPression:PHASE:MODE` is set to PHASE or BOTH.

To improve the noise of the phase measurements in GCX, an optional user supplied external reference can be added to the R1 loop. The LO for the reference mixer should be common with the LO for the DUT mixer. After this hardware is added, and this command is set to ON, the GCX application will use the reference mixer to improve the phase measurements.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
<bool> Choose from:

ON or (1) Enable mixer reference.

OFF or (0) Disable mixer reference.

Examples
SENS:GCS:MIX:REF 1
sense:gcsetup:mixer:reference off

Query Syntax SENSE<ch>:GCSetup:MIXer:REFerence?

Return Type Boolean

Default OFF

SENSe<ch>:GCSetup:PMAP <in>,<out>

Applicable Models: All with Gain Compression Option

(Write-only) Set the DUT-to-VNA port mapping for the Gain Compression measurement.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

<in> VNA port which is connected to the DUT input.

<out> VNA port which is connected to the DUT output.

Examples SENS:GCS:PMAP 1,2
sense:gcsetup:pmap 2,1

Query Syntax Not Applicable

Default 1,2

SENSe<ch>:GCSetup:PMAP:INPut?
Applicable Models: All with Gain Compression Option

(Read-only) Read the VNA port number to be connected to the DUT Input.

Use SENS:GCS:PORTMap to set the port mapping.

Parameters

Parameters

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

Examples

Examples

Examples

SENS:GCS:PMAP:INP?

sense:gcsetup:pmap:input?

Return Type Numeric

Default 1

SENSe<ch>:GCSetup:PMAP:OUTPut?

Applicable Models: All with Gain Compression Option

(Read-only) Read the VNA port number to be connected to the DUT Output.

Parameters

Parameters

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

Examples

Examples

Examples

SENS:GCS:PMAP:OUTP?

sense:gcsetup:pmap:output?

Return Type Numeric

Default 2

SENSe<ch>:GCSetup:PMAP:SOURce:OVERride <bool>
Applicable Models: All with Gain Compression Option

(Read-Write) This function is used when GCX channels with a reference mixer and one GCA channel without a reference mixer are set up simultaneously and you want to use the same path configuration for all channels. This remaps the source port to Port 1.

This is only used when \texttt{SENSe:GCSetup:COMPression:PHASe:MODE} is set to \texttt{PHASe} or \texttt{BOTH}.

Parameters

\begin{itemize}
\item \texttt{<ch>} Any existing GCA/GCX channel. If unspecified, value is set to 1.
\item \texttt{<bool>} Choose from:
\begin{itemize}
\item \texttt{ON} or (1) Use same path configuration for all channels.
\item \texttt{OFF} or (0) Do not use the same path configuration for all channels.
\end{itemize}
\end{itemize}

Examples
\begin{verbatim}
SENS:GCS:PMAP:SOUR:OVER 1
sense:gcsetup:pmap:source:override off
\end{verbatim}

Query Syntax \texttt{SENSe<ch>:GCSetup:POWer:LINear:INPut:COMPute:APERture?}

Return Type Boolean

Default OFF

SENSe<ch>:GCSetup:POWer:LINear:INPut:COMPute:APERture <num>

Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the aperture to use when computing the linear input power.

This is only used when \texttt{SENSe:GCSetup:COMPression:PHASe:MODE} is set to \texttt{PHASe} or \texttt{BOTH}. In that mode, the linear input power is computed using a least squares fit of the linear region. This command sets the width, or aperture, of the linear input region. The default is 5%.

Parameters

\begin{itemize}
\item \texttt{<ch>} Any existing GCA channel. If unspecified, value is set to 1.
\item \texttt{<num>} Input percentage. Choose a value from 0 to 25.
\end{itemize}

Examples
\begin{verbatim}
SENS:GCS:POW:LIN:INP:COMP:APER 5
\end{verbatim}

Query Syntax \texttt{SENSe<ch>:GCSetup:POWer:LINear:INPut:COMPute:APERture?}

Return Type Numeric
SENSe<ch>:GCSetup:POWer:LINear:INPut:LEVel <num>

Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the input power at which Linear Gain and all S-parameters are measured.

Parameters

- `<ch>`: Any existing GCA channel. If unspecified, value is set to 1.
- `<num>`: Input power level in dBm. Choose a value from +30 to (-30).

Examples

```
SENSe:GCS:POW:LIIN:INP:LEVL 0
sense:gcsetup:power:linear:input:level -10
```

Query Syntax

`SENSe<ch>:GCSetup:POWer:LINear:INPut:LEVel?`

Return Type

Numeric

Default

-25 dBm

SENSe<ch>:GCSetup:POWer:REVerse:LEVel <num>

Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the reverse power level to the DUT. This is applied to the DUT output port when making reverse measurements like S22.

Parameters

- `<ch>`: Any existing GCA channel. If unspecified, value is set to 1.
- `<num>`: Reverse power level in dBm. Choose a value from +30 to (-30).

Examples

```
SENSe:GCS:POW:REVE:LEVL 0
sense:gcsetup:power:reverse:level -5
```

Query Syntax

`SENSe<ch>:GCSetup:POWer:REVerse:LEVel?`

Return Type

Numeric

Default

-5
**Applicable Models:** All with Gain Compression Option

(Read-Write) Set and read the start power level.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Start power level in dBm. Choose a value from +30 to (-30).

**Examples**

```
SENS:GCS:POW:STAR:LEV 0
sense:gcsetup:power:start:level -5
```

**Query Syntax**

SENSe<ch>:GCSetup:POWer:STARt:LEVel?

**Return Type** Numeric

**Default** -25

SENSe<ch>:GCSetup:POWer:STOP:LEVel <num>

**Applicable Models:** All with Gain Compression Option

(Read-Write) Set and read the stop power level.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Stop power level in dBm. Choose a value from +30 to (-30).

**Examples**

```
SENS:GCS:POW:STOP:LEV 0
sense:gcsetup:power:stop:level -5
```

**Query Syntax**

SENSe<ch>:GCSetup:POWer:STOP:LEVel?

**Return Type** Numeric

**Default** -5

SENSe<ch>:GCSetup:SAFE:CPADjustment <num>
**Applicable Models:** All with Gain Compression Option

*(Read-Write)* Set and read the Safe Sweep COARSE power adjustment. [Learn more.]

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Coarse power adjustment setting in dBm. Choose a value between 0 and 6.

**Examples**

```
SENS:GCS:SAFE:CPAD 2
sense:gcsetup: safe:cpadjustment 3.5
```

**Query Syntax**

`SENSe<ch>:GCSetup:SAFE:CPADjustment?`

**Return Type** Numeric

**Default** 3.0

---

**SENSe<ch>:GCSetup:SAFE:DC:MLimit <num>**

**Applicable Models:** All with Gain Compression Option

*(Read-Write)* Set and read the maximum limit of the external DC device.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Maximum DC level.

**Examples**

```
SENS:GCS:SAFE:DC:MLIM -5
```

**Query Syntax**

`SENSe<ch>:GCSetup:SAFE:DC:MLIMIT?`

**Return Type** Numeric

**Default** -5

---

**SENSe<ch>:GCSetup:SAFE:DC:PARameter <value>**
Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the name of the external DC device.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

<value> (String) Name of the external DC device.

Examples

SENS:GCS:SAFE:DC:PAR "MyDCDevice"

sense:gcsetup:safe:dc:parameter "MYDCDevice"

Query Syntax

SENSe<ch>:GCSetup:SAFE:DC:PARameter?

Return Type String

Default Not Applicable

SENSe<ch>:GCSetup:SAFE:ENABle <bool>

Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the (ON | OFF) state of Safe Sweep mode.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

<num> (Boolean) - Safe Sweep state. Choose from:

OFF (or 0) - Disable Safe Sweep

ON (or 1) - Enable Safe Sweep

Examples

SENS:GCS:SAFE:ENAB 0

sense:gcsetup:safe:enable 1

Query Syntax

SENSe<ch>:GCSetup:SAFE:ENABle?

Return Type Boolean

Default 0

SENSe<ch>:GCSetup:SAFE:FPADjustment <num>
Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the Safe Sweep FINE power adjustment. Learn more

Parameters

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Fine power adjustment setting in dBm. Choose a value between 0 and 3.

Examples

```
SENS:GCS:SAFE:FPAD 2
sense:gcsetup:safe:fpadjustment .5
```

Query Syntax

SENS<ch>:GCSetup:SAFE:FPADjustment?

Return Type
Numeric

Default
1.0 dBm

SENSe<ch>:GCSetup:SAFE:FTHReshold <num>

Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the compression level in which Safe Sweep changes from the COARSE power adjustment to the FINE power adjustment. Learn more

Parameters

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Threshold setting in dB. Choose a value between 0 and 3.

Examples

```
SENS:GCS:SAFE:FTHR .1
sense:gcsetup:SAFE:fthreshold .75
```

Query Syntax

SENS<ch>:GCSetup:SAFE:FTHReshold?

Return Type
Numeric

Default
0.5 dB

SENSe<ch>:GCSetup:SAFE:MLimit <num>
(Read-Write) When the VNA port that is connected to the DUT Output measures this value, the input power to the DUT is no longer incremented at that frequency. Safe Mode must be enabled with SENS:GCS:SAFE:ENAB ON Learn more

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Maximum power limit in dBm. Choose a value from -100 to +100

**Examples**

```
SENS:GCS:SAFE:MLIM 20
sense:gcsetup:Safe:mlimit 30
```

**Query Syntax**

```
SENSe<ch>:GCSetup:SAFE:MLIMit?
```

**Return Type**

Numeric

**Default**

30

SENSe<ch>:GCSetup:SFAilures?

**Applicable Models:** All with Gain Compression Option

(Read-only) Returns a comma-separated list of the frequency indexes that were out of tolerance for SMART Sweep mode, or at the power limit for 2D acquisition modes. Zero (0) is the first frequency data point.

Must be Single triggered. Invalid results occur if the GCA channel is continuously sweeping.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.

**Examples**

```
SENS:GCS:SFA?
sense:gcsetup:sfailures?
```

**Return Type**

Comma-separated list of frequency indexes.

**Default**

Not Applicable

SENSe<ch>:GCSetup:SMARt:CDC <bool>
Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the DC readings at the compression point in the last iteration of a smart sweep. Taking only these DC readings improves measurement speed.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.
<bool> Choose from:

ON or (1) Enable reading DC value at compression point in the last iteration of a smart sweep.

OFF or (0) Disable reading DC value at compression point in the last iteration of a smart sweep.

Examples

SENS:GCS:SMAR:CDC 1
sense:gcsetup:smart:cdc off

Query Syntax
SENS<ch>:GCSetup:SMARt:CDC?

Return Type
Boolean

Default
OFF

SENSe<ch>:GCSetup:SMARt:MITerations <num>

Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the maximum permitted number of iterations which SMART Sweep may utilize to find the desired compression level, to within the specified tolerance.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.
<num> Maximum number of iterations. Choose a value between 1 and 500

Examples

SENS:GCS:SMAR:MIT 5
sense:gcsetup:smart:miterations 3

Query Syntax
SENSe<ch>:GCSetup:SMARt:MITerations?

Return Type
Numeric

Default
20

SENSe<ch>:GCSetup:SMARt:SIterations <bool>
Applicable Models: All with Gain Compression Option

(Read-Write) Set and read enable for showing intermediate results for each iteration of SMART Sweep

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.
<bool> Choose from:

ON or (1) Compression traces are updated after each iteration.
OFF or (0) Compression traces are updated after ALL iterations are complete.

Examples

SENSe<ch>:GCSetup:SMART:SIT 1
sense:gcsetup:smart:siterations off

Query Syntax
SENSe<ch>:GCSetup:SMART:SITerations?

Return Type Boolean

Default OFF

SENSe<ch>:GCSetup:SMART:STIME <num>

Applicable Models: All with Gain Compression Option

(Read-Write) Set and read the amount of time SMART Sweep will dwell at the first point where the input power changes by the Backoff or X level. Applies only to SMART Sweep when Backoff or XY compression methods are selected. Learn more.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.
<num> Settling time in seconds. Choose any positive value.

Examples

SENSe<ch>:GCSetup:SMART:STIM 1
sense:gcsetup:smart:stime .1

Query Syntax
SENSe<ch>:GCSetup:SMART:STIME?

Return Type Numeric

Default 0

SENSe<ch>:GCSetup:SMART:TOLerance <num>
**Applicable Models:** All with Gain Compression Option

(Read-Write) Set and read the acceptable range SMART Sweep will allow for the measured compression level.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Tolerance level in dBm. Choose a value between .01 and 10

**Examples**

```
SENS:GCS:SMAR:TOL .1
sense:gcsetup:smart:tolerance .05
```

**Query Syntax**

```
SENSe<ch>:GCSetup:SMART:TOLerance?
```

**Return Type** Numeric

**Default** .05

**SENSe<ch>:GCSetup:SWEep:FREQuency:POINts <num>**

**Applicable Models:** All with Gain Compression Option

(Read-Write) Set and read the number of data points in each frequency sweep. Learn more

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Frequency points. Do not exceed the max number of data points.

**Examples**

```
SENS:GCS:SWE:FREQ:POIN 201
sense:gcsetup:sweep:frequency:points 101
```

**Query Syntax**

```
SENSe<ch>:GCSetup:SWEep:FREQuency:POINts?
```

**Return Type** Numeric

**Default** 201

**SENSe<ch>:GCSetup:SWEep:POWer:POINts <num>**
**Applicable Models:** All with Gain Compression Option

*(Read-Write)* Set and read the number of data points in each power sweep. Applies ONLY to 2D acquisition modes.

**Parameters**
- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Power points. Do not exceed the max number of data points.

*See Data Points Limit*

**Examples**
```
SENS:GCS:SWE:POW:POIN 50
sense:gcsetup:sweep:power:points 21
```

**Query Syntax**
```
SENSe<ch>:GCSetup:SWEep:POWer:POINts?
```

**Return Type**
Numeric

**Default**
21

---

**SENSe<ch>:GCSetup:SWEep:POWer:SMOoth <bool>**

**Applicable Models:** All with Gain Compression Option

*(Read-Write)* Set and read the state of the power smoothing. This is only available in Gain Compression for Converters (GCX). This is only used when `SENSe:GCSetup:COMPression:PHASE:MODE` is set to PHASE or BOTH.

This enables an optional smoothing procedure of the power sweep.

**Parameters**
- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:

  - **ON** or (1) Enable power smoothing.
  - **OFF** or (0) Disable power smoothing.

**Examples**
```
SENS:GCS:SWE:POW:SMO 1
sense:gcsetup:sweep:power:smooth:off
```

**Query Syntax**
```
SENSe<ch>:GCSetup:SWEep:POWer:SMOoth?
```

**Return Type**
Boolean

**Default**
OFF
SENSe<ch>:GCSetup:SWEep:POWer:SMOoth:APERture <num>

**Applicable Models:** All with Gain Compression Option

*(Read-Write)* Set and read the power smoothing aperture in percent. This is only available in Gain Compression for Converters (GCX). This is only used when SENSE:GCSetup:COMPression:PHASe:MODE is set to PHASe or BOTH. It is only applied when smoothing has been enabled.

This SCPI command configures the smoothing aperture used. The default is 25%.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Power smoothing aperture.

**Examples**

```
SENSe:GCS:SWE:POW:SMO:APER 10
sense:gcsetup:sweep:power:smooth:aperture 10
```

**Query Syntax**

SENSe<ch>:GCSetup:SWEep:POWer:SMOoth:APERture?

**Return Type**

Numeric

**Default**

25
Sense:IF Commands

Controls the DSP filters for use with the VNA X.

| SENSE:IF |
| BANDwidth |
| FILTER |
| FILTER |
| AUTO |
| CMODE |
| ERRors? |
| STAGE |
| COEFFicients |
| COUNT? |
| FREQUENCY |
| CATalog? |
| PARameter |
| TYPE |
| PCATalog? |
| FREQUENCY |
| AUTO |
| VALUE |

- Click on a red keyword to view the command details.
- Synchronizing the Analyzer and Controller
For any of the Filter "Stage" parameters to take effect, SENS:IF:FILT:AUTO must be set to OFF (MANUAL) and mode, and SENS:IF:FILT:CMOD must be set to OFF.

Stage2 settings are ignored when using DSP 5 versions.

Programs that control these settings, or state files that are saved, will yield different results when run or recalled on VNAs with DSP 4 versions versus DSP 5 versions. Learn more about DSP versions.

**Critical Note:** These commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select.

```
SENSe<cnum>:IF:BANDwidth:FILTer <enum>
```

**Applicable Models:** N524xB

(Read-Write) Sets and returns the IF bandwidth filter shape. Learn more.

**Critical Note**

**Parameters**

- `<cnum>` Existing channel number. If unspecified, `<cnum>` is set to 1.
- `<enum>` (String) Filter shape. Chose from:
  - **STANdard**: Legacy filter. This filter is the fastest, but has side lobes that rise to ~-32 dBC.
  - **GAUSSian**: The Gaussian filter takes longer to acquire a point but has no side lobes.

**Examples**

```
SENS:IF:BAND:FILT GAUS
sense2:if:bandwidth:filter standard
```

**Query Syntax**

```
SENSe<cnum>:IF:BANDwidth:FILTer?
```
SENSe<cnm>:IF:FILT:CMOD <bool>

**Return Type**  String

**Default**  STANdard

**Applicable Models:** N524xB

*(Read-Write)* Sets and returns whether the VNA configures the 3-stage digital filter settings or they will be configured manually. When making manual settings, also send `SENS:IF:FILT:CMOD OFF` which routes the IF through the 3-stage filter.

**Critical Note**

**Parameters**

- `<cnm>`  Existing channel number. If unspecified, `<cnm>` is set to 1.
- `<bool>`  (Boolean)

- **ON** (or 1) - Automatic: VNA controls digital filter settings.
- **OFF** (or 0) - Manual: You control digital filter settings using other Sens:IF commands.

**Examples**

SENS:IF:FILT:CMOD 1
sense2:if:filter:auto 0

**Query Syntax**

SENS<cnm>:IF:FILT:CMOD?  

**Return Type**  Boolean

**Default**  ON

SENSe:IF:FILT:CMODe <bool>
Applicable Models: N524xB

(Read-Write) Sets and returns the ADC capture mode modeled as a 2-pole switch in the above diagram. The switch either bypasses or routes the IF through the 3-stage digital filter.

Critical Note

Parameters

<bool> (Boolean)

**ON** (or 1) - The digital filters are bypassed and the raw ADC readings are taken directly. With DSP 4 versions, a maximum of 4096 data points per sweep can be acquired.

With DSP 5 versions, the VNA maximum data points per sweep can be acquired.

Learn more about DSP Versions.

**OFF** (or 0) - The digital filters are used to process IF information. The filters can be configured automatically or manually using SENS:IF:FILT:AUTO.

Examples

SENS:IF:FILT:CMOD 1
sense2:if:filter:cmode 0

Query Syntax

SENSe<cnum>:IF:FILTer:CMODe?

Return Type

Boolean

Default

OFF

SENSe<cnum>:IF:FILTER:ERRors?

Applicable Models: N524xB

(Read-only) Returns the error string associated with the digital filters. The return string has three fields separated by commas: "stage1 status, stage2 status, stage3 status"

Each of these fields can contain one or more of the following error codes:

- **NO ERROR**
- **NUMBER-OF-COEFFICIENTS** - the number of coefficients is excessive for that filter-stage
- **COEFFICIENT VALUE** - one or more coefficients are out of range for that filter-stage
- **SUM-OF-COEFFICIENTS** - the sum of all coefficients is excessive for that filter-stage,
- **FREQUENCY** - the frequency for Stage 1 is out of range (only applies stage1 field),
- **PARAMETER** - one or more parameters are out of range (only applies to stage 3 field)
Critical Note

Parameters

<cnum> Existing channel number. If unspecified, <cnum> is set to 1.

<n> Stage number. Choose 1, 2, or 3.

<coef> Filter coefficients.

For stages 1 and 2, values can be sent as integers or in floating point format. If floating point, the values are truncated to integers. The valid range is 0 to 131071.

For stage 3: An array of floating point values. The valid range is any value within range of a floating point.

Examples

SENS:IF:FILT:ERR?

'example return strings''

NO ERROR, NO ERROR, NO ERROR

indicates no errors,

*SUM-OF-COEFFICIENTS, NO ERROR, NO ERROR

indicates that the sum of all filter coefficients exceed the maximum value for the Stage-1 filter,

*COEFFICIENT *SUM-OF-COEFFICIENTS, NO ERROR, *PARAMETER

indicates a problems with Stage 1 coefficients and a problem with one or more of the parameters associated with the Stage 3 filter.

Return Type

String

Default

Not Applicable

SENS<channel>:IF:FILT:STAG<n>:COEF<coeffs>

Applicable Models: N524xB

(Read-Write) Sets and returns the digital filter coefficients of the specified stage.

Note: Stage2 settings are ignored when using DSP Version 5. Learn more.

Critical Note

Parameters

<cnum> Existing channel number. If unspecified, <cnum> is set to 1.

<n> Stage number. Choose 1, 2, or 3.

<coef> Filter coefficients.

For stages 1 and 2, values can be sent as integers or in floating point format. If floating point, the values are truncated to integers. The valid range is 0 to 131071.

For stage 3: An array of floating point values. The valid range is any value within range of a floating point.

Examples

SENS:IF:FILT:STAG2:COEF 0,0.1,0.7,0.7,0.1

sense2:if:filter:stage3:coefficients

+0.0E+000,+6.4E+001,+2.56E+002

Query Syntax

SENS<channel>:IF:FILT:STAG<n>:COEF<coeffs>?
**Example**

`SENS:IF:FILT:STAG2:COEF?`

**Return Type**

Floating point values

**Default**

Stage dependent

---

`SENS<cnum>:IF:FILTer:STAGe<n>:COUNt? [char]`

**Applicable Models:** N524xB

(Read-only) Returns the number of taps in the digital filter of the specified stage. The filter sample count setting is only used when `SENS:IF:FILTer:AUTO` is set to False (MANUAL).

**Note:** Stage2 settings are ignored when using DSP Version 5. Learn more.

**Critical Note**

**Parameters**

- `<cnum>`: Existing channel number. If unspecified, `<cnum>` is set to 1.
- `<n>`: Stage number. Choose 1, 2, or 3
- `[char]`: Optional parameter. Choose from:
  - **MIN**: returns the minimum number of coefficients for the specified stage. Stage1: 10, Stage2: 1, Stage3: 2
  - **MAX**: returns the maximum number of coefficients for the specified stage. Stage1 & 2: 1024, Stage3: 102400

**Examples**

`SENS:IF:FILT:STAG2:COUN?`

`sense2:if:filter:stag1:count? max`

**Return Type**

Numeric

**Default**

Stage dependent

---

`SENS<cnum>:IF:FILTer:STAGe1:FREQuency <value>`

---

4371
Applicable Models: N524xB

(Read-Write) Sets and returns the Numerically Controlled Oscillator (NCO) frequency. This command is only used when SENSe:IF:FILTER:AUTO is set to False (Manual).

Critical Note

Parameters

- `<cnum>` Existing channel number. If unspecified, `<cnum>` is set to 1.
- `<value>` Stage 1 Frequency. Min value= 0 Hz

With DSP 4 versions, Max value= 15 MHz.

With DSP 5 versions, Max value = 38 MHz.

Or programmatically use the Max and Min queries to determine the range of settable values.

Learn more about DSP versions.

Examples

```
SENS:IF:FILT:STAGE1:FREQ 9e6
Sense2:if:filter:stage1:frequency 9.2e6
```

Query Syntax

```
SENSe<cnum>:IF:FILTER:STAGE1:FREQuency?
'returns the current parameter value

SENSe<cnum>:IF:FILTER:STAGE1:FREQuency? Min
'returns the minimum frequency value.

SENSe<cnum>:IF:FILTER:STAGE1:FREQuency? Max
'returns the maximum frequency value.
```

Return Type

Numeric

Default
Nominal IF Frequency. Learn more

SENSe<cnum>:IF:FILTER:STAGE3:CATalog?
**Applicable Models:** N524xB

*(Read-only)* Returns a list of strings for the currently supported filter types that can be used for the stage 3 filter. This command is only used when `SENS:IF:FIL:T:AUTO` is set to False (Manual). See `SENS:IF:FILT:STAGE3:TYPE` for a list of currently supported filter types.

**Critical Note**

**Parameters**

- **<cnum>** Existing channel number. If unspecified, `<cnum>` is set to 1.

**Examples**

```plaintext
SENS:IF:FILT:STAGE3:CAT?
sense2:if:filter:stage3:catalog?
```

**Return Type**

String array

**Default**

Not Applicable

---

**SENSe<cnum>:IF:FILTER:STAGE3:PARAMeter <p>, <value>**

**Applicable Models:** N524xB

*(Read-Write)* Sets and returns the Stage 3 filter parameters.

Must first select the filter type (`SENS:IF:FILT:STAGE3:TYPE`) before setting these parameters.

Use `SENS:IF:FILT:STAGE3:PCAT?` to return a list of the available parameters for the currently selected filter type.

**Critical Note**

**Parameters**

- **<cnum>** Existing channel number. If unspecified, `<cnum>` is set to 1.
- **<p>** (String) Filter parameter. Case Sensitive. Choose from:
  - "C" - Tap count (Tukey, RECT, PWIN)
  - "P" - Period (PWIN ONLY)
  - "D" - Delay (PWIN ONLY)
  - "W" - Width (PWIN ONLY)
  - "R" - Ramp Count (PWIN ONLY)
  - "M" - Number of times to repeat the user-supplied array for each data point (COEF ONLY)
- **<value>** (Numeric) Parameter Value for the specified stage 3 parameter. Use the query form to return the minimum and maximum values for the specified parameter.
Examples
SENS:IF:FILT:STAGe3:PAR "C", 64
sense2:if:filter:stage3:parameter "d", 0.5E-6

Query Syntax
SENSe<cnum>:IF:FILTer:STAGe3:PARameter? <p>
returns the current parameter value

SENSe<cnum>:IF:FILTer:STAGe3:PARameter? <p>, Min
returns the minimum parameter value.

SENSe<cnum>:IF:FILTer:STAGe3:PARameter? <p>, Max
returns the maximum parameter value.

Examples
SENS:IF:FILT:STAGe3:PAR? "C"
sense2:if:filter:stage3:parameter? "d", min

Return Type
Numeric

Default
RECT: C = 1

PWIN: C=1E6, P=10ms, D=50us, W=50us, R=7

TUKEY: C=1

COEF: M=1

SENSe<cnum>:IF:FILTer:STAGe3:PCATalog?
Applicable Models: N524xB

(Read-only) Returns a list of the available parameters for the currently selected filter type.

Critical Note

Parameters
<cnum> Existing channel number. If unspecified, <cnum> is set to 1.

Examples
SENS:IF:FILT:STAGe3:PCAT?
sense2:if:filter:stage3:pcatalog?

Return Type
String

Default
Not Applicable

SENSe<cnum>:IF:FILTer:STAGe3:TYPE <value>
**Applicable Models:** N524xB

*(Read-Write)* Sets and returns the Stage 3 filter type. This command is only used when `SENSe:IF:FILTer:AUTO` is set to False (Manual).

**Critical Note**

**Parameters**

- `<cnum>` Existing channel number. If unspecified, `<cnum>` is set to 1.
- `<value>` (String) Filter type. Chose from:
  - "RECT" Rectangular Window Filter
  - "TUKEY" Tukey Filter
  - "PWIN" Pulse Window Filter
  - "COEF" User-supplied array

**Examples**

```
SENS:IF:FILT:STAGe3:TYPE RECT
sense2:if:filter:stage3:type pwin
```

**Query Syntax**

`SENSe<cnum>:IF:FILTer:STAGe3:TYPE?`

**Return Type**

String

**Default** "TUKEY"

---

**SENSe<cnum>:IF:FREQuency:AUTO <bool>**

**Applicable Models:** N524xB

*(Read-Write)* Sets and returns the method for specifying the way the IF Frequency is determined.

**Critical Note**

**Parameters**

- `<cnum>` Existing channel number. If unspecified, `<cnum>` is set to 1.
- `<bool>` (Boolean)
  - **ON** (or 1) - Automatic. VNA determines the setting for the IF frequency. The IF frequency is based on many VNA settings, including measurement frequency. Therefore, it is NOT possible to read the IF frequency that is being used.
  - **OFF** (or 0) - Manual. Use `SENS:IF:FREQ` to set the frequency.

**Examples**

```
SENS:IF:FREQ:AUTO 1
sense2:if:frequency:auto 0
```

**Query Syntax**

`SENSe<cnum>:IF:FREQuency:AUTO?`
SENS<cnm>:IF:FREQuency[:VALue] <value>

**Applicable Models:** N524xB

(Read-Write) Sets and returns the IF frequency for ALL receiver paths being used for the specified channel. To set this frequency, **SENS:IF:FREQ:AUTO** must be set to OFF (Manual). Also returns the maximum and minimum allowable frequency settings. [Learn more.]

**Critical Note**

**Parameters**

- `<cnm>`: Existing channel number. If unspecified, `<cnm>` is set to 1.
- `<value>`: (Numeric) Frequency value. Use the Max and Min Queries to determine the range of this setting. (SENS:IF:FREQ? Max)

**Examples**

```
SENS:IF:FREQ 9.1e6
sense2:if:frequency:value 8.9e6
```

**Query Syntax**

SENS<cnm>:IF:FREQuency?

'returns the current frequency setting

SENS<cnm>:IF:FREQuency? Max

'returns the maximum allowable frequency setting

SENS<cnm>:IF:FREQuency? Min

'returns the minimum allowable frequency setting

**Return Type**

- **Numeric**
- **Default**: 9 MHz
## Sense:IMD Commands

Controls an IMD or IMDx measurement configuration.

<table>
<thead>
<tr>
<th>SENSe:IMD:</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWEep:TYPE</td>
</tr>
<tr>
<td>CSO:</td>
</tr>
<tr>
<td>NDPRoducts</td>
</tr>
<tr>
<td>NORMalized:POWer</td>
</tr>
<tr>
<td>OFFSet</td>
</tr>
<tr>
<td>CTB</td>
</tr>
<tr>
<td>NCARriers</td>
</tr>
<tr>
<td>NORMalized:POWer</td>
</tr>
<tr>
<td>OFFSet</td>
</tr>
<tr>
<td>FREQuency</td>
</tr>
<tr>
<td>DFRequency</td>
</tr>
<tr>
<td>[:CW]</td>
</tr>
<tr>
<td>STARt</td>
</tr>
<tr>
<td>STOP</td>
</tr>
<tr>
<td>F1[:CW]</td>
</tr>
<tr>
<td>F2[:CW]</td>
</tr>
<tr>
<td>FCENter</td>
</tr>
<tr>
<td>[:CW]</td>
</tr>
<tr>
<td>STARt</td>
</tr>
<tr>
<td>STOP</td>
</tr>
<tr>
<td>CENTer</td>
</tr>
<tr>
<td>SPAN</td>
</tr>
</tbody>
</table>

HOPRoduct?

HOPRoduct
<table>
<thead>
<tr>
<th>ACTive?</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFBWidth</td>
</tr>
<tr>
<td>MAIN</td>
</tr>
<tr>
<td>IMTone</td>
</tr>
<tr>
<td>NORMalized</td>
</tr>
<tr>
<td>MODE</td>
</tr>
<tr>
<td>PMAP</td>
</tr>
<tr>
<td>INPut</td>
</tr>
<tr>
<td>OUTPut</td>
</tr>
<tr>
<td>SORDer</td>
</tr>
<tr>
<td>ACTive?</td>
</tr>
<tr>
<td>TPOWer</td>
</tr>
<tr>
<td>COUPLE[:STATE]</td>
</tr>
<tr>
<td>EQUALize[:STATE]</td>
</tr>
<tr>
<td>F1</td>
</tr>
<tr>
<td>F2</td>
</tr>
<tr>
<td>F1:START</td>
</tr>
<tr>
<td>F1:STOP</td>
</tr>
<tr>
<td>F2:START</td>
</tr>
<tr>
<td>F2:STOP</td>
</tr>
<tr>
<td>LEVel</td>
</tr>
<tr>
<td>SET</td>
</tr>
</tbody>
</table>

Click on a keyword to view the command details.

**Blue** commands are superseded.

**Other Swept IMD commands**

- **CALCulate:MEASure:DEFine** - creates a Swept IMD measurement.
- **Swept IMD Calibration** - these are supplemental to the **Guided Cal** commands.
- Use std channel commands to set Source and Receiver Attenuation.
- SENS:ROLE:DEVice “RF2” - use an external source for f2.

See Also

- Example Create and Cal an IMD Measurement
- IM Spectrum commands
- Learn about Swept IMD
- Learn about Measurement Class
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

SENSe<cnum>:IMD:SWEep:TYPE <char>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the sweep type for the IMD measurement.

Parameters

- <cnum> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <char> Sweep type. Choose from:

  - **FCENter** - (Center Frequency) Maintaining a constant tone spacing (DeltaF) and tone powers (P1 and P2), the center frequency (FC) is swept from Start to Stop.
  
  - **DFRequency** - (Delta Frequency) The center frequency (FC) is held constant. The tone spacing is increased from StartDeltaF to StopDeltaF.
  
  - **POWER** - The main tone frequencies are specified as either F1 and F2, or as FC and DeltaF. These frequencies are held constant while the power of each tone is stepped from the Start Power to the Stop Power.
  
  - **CW** - The main tone frequencies (F1 and F2) and power levels (P1 and P2) are held constant. Measurements are taken for the specified Number of Points.
  
  - **SEGMent** - Not available for IMDx. Same as FCen ter sweep, except that the center frequencies for the sweep are constructed using the standard segment sweep commands.
  
  - **LOPower** - All frequencies are fixed while the LO power is swept from Start to Stop power.

For each sweep type, use the commands that follow:
FCENTer:

- SENS:IMD:FREQ:FCEN:STAR
- SENS:IMD:FREQ:FCEN:STOP
- SENS:IMD:FREQ:FCEN:CENT
- SENS:IMD:FREQ:FCEN:SPAN
- SENS:IMD:FREQ:DFR:[CW]
- SENS:IMD:TPOW:F1
- SENS:IMD:TPOW:F2

DFRequency

- SENS:IMD:FREQ:DFR:STAR
- SENS:IMD:FREQ:DFR:STOP
- SENS:IMD:FREQ:FCEN:[CW]
- SENS:IMD:TPOW:F1
- SENS:IMD:TPOW:F2

POWer

- SENS:IMD:FREQ:F1:[CW]
- SENS:IMD:FREQ:F2:[CW]
- SENS:IMD:FREQ:FCEN:[CW]
- SENS:IMD:FREQ:DFR:[CW]
- SENS:IMD:TPOW:F1:STAR
- SENS:IMD:TPOW:F1:STOP
- SENS:IMD:TPOW:F2:STAR
- SENS:IMD:TPOW:F2:STOP

CW

- SENS:IMD:FREQ:F1:[CW]
- SENS:IMD:FREQ:F2:[CW]
- SENS:IMD:FREQ:FCEN:[CW]
- SENS:IMD:FREQ:DFR:[CW]
- SENS:IMD:TPOW:F1
- SENS:IMD:TPOW:F2

**SEGMent - Not available for IMDx.**

- SENS:IMD:FREQ:DFR:[CW]
- SENS:IMD:TPOW:F1
- SENS:IMD:TPOW:F2
- Use the standard segment sweep commands to set freq start and stop

**LOPower - IMDx ONLY**

- SENS:MIX:LO:POW:STAR
- SENS:MIX:LO:POW:STOP
- SENS:IMD:FREQ:F1:[CW]
- SENS:IMD:FREQ:F2:[CW]
- SENS:IMD:FREQ:FCEN:[CW]
- SENS:IMD:FREQ:DFR:[CW]
- SENS:IMD:TPOW:F1
- SENS:IMD:TPOW:F2

<table>
<thead>
<tr>
<th>Examples</th>
<th>SENS:IMD:SWEep:TYPE CW</th>
<th>sense2:imd:sweep:type power</th>
</tr>
</thead>
</table>

**Query Syntax**  SENS<cnun>:IMD:SWEep:TYPE?

**Return Type**  Character

**Default**  FCENTer

SENS<cnun>:IMD:CSO:NDPRoducts <num>
**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and returns the “N = number of distortion products” value for the calculation of the CSO parameter. Learn more.

### Parameters
- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>` Number of distortion products.

### Examples
- `SENS:IMD:CSO:NDPR 30`
- `sense2:imd:cso:ndproducts 7`

### Query Syntax
- `SENSe<cnum>:IMD:CSO:NDPRoducts?`

### Return Type
- Numeric

### Default
- 40

---

**SENSe<cnum>:IMD:CSO:NORMalized:POWer <num>**

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and returns the CSO Power for POWER normalization mode. Valid only with measurement parameters: CSO2Lo and CSO2Hi and for Normalization Modes dBm and dBmV. Learn more.

### Parameters
- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>` Power level. The units are determined by `Sens:IMD:Norm:Mode`, which must be set first.

### Examples
- `SENS:IMD:CSO:NORM:POW 0`

### Query Syntax
- `SENSe<cnum>:IMD:CSO:NORMalized:POWer?`

### Return Type
- Numeric

### Default
- 0

---

**SENSe<cnum>:IMD:CSO:OFFSet <num>**
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the offset that is applied to CSO measurements. Valid only with measurement parameters: CSO2Lo and CSO2Hi. Learn more.

Parameters

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>` Offset value in dBm

Examples

- `SENS:IMD:CSO:OFFS 3`
- `sense2:imd:cs:offset 7`

Query Syntax `SENSe<cnum>:IMD:CSO:OFFSet?`

Return Type Numeric

Default 0

SENSe<cnum>:IMD:CTB:NCARriers <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the “N = Total number of carriers” value used in the calculation of the XMOD parameter. Learn more.

Parameters

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>` Number of carriers.

Examples

- `SENS:IMD:CTB:NCAR 10`
- `sense2:imd:ctb:ncarriers 50`

Query Syntax `SENSe<cnum>:IMD:CTB:NCARriers?`

Return Type Numeric

Default 40

SENSe<cnum>:IMD:CTB:NORMalized:POWer <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the CTB Power. Valid only with measurement parameters: CTBLo and CTBHi and for Normalization Modes dBm and dBmV. Learn more.

**Parameters**

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>` Power level. The units are determined by `Sens:IMD:Norm:Mode`, which must be set first.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SENS:IMD:CTB:NORM:POW 0</code></td>
</tr>
<tr>
<td><code>sense2:imd:ctb:normalized:power -5</code></td>
</tr>
</tbody>
</table>

**Query Syntax** `Sense<cnum>:IMD:CTB:NORMALized:POWer?`

**Return Type** Numeric

**Default** 0

---

SENSe<cnum>:IMD:CTB:OFFSet <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the offset that is applied to CTB measurements. Valid only with measurement parameters: CTB, CTBLo, CTBHi, CTBE, CTBELo, and CTBEHi. Learn more.

**Parameters**

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>` Offset value in dBm

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SENS:IMD:CTB:OFFS 3</code></td>
</tr>
<tr>
<td><code>sense2:imd:ctb:offset 7</code></td>
</tr>
</tbody>
</table>

**Query Syntax** `SENSe<cnum>:IMD:CTB:OFFSet?`

**Return Type** Numeric

**Default** 0

---

SENSe<cnum>:IMD:FREQuency:DFRequency[:CW] <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns fixed tone spacing.

Parameters

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>` Tone spacing frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

Examples

```
SENS:IMD:FREQ:DFR 1e6
sense2:imd:frequency:dfrequency:cw 2e7
```

Query Syntax

SENSe<cnum>:IMD:FREQuency:DFRequency[:CW]?

Return Type

Numeric

Default

1 MHz

---

SENSe<cnum>:IMD:FREQuency:DFRequency:STARt <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the starting main tone separation for sweep type = DFRequency (delta frequency).

Parameters

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>` Starting tone separation between F1 and F2 in Hz. Both F1 and F2 tones MUST be within the frequency range of the VNA where:

  \[
  F1 \text{ (start)} = \text{FREQ:FCEN} - \text{DFR:Start} / 2
  \]

  \[
  F2 \text{ (start)} = \text{FREQ:FCEN} + \text{DFR:Start} / 2
  \]

Examples

```
SENS:IMD:FREQ:DFR:STAR 1e6
sense2:imd:frequency:dfrequency:start 2e7
```

Query Syntax

SENSe<cnum>:IMD:FREQuency:DFRequency:STARt?

Return Type

Numeric

Default

1 MHz

---

SENSe<cnum>:IMD:FREQuency:DFRequency:STOP <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the stopping main tone separation for sweep type = DFRequency (delta frequency).

**Parameters**

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>` Stopping tone separation between F1 and F2 in Hz. Both F1 and F2 tones MUST be within the frequency range of the VNA where:

  \[
  F1 \text{ (stop)} = FREQ:FCEN - DFR:Stop / 2
  \]
  \[
  F2 \text{ (stop)} = FREQ:FCEN + DFR:Stop / 2
  \]

**Examples**

```
SENS:IMD:FREQ:DFR:STOP 1e6
sense2:imd:frequency:dfrequency:stop 2e7
```

**Query Syntax**

`SENSe<cnum>:IMD:FREQuency:DFRequency:STOP?`

**Return Type** Numeric

**Default** 10 MHz

---

`SENSe<cnum>:IMD:FREQuency:F1[:CW] <num>`

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the frequency of the F1 tone.

**Parameters**

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>` F1 tone frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

**Examples**

```
SENS:IMD:FREQ:F1 1e9
sense2:imd:frequency:F1:cw 2e7
```

**Query Syntax**

`SENSe<cnum>:IMD:FREQuency:F1[:CW]?`

**Return Type** Numeric

**Default** 0.9995 GHz

---

`SENSe<cnum>:IMD:FREQuency:F2[:CW] <num>`
(Read-Write) Sets and returns the frequency of the F2 tone.

Parameters

- `<cnum>`: Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>`: F2 tone frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

Examples

```
SENS:IMD:FREQ:F2 1e9
sense2:imd:frequency:F2: cw 2e7
```

Query Syntax

SENSe<cnum>:IMD:FREQuency:F2[:CW]?

Return Type: Numeric

Default: 1.0005 GHz

---

SENSe<cnum>:IMD:FREQuency:FCENter[:CW] <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the center frequency of the main tones.

Parameters

- `<cnum>`: Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>`: Tone center frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

Examples

```
SENS:IMD:FREQ:FCEN 1e9
sense2:imd:frequency:fcenter: cw 2e7
```

Query Syntax

SENSe<cnum>:IMD:FREQuency:FCENter[:CW]?

Return Type: Numeric

Default: 1.0 GHz

---

SENSe<cnum>:IMD:FREQuency:FCENter:CENTer <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the sweep center frequency when sweeping the main tones.

Parameters

- `<cnum>`: Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>`: Center frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

Examples

```
SENS:IMD:FREQ:FCEN:CENT 1e9
sense2:imd:frequency:fcenter:fcen 2e7
```

Query Syntax

SENSe<cnum>:IMD:FREQuency:FCENter:CENTer?

Return Type

Numeric

Default

13.255 GHz

SENSe<cnum>:IMD:FREQuency:FCENter:SPAN <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the frequency span when sweeping the main tones.

Parameters

- `<cnum>`: Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>`: Frequency span in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

Examples

```
SENS:IMD:FREQ:FCEN:SPAN 1e9
sense2:imd:frequency:fcenter:span 2e7
```

Query Syntax

SENSe<cnum>:IMD:FREQuency:FCENter:SPAN?

Return Type

Numeric

Default

26.489 GHz

SENSe<cnum>:IMD:FREQuency:FCENter:STARt <num>
(Read-Write) Sets and returns the start frequency when sweeping the main tones.

**Parameters**

- `<cnum>`
  - Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>`
  - Start frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

**Examples**

```
SENS:IMD:FREQ:FCEN:STAR 1e9
sense2:imd:frequency:fcenter:start 2e7
```

**Query Syntax**

SENSe<cnum>:IMD:FREQuency:FCENter:STARt?

**Return Type**

Numeric

**Default**

10.5 MHz

---

SENSe<cnum>:IMD:FREQuency:FCENter:STOP <num>

(Read-Write) Sets and returns the stop frequency when sweeping the main tones.

**Parameters**

- `<cnum>`
  - Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>`
  - Stop frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the VNA.

**Examples**

```
SENS:IMD:FREQ:FCEN:STOP 1e9
sense2:imd:frequency:fcenter:stop 2e9
```

**Query Syntax**

SENSe<cnum>:IMD:FREQuency:FCENter:STOP?

**Return Type**

Numeric

**Default**

26.4995 GHz

---

SENSe:IMD:HOPRoduct?
Applicable Models: N522xB, N524xB

(Read-only) Returns the highest order product that can be measured by SweptIMD.

**Parameters**

None

**Examples**

SENS:IMD:HOPR?

'always returns 9

**Return Type**

Numeric

**Default**

9

SENSe<cnm>:IMD:HOPRduct:ACTive?

(Read-only) Returns the highest order product measured by SweptIMD.

**Parameters**

<cnm> Channel number of the IMD measurement. If unspecified, value is set to 1.

**Examples**

SENS:IMD:HOPR:ACT?

**Return Type**

Numeric

**Default**

Not applicable

SENSe<cnm>:IMD:IFBWidth:MAIN <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the IF Bandwidth for measurement of the main F1 and F2 tones. Learn more about setting IFBW for IMD.

**Parameters**

<cnm> Channel number of the IMD measurement. If unspecified, value is set to 1.

<num> Choose from: 1 | 2 | 3 | 5 | 7 | 10 | 15 | 20 | 30 | 50 | 70 | 100 | 150 | 200 | 300 | 500 | 700 | 1k | 1.5k | 2k | 3k | 5k | 7k | 10k | 15k | 20k | 30k | 50k | 70k | 100k | 150k | 200k | 280k | 360k | 600k

If an invalid number is specified, the analyzer will round up to the closest valid number.

**Examples**

SENS:IMD:IFBW:MAIN 280e3

sense2:imd:ifbwidth:main 150K

**Query Syntax**

SENSe<cnm>:IMD:IFBWidth:MAIN?
SENSe<cnum>:IMD:IFBWidth:IMTone <num>

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and returns the IF Bandwidth for measurement of the intermodulation products. Learn more about setting IFBW for IMD.

**Parameters**

- **<cnum>** Channel number of the IMD measurement. If unspecified, value is set to 1.
- **<num>** Choose from: 1 | 2 | 3 | 5 | 7 | 10 | 15 | 20 | 30 | 50 | 70 | 100 | 150 | 200 | 300 | 500 | 700 | 1k | 1.5k | 2k | 3k | 5k | 7k | 10k | 15k | 20k | 30k | 50k | 70k | 100k | 150k | 200k | 280k | 360k | 600k

If an invalid number is specified, the analyzer will round up to the closest valid number.

**Examples**

```
SENSe:IMD:IFBW:IMT 50
sense2:imd:ifbwidth:imtone 200
```

**Query Syntax**

SENSe<cnum>:IMD:IFBWidth:IMTone?

**Return Type** Numeric

**Default** 1 kHz

SENSe<cnum>:IMD: Normalized:MODE <char>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the method by which CTB and CSO calculations are performed.

Parameters

<cnun> Channel number of the IMD measurement. If unspecified, value is set to 1.
<char> Normalization mode. Choose from:

NONE - the normalized power is not used in calculation

NCARrier - CTB and CSO is corrected by subtracting 10*\log(N/2), where

- \( N = \# \) of carriers for CTB
- \( N = \# \) of distortion products for CSO

DBM - the composited normalized power for CTB or CSO is treated as a dBm value

DBMV - the composited normalized power for CTB or CSO is treated as a dBmV value.

Note: Power values are stored using the currently-set units. Therefore, first set units with this command, then set power values using:


Examples

SENS:IMD:NORM:MODE NCAR
sense2:imd:normalized:mode none

Query Syntax

SENSe<cnun>:IMD:NORMalized:MODE?

Return Type

Character

Default

NCARrier

SENSe<cnun>:IMD:PMAP <input>,<output>
Applicable Models: N522xB, N524xB

(Write-only) Sets the input port and output port of an IMD or IMDx channel. This setting is necessary only when using the limited port mapping feature. Learn more.

Parameters

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<input>` VNA port connected to the DUT Input. Choose from 1 or 3. When input is 3, an external combiner must be used.
- `<output>` VNA port connected to the DUT Output.
  - When input is 1, output must be 2.
  - When input is 3, output must be 4.

Examples

```
SENS:IMD:PMAP 3,4
sense2:imd:pmap 3,4
```

Query Syntax Not Applicable

Default 1,2

SENSe<cnum>:IMD:PMAP:INPut?

Applicable Models: N522xB, N524xB

(Read-only) Returns the VNA test port to be connected to the DUT input for an IMD or IMDx channel. Set the VNA port to DUT mapping using SENS:IMD:PMAP

Parameters

- `<cnum>` IMD channel number. If unspecified, value is set to 1.

Examples

```
SENS:IMD:PMAP:INP?
sense2:imd:pmap:input?
```

Default 1

SENSe<cnum>:IMD:PMAP:OUTPut?
Applicable Models: N522xB, N524xB

(Read-only) Returns the VNA test port to be connected to the DUT output for an IMD or IMDx channel. Set the VNA port to DUT mapping using SENS:IMD:PMap

**Parameters**

- `<cnum>` IM3 channel number. If unspecified, value is set to 1.

**Examples**

```
SENS:IMD:PMap:OutP?
sense2:imd:pmap:output?
```

**Default** 2

**SENSe<cnum>:IMD:SORDer:ACTive?**

(Read-only) Returns the state of whether or not 2nd order parameters are measured. A "1" or "ON" indicates 2nd order parameters are measured.

**Parameters**

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.

**Examples**

```
SENS:IMD:SORD:ACT?
```

**Return Type** Boolean

**Default** Not applicable

**SENSe<cnum>:IMD:TPOWer:COUPle[:STATe] <bool>**

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the state of power coupling for F1 and F2. Learn more about tone power.

**Parameters**

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<bool>` Tone power level coupling state. Choose from:

  **ON (or 1)** - F1 and F2 power is coupled.

  **OFF (or 0)** - F1 and F2 power is NOT coupled. Set power levels individually.

**Examples**

```
SENS:IMD:TPower:COUPLE 0
sense2:imd:tpower:couple:state ON
```

**Query Syntax**

```
SENSe<cnum>:IMD:TPower:COUPLE[:STATe]?
```
SENSe<cnump>:IMD:TPOWer:EQUalize:STATe <bool> - **Superseded**

**Applicable Models:** N522xB, N524xB

*(Read-Write) This command is replaced with SENS:IMD:TPOW:LEV*

Sets and returns the state of Equal Tone Power setting. Learn more about tone power.

**Parameters**

- `<cnump>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<bool>` Equal Tone Power state. Choose from:
  - **ON (or 1)** - Equalize f1 and f2 power.
  - **OFF (or 0)** - Do NOT equalize f1 and f2 power. Use source power cal.

**Examples**

```
SENS:IMD:TPOW:EQU 0
sense2:imd:tpower:equalize:state ON
```

**Query Syntax**

`SENSe<cnump>:IMD:TPOWer:EQUalize:STATe?`

**Return Type** Boolean

**Default** OFF

SENSe<cnump>:IMD:TPOWer:F1 <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the power level of the F1 tone. When SENS:IMD:TPOW:COUP ON (tone power is coupled), setting either F1 or F2 power sets both. Learn more about tone power.

Parameters

- <cnum> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Tone power level in dBm. Choose a value between +30 dBm and -30 dBm.

Examples

```
SENS:IMD:TPOW:F1 0
sense2:imd:tpower:F1 -10
```

Query Syntax

SENS<cnum>:IMD:TPOWer:F1?

Return Type

Numeric

Default

-24 dBm

SENSe<cnum>:IMD:TPOWe:F2 <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the power level of the F2 tone. When SENS:IMD:TPOW:COUP ON (tone power is coupled), setting either F1 or F2 power sets both. Learn more about tone power.

Parameters

- <cnum> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Tone power level in dBm. Choose a value between +30 dBm and -30 dBm.

Examples

```
SENS:IMD:TPOW:F2 0
sense2:imd:tpower:F2 -10
```

Query Syntax

SENS<cnum>:IMD:TPOWer:F2?

Return Type

Numeric

Default

-24 dBm

SENSe<cnum>:IMD:TPOWe:F1:STARt <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the start power level of the F1 tone. Learn more about tone power.

Parameters

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>` Start power in dBm. Choose a value between +30 dBm and -30 dBm.

Examples

```
SENS:IMD:TPOW:F1:STAR 0
sense2:imd:tpower:F1:start -10
```

Query Syntax

SENSe<cnun>:IMD:TPOwer:F1:STARt?

Return Type Numeric

Default -24 dBm

SENSe<cnun>:IMD:TPOwer:F1:STOP <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the stop power level of the F1 tone. Learn more about tone power.

Parameters

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>` Stop power in dBm. Choose a value between +30 dBm and -30 dBm.

Examples

```
SENS:IMD:TPOW:F1:STOP 0
sense2:imd:tpower:F1:stop 10
```

Query Syntax

SENSe<cnun>:IMD:TPOwer:F1:STOP?

Return Type Numeric

Default -10 dBm

SENSe<cnun>:IMD:TPOwer:F2:STARt <num>
**Applicable Models:** N522xB, N524xB

(Read-Write) Sets and returns the start power level of the F2 tone. Learn more about tone power.

**Parameters**

- `<cnum>`: Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>`: Start power in dBm. Choose a value between +30 dBm and -30 dBm.

**Examples**

```
SENS:IMD:TPOW:F2:STAR 0
sense2:imd:tpower:F2:start -10
```

**Query Syntax**

`SENSe<cnum>:IMD:TPOWer:F2:STARt?`

**Return Type**

Numeric

**Default**

-24 dBm

---

**SENSe<cnum>:IMD:TPOWer:F2:STOP <num>**

**Applicable Models:** N522xB, N524xB

(Read-Write) Sets and returns the stop power level of the F2 tone. Learn more about tone power.

**Parameters**

- `<cnum>`: Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<num>`: Stop power in dBm. Choose a value between +30 dBm and -30 dBm.

**Examples**

```
SENS:IMD:TPOW:F2:STOP 0
sense2:imd:tpower:F2:stop 10
```

**Query Syntax**

`SENSe<cnum>:IMD:TPOWer:F2:STOP?`

**Return Type**

Numeric

**Default**

-10 dBm

---

`SENSe<cnum>:IMD:TPOWer:LEVel <char>`
**Applicable Models:** N522xB, N524xB

*(Read-Write)* This command replaces SENS:IMD:TPOW:EQU and SENS:IMD:TPOW:SET

Sets and returns the tone power leveling mode.

**Parameters**

<cnомер>  Channel number of the IMD measurement. If unspecified, value is set to 1.
<char>  Choose from:

- **NONE** - *(Set Input Power)* The specified f1 and f2 power levels are set at the DUT input.

- **INPut** - *(Set Input Power, receiver leveling)* The specified f1 and f2 power levels are set at the DUT input using receiver leveling at the input reference receiver.

- **EQUal** - *(Set Input Power, equal tones at output)* The specified f1 and f2 power levels are set at the DUT input and a measurement is made at the output.

- **OUTPut** - *(Set Output Power, receiver leveling)* The specified f1 and f2 power levels are set at the DUT output.

Learn more about these choices.

**Examples**

```plaintext
SENS:IMD:TPOW:LEV INP
sense2:imd:tpower:level output
```

**Query Syntax**

`SENSe<cnомер>:IMD:TPOWer:LEV?`

**Return Type**

Character

**Default**

NONE

**SENSe<cnомер>:IMD:TPOWer:SET <char>** - *Superseded*
Applicable Models: N522xB, N524xB

(Read-Write) This command is replaced with SENS:IMD:TPOW:LEV

Sets and returns whether tone power is specified at the DUT input or output.

**Parameters**

- `<cnum>` Channel number of the IMD measurement. If unspecified, value is set to 1.
- `<char>` Choose from:

  **INPUT** - Specified power level is set at the DUT input.
  
  **OUTPUT** - Specified power level is set at the DUT output.

**Examples**

```
SENS:IMD:TPOW:SET INPUT
sense2:imd:tpower:set output
```

**Query Syntax** SENSE<cnum>:IMD:TPOWer:SET?

**Return Type** Character

**Default** INPUT
Controls IM Spectrum measurement configuration.

<table>
<thead>
<tr>
<th>SENSe:IMS:</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMAP</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>RBW</td>
</tr>
<tr>
<td>RESPonse</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>STIMulus</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>SWEEP:</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>TPOWer</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
SENSe<cnum>:IMS:PMAP <input>,<output>

Applicable Models: N522xB, N524xB

(Write-only) Sets the input port and output port of an IMS or IMSx channel. This setting is necessary only when using the limited port mapping feature. Learn more.

Parameters

- `<cnum>` Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- `<input>` VNA port connected to the DUT Input. Choose from 1 or 3. When input is 3, an external combiner must be used.
- `<output>` VNA port connected to the DUT Output.
  - When input is 1, output must be 2.
  - When input is 3, output must be 4.

Examples

- `SENSe:IMS:PMAP 3,4`
- `sense2:ims:pmap 3,4`
**Query Syntax**
Not Applicable

**Default**
1,2

SENS<e<cnum>:IMS:PMAP:INPut?

**Applicable Models:** N522xB, N524xB

*(Read-only)* Returns the VNA test port to be connected to the DUT input for an IMS or IMSx channel. Set the VNA port to DUT mapping using SENS:IMS:PMAP

**Parameters**

<e<cnum> IMS channel number. If unspecified, value is set to 1.

**Examples**

- SENS:IMS:PMAP:INP?
- sense2:ims:pmap:input?

**Default**
1

SENS<e<cnum>:IMS:PMAP:OUTPut?

**Applicable Models:** N522xB, N524xB

*(Read-only)* Returns the VNA test port to be connected to the DUT output for an IMS or IMSx channel. Set the VNA port to DUT mapping using SENS:IMS:PMAP

**Parameters**

<e<cnum> IMS channel number. If unspecified, value is set to 1.

**Examples**

- SENS:IMS:PMAP:OUTP?
- sense2:ims:pmap:output?

**Default**
2

SENS<e<cnum>:IMS:RBW <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the Resolution Bandwidth for the IM Spectrum measurement.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;num&gt;</td>
<td>Resolution BW in Hz. Choose from: 60k</td>
</tr>
</tbody>
</table>

If an invalid number is specified, the VNA will round up to the closest valid number.

Examples

```
SENS:IMS:RBW 600e3
sense2:ims:rbw 1MHz
```

Query Syntax

SENSe<cnum>:IMS:RBW?

Return Type Numeric

Default 600 kHz

SENSe<cnum>:IMS:RESPonse:STARt <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the receiver Start frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when Sweep Type = Linear. Otherwise, this setting is ignored.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;num&gt;</td>
<td>Start frequency in Hz. Choose a frequency within the range of the VNA.</td>
</tr>
</tbody>
</table>

Examples

```
SENS:IMS:RESP:STAR 1e9
sense2:ims:response:start 100e6
```

Query Syntax

SENSe<cnum>:IMS:RESPonse:STARt?

Return Type Numeric

Default 950 MHz
SENSe<cnum>:IMS:RESPonse:STOP <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the receiver Stop frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when Sweep Type = Linear. Otherwise, this setting is ignored.

Parameters

- <cnum> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <num> Stop frequency in Hz. Choose a frequency within the range of the VNA.

Examples

```
SENS:IMS:RESP:STOP 26e9
sense2:ims:response:stop 100e6
```

Query Syntax
SENSe<cnum>:IMS:RESPonse:STOP?

Return Type Numeric

Default 1.05 MHz

SENSe<cnum>:IMS:RESPonse:CENTer <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the receiver Center frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when Sweep Type = Linear. Otherwise, this setting is ignored.

Parameters

- <cnum> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <num> Center frequency in Hz. Choose a frequency within the range of the VNA.

Examples

```
SENS:IMS:RESP:CENT 26e9
sense2:ims:response:center 100e6
```

Query Syntax
SENSe<cnum>:IMS:RESPonse:CENTer?

Return Type Numeric

Default 1.0 GHz

SENSe<cnum>:IMS:RESPonse:SPAN <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the Span of receiver frequencies for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when Sweep Type = Linear. Otherwise, this setting is ignored.

Parameters

- `<cnum>`: Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- `<num>`: Frequency span in Hz. All receiver frequencies should be within the range of the VNA.

Examples

```
SENS:IMS:RESP:SPAN 10e9
sense2:ims:response:span 100e6
```

Query Syntax

```
SENSe<cnum>:IMS:RESPonse:SPAN?
```

Return Type: Numeric

Default: 100 MHz

---

SENSe<cnum>:IMS:STIMulus:DFRequency <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the DeltaF (tone spacing) for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled. Otherwise, this setting is ignored.

Parameters

- `<cnum>`: Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- `<num>`: Delta frequency in Hz. All stimulus settings MUST be within the frequency range of the VNA.

Examples

```
SENS:IMS:STIM:DFR 1e6
sense2:ims:stimulus:dfrequency 100e6
```

Query Syntax

```
SENSe<cnum>:IMS:STIMulus:DFRequency?
```

Return Type: Numeric

Default: 10 MHz

---

SENSe<cnum>:IMS:STIMulus:FCENter <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the Center frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled. Otherwise, this setting is ignored.

Parameters

- `<cnum>`  Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- `<num>`  Center frequency in Hz. All stimulus settings MUST be within the frequency range of the VNA.

Examples

```
SENS:IMS:STIM:FCEN 1e6
sense2:ims:stimulus:fcenter 100e6
```

Query Syntax  SENSe<cnum>:IMS:STIMulus:FCENter?

Return Type  Numeric

Default  1.0 GHz

SENSe<cnum>:IMS:STIMulus:F1FRequency <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the F1 frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled. Otherwise, this setting is ignored.

Parameters

- `<cnum>`  Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- `<num>`  F1 frequency in Hz. All stimulus settings MUST be within the frequency range of the VNA.

Examples

```
SENS:IMS:STIM:F1FR 1e6
sense2:ims:stimulus:f1frequency 100e6
```

Query Syntax  SENSe<cnum>:IMS:STIMulus:F1FRequency?

Return Type  Numeric

Default  995 MHz

SENSe<cnum>:IMS:STIMulus:F2FRequency <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the F2 frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled. Otherwise, this setting is ignored.

Parameters

- `<cnum>` Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- `<num>` F2 frequency in Hz. All stimulus settings MUST be within the frequency range of the VNA.

Examples

```
SENS:IMS:STIM:F2FR 1e6
sense2:ims:stimulus:f2frequency 100e6
```

Query Syntax

```
SENSe<cnum>:IMS:SWEep:TYPE <char>
```

Return Type

Numeric

Default

1.005 MHz

SENSe<cnum>:IMS:SWEep:TYPE <char>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the method in which the spectrum of signals to view are specified. When Tracking is enabled the frequency of the main tones (f1 and f2) are always determined by the Swept IMD channel.

Parameters

- `<cnum>` Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- `<char>` IM Spectrum sweep type. Choose from:
  
  - **LINear** When Tracking is enabled, allows tuning the Response Settings (receiver) to any values within the frequency range of the VNA. When Tracking is NOT enabled also allows setting the Stimulus (sources) to any values within the frequency range or the VNA.
  
  - **SECond** The receiver is tuned to view the 2nd order products (f2-f1 and f1+f2) of the main tones that are currently specified in Stimulus Settings. When Tracking is enabled, the main tones are specified in the Swept IMD channel.
  
  - **THIRd** The receiver is tuned to view the 3rd order products (2f1 -f2 and 2f2-f1) of the main tones that are currently specified in Stimulus Settings. When Tracking is enabled, the main tones are specified in the Swept IMD channel.
The frequency range is set to $N \times \Delta F$. This algorithm will NOT tune the receivers to view the EVEN order products.

### Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:IMS:SWEep:TYPE LIN</td>
<td></td>
</tr>
<tr>
<td>sense2:ims:sweep:type nth</td>
<td></td>
</tr>
</tbody>
</table>

### Query Syntax

SENSe<cnum>:IMS:SWEep:TYPE? | Character | Default: NTH

### Return Type

SENSe<cnum>:IMS:SWEep:ORDer <num>

**Applicable Models:** N522xB, N524xB

(Read-Write) Sets and returns the order number of signals to view when SENS:IMS:SWE:TYPE NTH is specified.

#### Parameters

- **<cnum>** Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- **<num>** Order number of IM products to view. The frequency range is set to $N$ (this number) x DeltaF (set with SENS:IMS:STIM:DFR).

### Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:IMS:SWEep:ORD 5</td>
<td></td>
</tr>
<tr>
<td>sense2:ims:sweep:order 12</td>
<td></td>
</tr>
</tbody>
</table>

### Query Syntax

SENSe<cnum>:IMS:SWEep:ORDer? | Numeric | Default: 9

### Return Type

SENSe<cnum>:IMS:TPOWer:COUPle[:STATe] <bool>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the state of power coupling for F1 and F2.

Parameters

- `<cnum>` Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- `<bool>` Tone power level coupling. Choose from:
  - **ON (or 1)** - F1 and F2 power is coupled.
  - **OFF (or 0)** - F1 and F2 power is NOT coupled. Set power levels individually.

Examples

```plaintext
SENS:IMS:TPOW:COUP 0
sense2:ims:tpower:couple:state ON
```

Query Syntax

```
SENSe<cnum>:IMS:TPOWer:COUPle[:STATe]?
```

Return Type

Boolean

Default

ON

SENS<e>:IMS:TPOWer:EQUalize:STATe <bool> - Superseded

Applicable Models: N522xB, N524xB

(Read-Write) This command is replaced with SENS:IMS:TPOW:LEV

Sets and returns the state of Equal Tone Power setting. Learn more about tone power.

Parameters

- `<cnum>` Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- `<bool>` Equal Tone Power state. Choose from:
  - **ON (or 1)** - Equalize f1 and f2 power.
  - **OFF (or 0)** - Do NOT equalize f1 and f2 power. Use source power cal.

Examples

```plaintext
SENS:IMS:TPOW:EQU 0
sense2:ims:tpower:equalize:state ON
```

Query Syntax

```
SENSe<cnum>:IMS:TPOWer:EQUalize:STATe?
```

Return Type

Boolean
SENSe<cnum>:IMS:STIMulus:TPOWer:F1 <value>

**Applicable Models:** N522xB, N524xB

**(Read-Write)** Sets and returns the power level of the F1 tone. When SENS:IMS:TPOW:COUP ON (tone power is coupled), setting either F1 or F2 power sets both. This setting is ignored if SENS:IMS:TRAC:STAT is enabled.

**Parameters**
- `<cnum>`: Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- `<value>`: F1 tone power level in dBm. Choose a value between +30 dBm and -30 dBm.

**Examples**
- SENS:IMS:STIM:TPOW:F1 -10
- sense2:ims:stimulus:tpower:f1 0

**Query Syntax**
SENSe<cnum>:IMS:STIMulus:TPOWer:F1?

**Return Type**
Numeric

**Default**
-20

SENSe<cnum>:IMS:STIMulus:TPOWer:F2 <value>

**Applicable Models:** N522xB, N524xB

**(Read-Write)** Sets and returns the power level of the F2 tone. When SENS:IMS:TPOW:COUP ON (tone power is coupled), setting either F1 or F2 power sets both. This setting is ignored if SENS:IMS:TRAC:STAT is enabled.

**Parameters**
- `<cnum>`: Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- `<value>`: F2 tone power level in dBm. Choose a value between +30 dBm and -30 dBm.

**Examples**
- SENS:IMS:STIM:TPOW:F2 -10
- sense2:ims:stimulus:tpower:f2 0

**Query Syntax**
SENSe<cnum>:IMS:STIMulus:TPOWer:F2?

**Return Type**
Numeric

**Default**
-20
SENSe<nun>:IMS:TPower:LEVel <char>

Applicable Models: N522xB, N524xB

(Read-Write) This command replaces SENS:IMS:TPOW:EQU and SENS:IMS:TPOW:SET

Sets and returns the tone power leveling mode.

Parameters

<cnun>  Channel number of the IMD measurement. If unspecified, value is set to 1.
<char>  Choose from:

NONE - (Set Input Power) The specified f1 and f2 power levels are set at the DUT input.

INPUT - (Set Input Power, receiver leveling) The specified f1 and f2 power levels are set at the DUT input using receiver leveling at the input reference receiver.

EQUAL - (Set Input Power, equal tones at output) The specified f1 and f2 power levels are set at the DUT input and a measurement is made at the output.

OUTPUT - (Set Output Power, receiver leveling) The specified f1 and f2 power levels are set at the DUT output.

Learn more about these choices.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:IMS:TPow:LEV INPUT</td>
<td>SET Input Power</td>
</tr>
<tr>
<td>sense2:ims:tpower:level output</td>
<td>SET Output Power</td>
</tr>
</tbody>
</table>

Query Syntax  SENSe<nun>:IMS:TPOWer:LEV?
Return Type  Character
Default  NONE

SENSe<nun>:IMS:TPower:SET <char> - Superseded
Applicable Models: N522xB, N524xB

(Read-Write) This command is replaced with `SENS:IMS:TPOW:LEV`

Sets and returns whether tone power is specified at the DUT input or output.

Parameters

- `<cnum>` Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- `<char>` Choose from:
  - `INPUT` - Specified power level is set at the DUT input.
  - `OUTPUT` - Specified power level is set at the DUT output.

Examples

```
SENS:IMS:TPOW:SET INPUT
sense2:ims:tpower:set output
```

Query Syntax

`SENSe<cnum>:IMS:TPower:SET?`

Return Type

Character

Default

`INPUT`

---

`SENSe<cnum>:IMS:TRACking:CHANnel <num>`

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the IMD channel number to which the IM Spectrum channel is coupled.

Parameters

- `<cnum>` IMS channel. If unspecified, value is set to 1.
- `<num>` Existing IMD channel to which frequency and power settings are coupled.

Examples

```
SENS:IMS:TRAC:CHAN 2
sense2:ims:tracking:channel 1
```

Query Syntax

`SENSe<cnum>:IMS:TRACking:CHANnel?`

Return Type

Numeric

Default

First IMD channel

---

`SENSe<cnum>:IMS:TRACking:MSENable <bool>`
**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and returns the step sweep mode for the IM Spectrum channel.

**Parameters**

- `<cnum>`: IMS channel. If unspecified, value is set to 1.
- `<bool>`: Choose from:
  - **OFF (or 0)** - Automatic Step
  - **ON (or 1)** - Manual Step

**Examples**

```plaintext
SENS:IMS:TRAC:MSEN 1
sense2:ims:tracking:msenable 0
```

**Query Syntax**

`SENSe<cnum>:IMS:TRACking:MSENable?`

**Return Type**

Boolean

**Default**

0 - Automatic

---

**SENSe<cnum>:IMS:TRACking:SINDex <num>**

**Applicable Models:** N522xB, N524xB

*(Read-Write)* When `SENS:IMS:TRAC:MSEN = Manual`, sets and returns the data point number at which a sweep is performed.

**Parameters**

- `<cnum>`: IMS channel. If unspecified, value is set to 1.
- `<num>`: Step index. Choose from 1 to the specified number of points.

**Examples**

```plaintext
SENS:IMS:TRAC:SINDex 201
sense2:ims:tracking:sindex 1
```

**Query Syntax**

`SENSe<cnum>:IMS:TRACking:SINDex?`

**Return Type**

Numeric

**Default**

1

---

**SENSe<cnum>:IMS:TRACking:STATe <bool>**
Applicable Models: N522xB, N524xB

(Read-Write) When an IMD channel exists, allows the IM Spectrum frequency and power setting to track (couple with) the IMD channel settings.

Parameters

- `<cnum>` IMS channel. If unspecified, value is set to 1.
- `<bool>` Tracking state. Choose from:
  - **ON (or 1)** - IM Spectrum frequency and power settings track the IMD channel settings.
  - **OFF (or 0)** - IM Spectrum frequency and power settings are specified in the IMS channel.

Examples

```
SENS:IMS:TRAC:STAT 0
sense2:ims:tracking:state ON
```

Query Syntax

```
SENSe<cnum>:IMS:TRACking:STATe?
```

Return Type

Boolean

Default

OFF
Sense:Mixer Commands

Performs Mixer setup and configuration.

SENSe:MIXer:

  APPLy
  AVOidspurs
  CALCulate
  DISCard

ELO - More Commands

IF:FREQ:

  SIDeband
  STARt
  STOP

INPut:FREQ:

  DENominator
  FIXed
  MODE
  NUMerator
  STARt
  STOP

INPut:POWer

  STARt
  STOP
  USENominal

LO:FREQ:

  DENominator
  FIXed
  ILTI
  MODE
  NUMerator
  STARt
  STOP

LO:NAME

LOAD

NORMalize:POINT

OUTPut:FREQ:

  FIXed
  MODE
  SIDeband
Click on a keyword to view the command details.

See Also

- Example Programs
- Learn about the Frequency Converter Application
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**Note:** If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

---

**SENSe<ch>:MIXer:APPLy**

**Applicable Models:** All

*(Write only)* Applies the mixer setup settings and turns the channel ON. (Performs the same function as the Apply button on the mixer setup dialog box).

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1

**Examples**

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:MIX:APPL</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**Query Syntax** Not Applicable

**Default** Not Applicable
SENSe<ch>:MIxED:AVOidspurs <bool>

Applicable Models: N522xB, N523xB, N524xB

(Read Write) Sets and returns the state of the avoid spurs feature. Learn more about avoid spurs.

Parameters
   <ch> Any existing channel number. If unspecified, value is set to 1
   <bool> Avoid spurs state. Choose from
           0 - Avoid spurs OFF
           1 - Avoid spurs ON

Examples
   SENS:MIX:AVO
   sense2:mixer:avoidspurs 1

Query Syntax
   SENSe<ch>:MIX:AVOidspurs?

Return Type
   Boolean

   Default
   0 (OFF)

SENSe<ch>:MIxED:CALCulate <char>

Applicable Models: All

(Write only) Calculates the Input, IF, or Output frequencies of the mixer setup and updates the channel settings.

Note: The target mode must be swept. This command does not allow calculation of fixed values.

Parameters
   <ch> Any existing channel number. If unspecified, value is set to 1
   <char> Mixer port to be calculated. Choose from:
<table>
<thead>
<tr>
<th>&lt;char&gt;</th>
<th>1st or only stage requires:</th>
<th>In addition, 2nd stage requires:</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPut</td>
<td>• Output Start/Stop/Fixed frequencies</td>
<td>• IF Start/Stop/Fixed frequencies</td>
</tr>
<tr>
<td></td>
<td>• LO Start/Stop/Fixed frequencies</td>
<td>• 2nd Start/Stop/Fixed frequencies</td>
</tr>
<tr>
<td></td>
<td>• Output sideband (High or Low)</td>
<td>• IF sideband (High or Low)</td>
</tr>
<tr>
<td>BOTH</td>
<td>NA</td>
<td>• IF Start/Stop/Fixed frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Both Start/Stop/Fixed frequencies</td>
</tr>
<tr>
<td>OUTPut</td>
<td>• Input Start/Stop/Fixed frequencies</td>
<td>• IF Start/Stop/Fixed frequencies</td>
</tr>
<tr>
<td></td>
<td>• LO Start/Stop/Fixed frequencies</td>
<td>• 2nd Start/Stop/Fixed frequencies</td>
</tr>
<tr>
<td></td>
<td>• Output sideband (High or Low)</td>
<td>• IF sideband (High or Low)</td>
</tr>
<tr>
<td>LO_1</td>
<td>• Input Start/Stop/Fixed frequencies</td>
<td>• IF Start/Stop/Fixed frequencies</td>
</tr>
<tr>
<td></td>
<td>• Output Start/Stop/Fixed frequencies</td>
<td>• 2nd Start/Stop/Fixed frequencies</td>
</tr>
<tr>
<td></td>
<td>• Output sideband (High or Low)</td>
<td>• IF sideband (High or Low)</td>
</tr>
<tr>
<td>LO_2</td>
<td>NA</td>
<td>• Input Start/Stop/Fixed frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1st LO Start/Stop/Fixed frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Output Start/Stop/Fixed frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IF sideband (High or Low)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Output sideband (High or Low)</td>
</tr>
</tbody>
</table>

**Examples**

SENS:MIX:CALC Output
SENSe<ch>:MIXer:DISCard

Applicable Models: All

(Write only) Cancels changes that have been made to the Converter setup and reverts to the previously-saved setup. Same as the Cancel button on the mixer setup dialog box.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

Examples 

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:MIXer:IF:FREQuency:SIDeband <char>

Applicable Models: All

(Read-Write) When two LO stages are used, sets or returns whether to select the sum or difference for the IF1 product. (Input + or - LO1 = IF1)

- This setting corresponds to the buttons on LO1 on the Mixer setup dialog
- This setting is ignored when ONE LO stage is selected.
- Also set SENS:MIX:OUTP:FREQ:SID to LOW or HIGH to determine the output frequency of the mixer.

See Note

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<char> Sideband value. Choose from

LOW - Difference (-)

HIGH - Sum (+)

Examples

SENS:MIX:IF:FREQ:SID LOW
SENSe2:MIXer:IF:FREQ:SIDeband HIGH
<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>SENSe&lt;ch&gt;:MIXer:IF:FREQuency:SIDeband?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Character</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>LOW</td>
</tr>
</tbody>
</table>

SENSe<ch>:MIXer:IF:FREQuency:STARt <num>

**Applicable Models:** All

*(Read-Write)* Sets or returns the IF start frequency value of the mixer. **See Note**

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<num>` IF Start Frequency value

**Examples**

<table>
<thead>
<tr>
<th>SENSe:MIX:IF:FREQ:STAR 1e9</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe2:MIXer:IF:FREQ:STARt 1000000000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>SENSe&lt;ch&gt;:MIXer:IF:FREQuency:STARt?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

SENSe<ch>:MIXer:IF:FREQuency:STOP <num>

**Applicable Models:** All

*(Read-Write)* Sets or returns the stop frequency value of the mixer IF frequency. **See Note**

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<num>` IF Stop Frequency value

**Examples**

<table>
<thead>
<tr>
<th>SENSe:MIX:IF:FREQ:STOP 2e9</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe2:MIXer:IF:FREQ:STOP 2000000000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>SENSe&lt;ch&gt;:MIXer:IF:FREQuency:STOP?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

SENSe<ch>:MIXer:INPut:FREQuency:DENominator <value>
Applicable Models: All

(Read-Write) Sets or returns the denominator value of the Input Fractional Multiplier. See Note

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Input denominator value.

Examples

- SENS:MIX:INP:FREQ:DEN 5
- SENS2:MIXer:INPut:FREQ:DENominator 4

Query Syntax

SENSe<ch>:MIXer:INPut:FREQuency:DENominator?

Return Type

- Numeric

Default

- Not Applicable

SENSe<ch>:MIXer:INPut:FREQuency:FIXed<value>

Applicable Models: All

(Read-Write) Sets or returns the fixed frequency of the input. See Note

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Input frequency.

Examples

- SENS:MIX:INPut:FREQ:FIXed 1e9
- SENS2:MIXer:INPut:FREQ:FIXed 1000000000

Query Syntax

SENSe<ch>:MIXer:INPut:FREQuency:FIXed?

Return Type

- Numeric

Default

- Not Applicable

SENSe<ch>:MIXer:INPut:FREQuency:MODE <char>
Applicable Models: All

(Read-Write) Sets or returns the Input sweep mode.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<char>` Input sweep mode. Choose either fixed or swept.

Examples

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>

Query Syntax

SENSe<ch>:MIXer:INPut:FREQuency:MODE?

Return Type

Character

Default

Fixed

---

SENSe<ch>:MIXer:INPut:FREQuency:NUMerator <value>

Applicable Models: All

(Read-Write) Sets or returns the numerator value of the Input Fractional Multiplier. See Note

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Input numerator value.

Examples

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:MIX:INP:FREQ:NUM 3</td>
<td>SENS2:MIXer:INP:FREQ:NUMerator 1</td>
</tr>
</tbody>
</table>

Query Syntax

SENSe<ch>:MIXer:INPut:FREQ:NUMerator?

Return Type

Numeric

Default

Not Applicable

---

SENSe<ch>:MIXer:INPut:FREQuency:STARt <value>
Applicable Models: All

(Read-Write) Sets or returns the Input start frequency value of the mixer. See Note

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.
<value> Input Start frequency

Examples

SENS:MIX:INP:FREQ:STAR 1e9
SENSe2:MIXer:INPut:FREQ:STARt 1000000000

Query Syntax

SENSe<ch>:MIXer:INPut:FREQ:START?

Return Type

Numeric

Default

Not Applicable

SENS<ch>:MIXer:INPut:FREQ:STOP <value>

Applicable Models: All

(Read-Write) Sets or returns the Input stop frequency value of the mixer. See Note

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.
<value> Input stop frequency

Examples

SENS:MIX:INP:FREQ:STOP 2e9
SENSe2:MIXer:INPut:FREQ:STOP 2000000000

Query Syntax

SENSe<ch>:MIXer:INPut:FREQ:STOP?

Return Type

Numeric

Default

Not Applicable

SENS<ch>:MIXer:INPut:POWer <value>
Applicable Models: All

(Read-Write) Sets or returns the value of the Input Power.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<value> Input power in dBm.

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:MIX:INP:POW 9</td>
</tr>
<tr>
<td>SENSE2:MIXer:INPut:POWer 5</td>
</tr>
</tbody>
</table>

Query Syntax SENSE<ch>:MIXer:INPut:POWer?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:INPut:POWer:STARt <value>

Applicable Models: All

(Read-Write) Sets the input start power for a power sweep in a mixer channel like SMC. The value is only used when the sweep type is power sweep.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<value> Input power value in units of dBm.

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:MIX:INP:POW STAR 6</td>
</tr>
<tr>
<td>SENSE2:MIXer:INPut:POWer:STARt 5</td>
</tr>
</tbody>
</table>

Query Syntax SENSE<ch>:MIXer:INPut:POWer:STARt?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:INPut:POWer:STOP <value>

Applicable Models: All
(Read-Write) Sets the input stop power for a power sweep in a mixer channel. The value is only used when the sweep type is power sweep.

**Parameters**

- **<ch>**
  - Any existing channel number. If unspecified, value is set to 1.
- **<value>**
  - Input power value in units of dBm.

**Examples**

- `SENS:MIX:INP:POW STOP 9`
- `SENS2:MIXer:INPut:POWer:STOP 5`

**Query Syntax**

`SENS<ch>:MIXer:INPut:POWer:STOP?`

**Return Type**

Numeric

**Default**

Not Applicable

---

**SENSe<ch>:MIXer:INPut:POWer:USENominal <bool>**

**Applicable Models:** All

(Read-Write) Toggles the Use Nominal Incident Power setting ON and OFF. This setting is ONLY to be used with SMC measurements. Learn more about Nominal Incident Power.

**Parameters**

- **<ch>**
  - Any existing channel number. If unspecified, value is set to 1.
- **<value>**
  - (boolean) - Nominal Incident Power State. Choose from:
    - **ON (1)** - Turn nominal incident power ON
    - **OFF (0)** - Turn nominal incident power OFF

**Examples**

- `SENS:MIX:INP:POW:USEN 1`
- `SENS2:MIXer:INPut:POWer:USENominal OFF`

**Query Syntax**

`SENS<ch>:MIXer:INPut:POWer:USENominal?`

**Return Type**

Boolean

**Default**

OFF

---

**SENSe<ch>:MIXer:LO<n>:FREQuency:DENominator <value>**
Applicable Models: All

(Read-Write) Sets or returns the denominator value of the LO Fractional Multiplier. See Note

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.
<n> LO stage number. Choose 1 or 2.
<value> LO denominator.

Examples

SENS:MIX:LO:FREQ:DEN 5
SENS2:MIXer:LO2:FREQ:DENominator 4

Query Syntax

SENSe<ch>:MIXer:LO<n>:FREQuency:DENominator?

Return Type

Numeric

Default

1

SENSe<ch>:MIXer:LO<n>:FREQuency:FIXed <value>

Applicable Models: All

(Read-Write) Sets or returns the fixed frequency of the specified mixer LO. See Note

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.
<n> LO stage number. Choose 1 or 2
<value> LO frequency.

Examples

SENS:MIX:LO:FREQ:FIX 1e9
SENS2:MIXer:LO2:FREQ:FIXed 1000000000

Query Syntax

SENSe<ch>:MIXer:LO<n>:FREQuency:FIXed?

Return Type

Numeric

Default

Not Applicable

SENSe<ch>:MIXer:LO<n>:FREQuency:ILTI <bool>
Applicable Models: All

*(Read-Write)* Specifies whether to use the Input frequency that is **greater than** the LO or **less than** the LO. To learn more, see the mixer setup dialog box help.

Parameters

- **<ch>**: Any existing channel number. If unspecified, value is set to 1.
- **<n>**: LO stage number. Choose 1 or 2
- **<bool>**: ON (1) - Use the Input that is Greater than the specified LO.
  
  OFF (0) - Use the Input that is Less than the specified LO.

Examples

```
SENS:MIX:LO1:FREQ:ILTI 1
sense2:mixer:lo2:frequency:ilti ON
```

Query Syntax

```
SENSe<ch>:MIXer:LO<n>:FREQuency:ILTI?
```

Return Type: Boolean

Default: OFF

---

**SENSe<ch>:MIXer:LO<n>:FREQuency:MODE <char>**

Applicable Models: All

*(Read-Write)* Sets or returns the LO sweep mode.

Parameters

- **<ch>**: Any existing channel number. If unspecified, value is set to 1.
- **<n>**: LO stage number. Choose 1 or 2
- **<char>**: LO sweep mode. Choose either FIXED or SWEPT

Examples

```
SENS:MIX:LO:FREQ:MODE FIXED
SENSe2:MIXer:LO2:FREQ:MODE swept
```

Query Syntax

```
SENSe<ch>:MIXer:LO<n>:FREQuency:MODE?
```

Return Type: Character

Default: Fixed

---

**SENSe<ch>:MIXer:LO<n>:FREQuency:NUMerator <value>**
Applicable Models: All

(Read-Write) Sets or returns the numerator value of the LO Fractional Multiplier. See Note

Parameters
<ch> Any existing channel number. If unspecified, value is set to 1.
<n> LO stage number. Choose 1 or 2
<value> LO Numerator.

Examples
SENS:MIX:LO:FREQ:NUM 5
SENSe2:MIXer:LO2:FREQ:NUMerat0r 4

Query Syntax
SENS<ch>:MIXer:LO<n>:FREQuency:NUMerator?

Return Type
Numeric

Default
Not Applicable

SENSe<ch>:MIXer:LO<n>:FREQuency:STARt <value>

Applicable Models: All

(Read-Write) Sets or returns the LO start frequency value. See Note

Parameters
<ch> Any existing channel number. If unspecified, value is set to 1.
<n> LO stage number. Choose 1 or 2
<value> LO Start Frequency in Hertz.

Examples
SENS:MIX:LO:FREQ:STAR 5E9

Query Syntax
SENS<ch>:MIXer:LO<n>:FREQuency:STARt?

Return Type
Numeric

Default
Not Applicable

SENSe<ch>:MIXer:LO<n>:FREQuency:STOP <value>
Applicable Models: All

(Read-Write) Sets or returns the LO stop frequency value. See Note

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` LO stage number. Choose 1 or 2
- `<value>` LO Stop Frequency in Hertz.

Examples

```
SENS:MIX:LO:FREQ:STOP 5E9
```

Query Syntax

```
SENSe<ch>:MIXer:LO<n>:FREQuency:STOP?
```

Return Type Numeric

Default Not Applicable

---

SENSe<ch>:MIXer:LO<n>:NAME <value>

Applicable Models: All

(Read-Write) Sets or returns the name of the VNA internal source or external source to use as the LO in a converter measurement.

Important Note: This setting is immediately send to the channel configuration. First set and apply mixer frequency settings, then send this command. Otherwise, ‘invalid setting’ errors may occur. See Remotely Specifying a Source Port.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` LO stage number. Choose 1 or 2.
- `<value>` (string) - LO Source name. Use `Source:CAT?` to return a list of valid source ports. An external source must be configured and selected to be valid. Learn more about external source configuration.

Examples

```
SENS:MIX:LO:NAME "MySource"
```

Query Syntax

```
SENSe<ch>:MIXer:LO<n>:NAME?
```

Return Type String

Default "Not Controlled"

---

SENSe<ch>:MIXer:LO<n>:POWer <value>
Applicable Models: All

(Read-Write) Sets or returns the LO Power fixed value.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` LO stage. Choose 1 or 2
- `<value>` LO Power in dBm

Examples

```
SENS:MIX:LO:POW 9
```

Query Syntax

```
SENSe<ch>:MIXer:LO<n>:POWer?
```

Return Type

Numeric

Default

Not Applicable

---

SENSe<ch>:MIXer:LO<n>:POWer:STARt <value>

Applicable Models: All

(Read-Write) For an LO power sweep, sets or returns the LO power start value.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` LO stage. Choose 1
- `<value>` LO start power in dBm

Examples

```
SENS:MIX:LO1:POW:STARt -10
```

Query Syntax

```
SENSe<ch>:MIXer:LO1:POWer:STARt?
```

Return Type

Numeric

Default

- 20 dBm

---

SENSe<ch>:MIXer:LO<n>:POWer:STOP <value>
**Applicable Models:** All

*(Read-Write)* For an LO power sweep, sets or returns the LO power stop value.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` LO stage. Choose 1
- `<value>` LO stop power in dBm

**Examples**

```
SENS:MIX:LO1:POW:STOP 10
```

**Query Syntax**

`SENSe<ch>:MIXer:LO1:POWer:STOP?`

**Return Type**

Numeric

**Default**

-10 dBm

---

**SENSe<ch>:MIXer:LOAD <name>**

**Applicable Models:** All

*(Write-only)* Loads a previously-configured mixer attributes file (.mxr)

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<name>` Path and file name (including .mxr extension) to load.

**Examples**

```
SENSe:MIXer:LOAD "C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Mixer\MyMixer.mxr"
```

**Default**

Not Applicable

---

**SENSe<ch>:MIXer:NORMALize:POINt <value>**
Applicable Models: All

(Read-Write) Sets or returns the data point for normalizing the phase measurement. Learn more.

**Parameters**

- `<ch>` Channel number of the SMC measurement. If unspecified, value is set to 1.
- `<value>` Normalization data point. Choose a data point number between 1 and the max number of data points in the sweep that has the least amount of expected noise.

**Examples**

SEN:$\text{S}:\text{MIX:} \text{NORM:} \text{POIN} \ 101$

sense2:.mixer:normalize:point 50

**Query Syntax**

SEN$\text{S}<\text{ch}>:\text{MIXer:} \text{NORMalize:} \text{POINt}$?

**Return Type** Numeric

**Default** Middle point in the sweep

SEN$\text{S}<\text{ch}>:\text{MIXer:} \text{OUTPut:} \text{FREQuency:} \text{FIXed} \ <\text{value}>$

**Applicable Models:** All

(Read-Write) Sets or returns the output fixed frequency of the mixer. See Note

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Output fixed frequency in Hertz.

**Examples**

SEN$\text{S}<\text{ch}>:\text{MIXer:} \text{OUTPut:} \text{FREQuency:} \text{FIX} \ 5e9$

**Query Syntax**

SEN$\text{S}<\text{ch}>:\text{MIXer:} \text{OUTPut:} \text{FREQuency:} \text{FIXed}$?

**Return Type** Numeric

**Default** Not Applicable

SEN$\text{S}<\text{ch}>:\text{MIXer:} \text{OUTPut:} \text{FREQuency:} \text{MODE} \ <\text{char}>$
Applicable Models: All

(Read-Write) Sets or returns the Output sweep mode.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<char>` Output sweep mode. Choose either **FIXED** or **SWEPT**

**Examples**

```
SENS:MIX:OUTP:FREQ:MODE FIXED
SENS2:MIXer:OUTput:FREQuency:MODE swept
```

**Query Syntax**

SENS<ch>:MIXer:OUTPut:FREQuency:MODE?

**Return Type** Character

**Default** Fixed

---

SENS<ch>:MIXer:OUTPut:FREQuency:SIDeband <value>

Applicable Models: All

(Read-Write) Specify whether to select the sum (High) or difference (Low) products.

- When one LO is used: Input + or - LO1 = Output frequency
- When two LOs are used: IF1 + or - LO2 = Output frequency

Use **SENS:MIX:IF:FREQ:SID** when two LOs are used to determine the IF1 frequency.

Use **Sens:Mixer:Stage** to set 1 or 2 LOs

See Note

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Sideband value. Choose from

  - **LOW** - Low or Difference (-)
  - **HIGH** - High or Sum (+)

**Examples**

```
SENS:MIX:OUTP:FREQ:SID LOW
SENS2:MIXer:OUTput:FREQuency:SIDeband HIGH
```

**Query Syntax**

SENS<ch>:MIXer:OUTPut:FREQuency:SIDeband?

**Return Type** Character
SENSe<ch>:MIXer:OUTPut:FREQuency:STARt <value>

**Applicable Models:** All

*(Read-Write)* Sets or returns the Output start frequency of the mixer. [See Note]

**Parameters**
- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Output start frequency

**Examples**
- `SENSe:MIX:OUTP:FREQ:STAR 1e9`
- `SENSe2:MIXer:OUTPut:FREQ:STARt 1000000000`

**Query Syntax** `SENSe<ch>:MIXer:OUTPut:FREQuency:STARt?`

**Return Type** Numeric

**Default** Not Applicable

SENSe<ch>:MIXer:OUTPut:FREQuency:STOP <value>

**Applicable Models:** All

*(Read-Write)* Sets or returns the Output stop frequency of the mixer. [See Note]

**Parameters**
- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Output stop frequency

**Examples**
- `SENSe:MIX:OUTP:FREQ:STOP 1e9`
- `SENSe2:MIXer:OUTPut:FREQ:STOP 1000000000`

**Query Syntax** `SENSe<ch>:MIXer:OUTPut:FREQuency:STOP?`

**Return Type** Numeric

**Default** Not Applicable

SENSe<ch>:MIXer:PHASe <bool>
Applicable Models: N522xB, N523xB, N524xB

(Read Write) Sets and returns the state of SMC Phase measurements and calibrations. Learn more.

In the User Interface, there are two "enable phase" checkboxes: in the Phase Settings dialog and in the Calibration Wizard. Checking one enables both. This single command also enables both.

Parameters
<ch> Any existing channel number. If unspecified, value is set to 1
<bool> Include Phase measurement state. Choose from
ON or 1 - Include phase in SMC measurements
OFF or 0 - Do NOT include phase in SMC measurements

Examples
SENS:MIX:PHAS 1
sense2:mixer:phase off

Query Syntax
SENS<ch>:MIXer:PHASe?

Return Type
Boolean
Default
0 (OFF)

SENSe<ch>:MIXer:PMap <in>,<out>

Applicable Models: All

(Write-only) Sets the VNA to DUT port map for FCA measurements. Use SENS:MIX:PMAP:INP? and SENS:MIX:PMAP:OUTP? to read these values. Learn about selectable FCA DUT ports.

Changing the ports may limit your ability to use an internal second source. If a selected port is shared by one of the sources, then that source will not be available as an LO source. Learn more about Internal second sources.

Parameters
<ch> Any existing channel number. If unspecified, value is set to 1.
<in> VNA port to connect to the DUT input.

- For SMC, choose any unused VNA port.
- For VMC, set to 1.

<out> VNA port to connect to the DUT output. Choose any unused port for SMC and VMC.
**SENSe<ch>:MIXer:PMAP:INPut?**

**Applicable Models:** All

*(Read-only)* Returns the VNA port that is mapped to the DUT input. Use `SENS:MIX:PMAP` to set this value.

Learn about selectable FCA DUT ports.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.

**Examples**

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SENS:MIX:PMAP?</code></td>
</tr>
<tr>
<td><code>sense2:mixer:pmap?</code></td>
</tr>
</tbody>
</table>

**Default** 1

**SENSe<ch>:MIXer:PMAP:OUTPut?**

**Applicable Models:** All

*(Read-only)* Returns the VNA port that is mapped to the DUT output. Use `SENS:MIX:PMAP` to set this value.

Learn about selectable FCA DUT ports.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.

**Examples**

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SENS:MIX:PMAP?</code></td>
</tr>
<tr>
<td><code>sense2:mixer:pmap:output?</code></td>
</tr>
</tbody>
</table>

**Default** 2

**SENSe<ch>:MIXer:RECalculate**
Applicable Models: All

(Write only) Repeats the last calculation that was performed, including all ON (state) segments in segment table.

**Parameters**
- `<ch>` Any existing channel number. If unspecified, value is set to 1

**Examples**
- **SENS:MIX:REC**

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**SENS<ch>:MIX:REV <bool>**

**Applicable Models:** All

(Read-Write) Sets whether to include SC12 sweeps during measurements.

**Parameters**
- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<bool>` (Boolean) Choose from:
  - **ON (1)** - Include the SC12 (reverse) sweep.
  - **OFF (0)** - Do NOT Include the SC12 (reverse) sweep.

**Examples**
- **SENS:MIX:REV 1**
- **sense2:mixer:reverse ON**

**Query Syntax** **SENS<ch>:MIX:REV?**

**Return Type** Boolean

**Default** ON (1)

---

**SENS<ch>:MIX:ROLE:CATalog?** - Superseded
Applicable Models: All

(Read-only) This command is replaced with SENSE:ROLE:CATalog which can be used by all channels.

Returns a list of valid roles for the IMD Converter application.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.

Examples

- SENSE:MIX:ROLE:CAT?
- sense2:mixer:role:catalog?

Default Not Applicable

SENSe<ch>:MIXer:ROLE:DEVice <role>,<source> Superseded

Applicable Models: All

(Read-Write) This command is replaced with SENSE:ROLE:DEVice which can be used by all channels.

Assigns a configured external source to the specified role for the converter application.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<role>` (String) Role to which the external source is assigned. Choose from:
  - For IMDX and IMSX, choose from:
    - "RF2"
    - "LO1"
    - "LO2"
  - For all other converter applications, choose from:
    - "LO1"
    - "LO2"
- `<source>` (String) Source name from Source Configuration dialog.

Examples

- SENS:MIX:ROLE:DEV "LO1","LO1Name"
- sense2:mixer:role:device "LO1","LO1Name"

Query Syntax

SENSe<ch>:MIXer:ROLE:DEVice? <source>

Return Type String
**SENSe<ch>:MIXer:SAVE <name>**

**Applicable Models:** All

*(Write-only)* Saves the settings for the mixer/ converter test setup to a mixer attributes file.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;name&gt;</td>
<td>Path and file name (including .mxrx extension) to save.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENSe:MIXer:SAVE "C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Mixer\MyMixer.mxr"
```

**Default** Not Applicable

---

**SENSe<ch>:MIXer:STAGe <n>**

**Applicable Models:** All

*(Read-Write)* Number of IF stages (LOs) used in the mixer.  See Note

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;n&gt;</td>
<td>Number of stages. Choose either 1 or 2</td>
</tr>
</tbody>
</table>

**Examples**

```
SENSe1:MIXer:LO1:FREQ:NUMerator 6
SENSe1:BWID 1000
SENSe1:MIXer:SEGment1:ADD 165
'New segments will reset stage to single stage mode. Therefore, always add dual stage setting after adding segments
SENSe1:MIXer:STAGE 2
```

**Query Syntax**  SENSe<ch>:MIXer:STAGe?

**Return Type**  Numeric

**Default**  1

---

**SENSe<ch>:MIXer:XAXis <char>**
Applicable Models: All

(Read-Write) Sets or returns the swept frequency range to display on the X-axis for the IMDx or NFx channel.

For FCA and GCX measurements, use \texttt{CALC:MEAS:MIXer:XAXis}

**Parameters**

- \texttt{<ch> Channel number of the IMDx or NFx Converter measurement. If unspecified, value is set to 1.}
- \texttt{<char> Frequency range to display on the X-Axis. NOT case-sensitive. Choose from:}
  - \texttt{INPUT} - Input frequency range
  - \texttt{LO\_1} - LO frequency range
  - \texttt{LO\_2} - LO 2 frequency range
  - \texttt{OUTPUT} - Output frequency range

If the specified frequency range is not swept, the default swept range is used.

**Examples**

\begin{verbatim}
SENSe:MIXer:XAXis INPUT
sense2:mixer:xaxis LO_1
\end{verbatim}

**Return Type** Character

**Default** Search is made in the following order until a swept range is found:

1. OUTPUT
2. INPUT (If the OUTPUT is fixed)
3. Number of Points (If ALL ranges are fixed)
Sense:Mixer:ELO Commands

Allows measurements of a Mixer with an Embedded LO.

SENSe:MIXer:ELO
  LO:DELTa
  LO:REset
  NORMalize:POInt
  RESet
  STATe

TUNing
  IFBW
  INTerval
  ITERations
  MODE
  NBW
  REnet
  SPAN
  TOLerance

DIAGnostic
  CLEar
  STATus
  SWEeep
  COUNT?
  LO:DELTa?
  MARKer
  ANNotation?
  POSItion?
  STATe?
  PARameter?
  X
  ANNotation?
  START?
  STOP?
  Y
  ANNotation?

Click on a keyword to view the command details.

Note: The Find Now feature on the Embedded LO dialog is performed remotely by doing a 'Single Sweep' with ELO enabled.
See example program.

- Example Programs
- Learn about Embedded LO Settings
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**Note:** The Embedded LO **DIAGnostic** commands read data from the various broadband and precise tuning sweeps, similar to the textual and graphical data that are available in the user interface.

---

**SENSe<ch>:MIXer:ELO:LO:DELTa <num>**

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Sets and returns LO Frequency Delta. There is usually no need to set this value. Read this value to determine the difference between the LO Frequency that is stated in the Mixer dialog box and the last measured LO Frequency.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<num>` LO Frequency delta in Hertz.

**Examples**

```plaintext
SENS:MIX:ELO:LO:DELTa 10.3
```

**Query Syntax**

`SENSe<ch>:MIXer:ELO:LO:DELTa?`

**Return Type**

Numeric

**Default**

Not Applicable

---

**SENSe<ch>:MIXer:ELO:LO:RESet**
Applicable Models: N522xB, N524xB

(Write-only) Resets the LO Delta Frequency to 0 (zero).

Parameters

<ch>  Any existing channel number. If unspecified, value is set to 1

Examples

SENS:MIX:ELO:LO:RES
sense2:mixer:elo:lo:reset

Query Syntax  Not Applicable

Default  Not Applicable

SENSe<ch>:MIXer:ELO:NORMalize:POINt <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the sweep data point around which to perform broadband and precise tuning.

Parameters

<ch>  Any existing channel number. If unspecified, value is set to 1

<num>  Mixer Sweep data point. Choose a data point number, between 1 and the max number of data points in the sweep, that has the least amount of expected noise.

Examples

SENS:MIX:ELO:LO:NORM:POIN 200
sense2:mixer:elo:normalize:point 101

Query Syntax  SENSe<ch>:MIXer:ELO:NORMalize:POINt?

Return Type  Numeric

Default  Center point in the sweep span

SENSe<ch>:MIXer:ELO:RESet
Applicable Models: N522xB, N524xB

(Write-only) Resets the LO Frequency Delta and Tuning parameters to their default settings.

**Parameters**

- **<ch>** Any existing channel number. If unspecified, value is set to 1

**Examples**

```plaintext
SENS:MIX:ELO:RES
sense2:mixer:elo:reset
```

**Query Syntax**

- Not Applicable

**Default**

- Not Applicable

SENSe<ch>:MIXer:ELO:STATe <bool>

**Applicable Models:** N522xB, N524xB

(Read-Write) Sets and returns the ON |OFF state of Embedded LO.

**Parameters**

- **<ch>** Any existing channel number. If unspecified, value is set to 1
- **<bool>** ON | OFF state. Choose from
  - 0 - Embedded LO OFF
  - 1 - Embedded LO ON

**Examples**

```plaintext
SENS:MIX:ELO:STAT 1
sense2:mixer:elo:state 0
```

**Query Syntax**

- SENSe<ch>:MIXer:ELO:STATe?

**Return Type**

- Boolean

**Default**

- OFF

SENSe<ch>:MIXer:ELO:TUNing:IFBW <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the IF Bandwidth for Broadband and Precise tuning sweeps.

Parameters

- `<ch>`  Any existing channel number. If unspecified, value is set to 1
- `<num>`  IF Bandwidth

Examples

```
SENS:MIX:ELO:TUN:IFBW 10kHz
sense2:mixer:elo:tuning:ifbw 20e3
```

Query Syntax

SENSe<ch>:MIXer:ELO:TUNing:IFBW?

Return Type  Numeric

Default  30kHz

SENSe<ch>:MIXer:ELO:TUNing:INTerval <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns how often a tuning sweep is performed.

Parameters

- `<ch>`  Any existing channel number. If unspecified, value is set to 1
- `<num>`  Tuning sweep interval

Examples

```
SENS:MIX:ELO:TUN:INT 2
sense2:mixer:elo:tuning:interval 1
```

Query Syntax

SENSe<ch>:MIXer:ELO:TUNing:INTerval?

Return Type  Numeric

Default  1

SENSe<ch>:MIXer:ELO:TUNing:ITERations <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the maximum number of tuning iterations to achieve the precise tolerance.

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1

<num> Number of tuning iterations. Choose a number between 1 and 100.

**Examples**

```
SENS:MIX:ELO:TUN:ITER 5
sense2:mixer:elo:tuning:iterations 3
```

**Query Syntax**

```
SENSe<ch>:MIXer:ELO:TUNing:ITERations?
```

**Return Type** Numeric

**Default** 5

---

SENSe<ch>:MIXer:ELO:TUNing:MODE <char>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the method used to determine the embedded LO Frequency.

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1

<char> Tuning mode. Choose from:

- **BROADband** Both broadband and precise tuning
- **PRECise** Precise tuning only
- **NONE** No tuning; just apply the LO Frequency Delta value.

**Examples**

```
SENS:MIX:ELO:TUN:MODE BRO
sense2:mixer:elo:tuning:mode precise
```

**Query Syntax**

```
SENSe<ch>:MIXer:ELO:TUNing:MODE?
```

**Return Type** Character

**Default** BROadband

---

SENSe<ch>:MIXer:ELO:TUNing:NBW <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the Noise Bandwidth for Broadband and Precise tuning sweeps.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Noise Bandwidth

**Examples**

- `SENS:MIX:ELO:TUN:NBW 3.2kHz`
- `sense2:mixer:elo:tuning:nbw 3.2e3`

**Query Syntax**

SENSe<ch>:MIXer:ELO:TUNing:NBW?

**Return Type**

Numeric

**Default**

3.2 kHz

---

SENSe<ch>:MIXer:ELO:TUNing:RESet

Applicable Models: N522xB, N524xB

(Write-only) Resets the tuning parameters to their default values.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1

**Examples**

- `SENS:MIX:ELO:TUN:RES`
- `sense2:mixer:elo:tuning:reset`

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

SENSe<ch>:MIXer:ELO:TUNing:SPAN <num>
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the frequency span for the broadband tuning sweep.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Broadband frequency span in Hz.

**Examples**

- SENS:MIX:ELO:TUN:SPAN 1e6
- sense2:mixer:elo:tuning:span 1mhz

**Query Syntax**

SENSe<ch>:MIXer:ELO:TUNing:SPAN?

**Return Type**

Numeric

Default: 3 MHz

SENSe<ch>:MIXer:ELO:TUNing:TOLerance <num>

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the tuning tolerance for precise tuning.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Tuning tolerance in Hz. Choose a number between .001 and 1e3.

**Examples**

- SENS:MIX:ELO:TUN:TOL .5
- sense2:mixer:elo:tuning:tolerance 1

**Query Syntax**

SENSe<ch>:MIXer:ELO:TUNing:TOLerance?

**Return Type**

Numeric

Default: 1 Hz

SENSe<ch>:MIXer:ELO:DIAGnostic:CLEar
Applicable Models: N522xB, N524xB

(Write-only) Clears current diagnostic information.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:MIX:ELO:DIAG:CLEar</td>
</tr>
<tr>
<td>sense2:mixer:elo:diagnostic:clear</td>
</tr>
</tbody>
</table>

**Query Syntax** Not Applicable

**Default** Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:STATus?

Applicable Models: N522xB, N524xB

(Read-only) Returns a string that describes the result of the last tuning sweeps.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:MIX:ELO:DIAG:STAT?</td>
</tr>
<tr>
<td>sense2:mixer:elo:diagnostic:status</td>
</tr>
</tbody>
</table>

**Return Type** String

**Default** Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep:COUNt?

Applicable Models: N522xB, N524xB

(Read-only) Returns the number of tuning sweeps used for the latest embedded LO measurement.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:MIX:ELO:DIAG:SWEep:COUNt</td>
</tr>
<tr>
<td>sense2:mixer:elo:diagnostic:sweep:count?</td>
</tr>
</tbody>
</table>

**Return Type** Numeric

**Default** Not Applicable
SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:LO:DELTa?

Applicable Models: N522xB, N524xB

(Read-only) Returns the LO frequency delta from the specified tuning sweep.

Parameters

<ch>  Any existing channel number. If unspecified, value is set to 1

<n>  Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUNT? to find the number of sweeps taken.

Examples

SENSe:MIX:ELO:DIAG:SWEep2:LO:DELTa?
sense2:mixer:elo:diagnostic:sweep1:lo:delta?

Return Type  Numeric

Default  Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:MARKer:ANNotation?

Applicable Models: N522xB, N524xB

(Read-only) Returns the Y-axis marker value from the specified tuning sweep. This command assumes that a marker was used. Use SENS:MIX:ELO:DIAG:SWE:MARK:STATE? to confirm if a marker was used for the tuning sweep.

Parameters

<ch>  Any existing channel number. If unspecified, value is set to 1

<n>  Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUNT? to find the number of sweeps taken.

Examples

SENSe:MIX:ELO:DIAG:SWEep2:MARKer:ANN?

Return Type  Numeric

Default  Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:MARKer:POSition?
Applicable Models: N522xB, N524xB

(Read-only) Returns the X-axis marker position from the specified tuning sweep.

Parameters

- `<ch>`: Any existing channel number. If unspecified, value is set to 1
- `<n>`: Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUNt? to find the number of sweeps taken.

Examples


Return Type: Numeric

Default: Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:MARKer:STATE?

Applicable Models: N522xB, N524xB

(Read-only) Returns whether or not a marker was used for the specified tuning sweep.

Parameters

- `<ch>`: Any existing channel number. If unspecified, value is set to 1
- `<n>`: Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUNt? to find the number of sweeps taken.

Examples


Return Type: Numeric

Default: Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:PARameter?
Applicable Models: N522xB, N524xB

(Read-only) Returns the name of the parameter of the specified tuning sweep.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUN? to find the number of sweeps taken.

Examples

```
SENS:MIX:ELO:DIAG:SWE2:PAR?
sense2:mixer:elo:diagnostic:sweep1:parameter?
```

Return Type String - either "VC21" or "B,1"

Default Not Applicable

---

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:TITL?

Applicable Models: N522xB, N524xB

(Read-only) Returns the title of the specified tuning sweep.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUN? to find the number of sweeps taken.

Examples

```
SENS:MIX:ELO:DIAG:SWE2:TITL?
sense2:mixer:elo:diagnostic:sweep1:title?
```

Return Type String

Default Not Applicable

---

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:X:ANNotation?
Applicable Models: N522xB, N524xB

(Read-only) Returns the X-Axis annotation of the specified tuning sweep.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Tuning sweep number. Use `SENS:MIX:ELO:DIAG:SWE:COUNT?` to find the number of sweeps taken.

**Examples**

```
sense2:mixer:elo:diagnostic:sweep1:x:annotation?
```

**Return Type** String - either "Hz" or "s"

**Default** Not Applicable

---

`SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:X:STARt?`

Applicable Models: N522xB, N524xB

(Read-only) Returns the start value of the specified tuning sweep.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Tuning sweep number. Use `SENS:MIX:ELO:DIAG:SWE:COUNT?` to find the number of sweeps taken.

**Examples**

```
sense2:mixer:elo:diagnostic:sweep1:x:start?
```

**Return Type** Numeric

**Default** Not Applicable

---

`SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:X:STOP?`
### Applicable Models:
N522xB, N524xB

(Read-only) Returns the stop value of the specified tuning sweep.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1
- `<n>`: Tuning sweep number. Use `SENS:MIX:ELO:DIAG:SWE:COUN?` to find the number of sweeps taken.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SENS:MIX:ELO:DIAG:SWE2:X:STOP?</code></td>
</tr>
<tr>
<td><code>sense2:mixer:elo:diagnostic:sweep1:x:stop?</code></td>
</tr>
</tbody>
</table>

**Return Type**

- Numeric
- **Default**: Not Applicable

---

### Applicable Models:
N522xB, N524xB

(Read-only) Returns Y-axis annotation value of the specified tuning sweep.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1
- `<n>`: Tuning sweep number. Use `SENS:MIX:ELO:DIAG:SWE:COUN?` to find the number of sweeps taken.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sense2:mixer:elo:diagnostic:sweep1:y:annotation?</code></td>
</tr>
</tbody>
</table>

**Return Type**

- String - either "U" or "Phase"
- **Default**: Not Applicable

---

4455
Sense:Multiplexer Commands


```
SENSe:MUlTIplexer:
  ADDRess
  ALLPorts
  CATalog?
  COUNt?
  DISPlay
  INCount?
  LABel
  OUTPut
    | A|B|C|D[DATA]
    | A|B|C|D:VOLTage[DATA]
    | [DATA]
  PORT
    | CATalog?
    | SELect
  STATe
  TSET9
    | OUTPut
    | PORT1
    | PORT2
    | PORT3
    | PORT4
  TYPE
```
Click on a keyword to view the command details.

**Red** commands are superseded.

See Also

- See an example program using these commands.
- Learn about External Test Set Control
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

---

**SENSe:MULTiplexer<id>:ADDResS <address>**

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

*(Read-Write)* Sets and returns the address for the external test set at the specified ID. This command should be immediately preceded by the SENSe:MULT:TYPE command.

**Note:** This command is not applicable to the E509xA USB test sets, on which the address is set by DIP switches on the rear panel.

**Parameters**

- `<id>`  
  Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the SENSe:MULT:TYPE command.

- `<address>`  
  Integer  
  The test set address.

- For a GPIB test set (N44xx and some specials), this is the GPIB address.

- For a test set I/O test set (some specials), it is the position of the test set in the chain (starting at 0).

**Examples**

```
SENS:MULT1:TYPE "Z5623A_K66"  ' use K66 test set, and reference it through ID 1
SENS:MULT1:ADDR 0  ' first test set in sequence
' All subsequent commands using SENS:MULT1 will refer to this test set
```

**Query Syntax**  
SENSe:MULTiplexer<id>:ADDResS?

**Return Type**  
Numeric

**Default**  
Not Applicable
SENSe<cnum>:MULTiplexer<id>:ALLPorts <string>

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

*(Read-Write)* Sets or gets the port selections for all available ports on the specified channel.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<id>`: Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the SENSe:MULT:TYPE command.
- `<string>`: Comma-separated list of port selections, one for each port. Each port selection must correspond to one of the values returned by SENSe:MULT:PORT:CAT?.

Do NOT include + and -.  

**Examples**

```plaintext
' for channel 5 and test set 1, set port 1 to T1,
' port 2 to A, port 3 to R2+, port 4 to R3-.
SENS5:MULT1:ALLP "T1,A,R2,R3 "
```

**Query Syntax** SENSe<cnum>:MULTiplexer<id>:ALLPorts?

**Return Type** STRING

**Default** Not Applicable

---

SENSe:MULTiplexer<id>:CATalog?

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

*(Read-Only)* Returns a comma-separated list of the external test sets models that are currently supported. Choose one of these items to send SENSe:MULT1:TYPE.

**Examples** SENSe:MULT:CAT?

**Return Type** String

**Default** Not Applicable

---

SENSe:MULTiplexer<id>:COUNt?
Applicable Models: N522xB, N523xB, N524xB, E5080A

(Read-Only) Returns the total number of ports of the specified test set.

Returns 0 if no test set is connected (GPIB test sets only).

**Parameters**

- `<id>`  
  Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the SENSe:MULT:TYPE command.

**Examples**

- `SENS:MULT1:COUN?`
- `sense:multiplexer2:count?`

**Return Type**  
Numeric

**Default**  
Not Applicable

---

SENSe:MULTiplexer<id>:DISPlay[:STATe] <bool>

**Applicable Models:** N522xB, N523xB, N524xB, E5080B, M980xA

(Red-Write) Turns ON and OFF the display of the test set control status bar. This status bar indicates the test set that is being controlled and the current port mappings of the active channel.

**Note:** To use this command with the M980xA PXI VNA, the firmware version must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required. This command is effective only in the standard class.

**Parameters**

- `<id>`  
  Id of the external test set. If unspecified, Id is assumed to be 1.

- `<bool>`  
  
  ON (1) Turns ON when in multiport mode.

  OFF (0) Turns OFF when not in multiport mode.

**Examples**

- `SENS:MULT1:DISP 1`
- `sense:multiplexer2:display:state on`

**Query Syntax**  
SENSe:MULTiplexer<id>:DISPlay[:STATe]?

**Return Type**  
Boolean

**Default**  
OFF (0)

---

SENSe:MULTiplexer<id>:INCount?
Applicable Models: N522xB, N523xB, N524xB, E5080B, M980xA

(Read-Only) Returns the number of test set input ports (VNA connection ports).

Note: To use this command with the M980xA PXI VNA, the firmware version must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required.

Parameters

- `<id>`: Id of the external test set. If unspecified, Id is assumed to be 1.

Examples

```
SENS3:MULT1:INC?  ' returns the number of input ports for test set 1 on channel 3
```

Return Type: Numeric

Default: Not Applicable

SENSe<cnum>:MULTiplexer:LABel <string>

Applicable Models: N522xB, N523xB, N524xB, E5080A

(Read-Write) Sets and returns the display label for the testset on the specified channel. The label appears in a status bar at the bottom of the VNA display when SENS:MULT:DISP is set to ON.

Note: This command does not apply to the use of the E509xA test sets.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<string>`: Display label text.

Examples

```
SENS3:MULT:LAB 'High-power output'
```

Query Syntax

SENSe<cnum>:MULTiplexer:LABel?

Return Type: String

Default: Not Applicable

SENSe<cnum>:MULTiplexer<id>:OUTPut:<grp>[[:DATA]] <num>
**Applicable Models:** N522xB, N523xB, N524xB, E5080B, M980xA

*(Read-Write)* Sets or returns the output port data for specified group of the E5092A's control lines. The output port data is set at the beginning of each sweep of the specified channel.

**Note:** To use this command with the M980xA PXI VNA, the firmware version must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required. This command can be used only in the standard class.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<id>`: Id of the external test set either 1 or 2. If unspecified, Id is assumed to be 1.
- `<grp>`: A | B | C | D
- `<num>`: An integer specifying the decimal value of the control line. Values are obtained by adding weights from the following table that correspond to individual lines.

The output port data range is between 0 to 255 (0=All lines are turned OFF and 255 all lines are turned ON).

<table>
<thead>
<tr>
<th>Line</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td>8</td>
<td>128</td>
</tr>
</tbody>
</table>

**Examples**

- `SENS:MULT1:STAT ON`
- `SENS3:MULT1:OUTP:B 8`
  ' let the control line B output the data 8 at the beginning of channel-3 sweep.

**Query Syntax**

```
SENSe<cnum>:MULTiplexer<id>:OUTPut<grp>[:DATa]?
```

**Return Type**

Numeric

**Default**

0
SENSe<cnum>:MULTiplexer<id>:OUTPut<grp>:VOLTage[:DATA] <volt>

Applicable Models: N522xB, N523xB, N524xB, E5080B, M980xA

*(Read-Write)* Sets or returns the output voltage for specified group of the E5092A's control lines. The output voltage is set at the beginning of each sweep of the specified channel.

**Note:** To use this command with the M980xA PXI VNA, the firmware version must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required. This command is effective only in the standard class.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<id>` Id of the external test set either 1 or 2. If unspecified, Id is assumed to be 1.
- `<grp>` A | B | C | D
- `<volt>` Output voltage range for <grp> is between 0 to 5.2V and resolution is 10mV.

**Examples**

```
SENSe:MULT1:STAT ON
SENSe:MULT3:OUTP:B:VOLT 4.2
'Set the control line B voltage to 4.2 V at the beginning of channel-3 sweep.
```

**Query Syntax**

SENSe<cnum>:MULTiplexer<id>:OUTPut<grp>:VOLTage[:DATa]?

**Return Type** Numeric

**Default** 0 V

---

SENSe<cnum>:MULTiplexer<id>:OUTPut[:DATa] <num>

Applicable Models: N522xB, N523xB, N524xB, E5080B, M980xA

*(Read-Write)* Sets or returns the control line value for the specified channel. The output port data is set at the beginning of each sweep of the specified channel. If this command is used when the test set is the E5092A, it reads/writes data just for "group A" of the test set's output lines.

**Note:** To use this command with the M980xA PXI VNA, the firmware version must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required. This command is effective only in the standard class.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<id>` Id of the external test set. If unspecified, Id is assumed to be 1.
- `<numr>` An integer specifying the decimal value of the control line. Values are obtained
by adding weights from the following table that correspond to individual lines.

<table>
<thead>
<tr>
<th>Line</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td>8</td>
<td>128</td>
</tr>
</tbody>
</table>

Note:
- The E5092A interprets SENS:MULT1:OUTP 0 as all lines OFF.

Refer to your test set documentation for setting control line values.

Examples

SENS:MULT1:STAT ON
SENS3:MULT1:OUTP 48
' let the control line output the data 48 at the beginning of channel-3 sweep.

Query Syntax
SENSe<cnum>:MULTiplexer<id>:OUTPut[:DATa]?

Return Type Numeric

Default Not Applicable

SENSe:MULTiplexer<id>:PORT<pnum>:CATalog?
Applicable Models: N522xB, N523xB, N524xB, E5080B, M980xA

(Read-Only) Returns a comma-separated list of valid measurement port selections for the specified VNA connection port.

**Note:** To use this command with the M980xA PXI VNA, the firmware version must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required.

**Parameters**

- `<id>`: Id of the external test set. If unspecified, Id is assumed to be 1.
- `<pnum>`: The VNA connection port number (1 to 4) for which to return the valid test set measurement port selections.

If the E5092A’s configuration is E5092_28, this parameter specifies the individual switch number (1 to 4: SP4T switches, 5 to 10: SPDT switches).

**Examples**

```
SENS:MULT1:PORT3:CAT?   ' returns the valid port selections for port 3
```

**Return Type**

String

**Default**

Not Applicable

---

**SENSe<cnum>:MULTiplexer<id>:PORT<pnum>:SESelect <string>**

Applicable Models: N522xB, N523xB, N524xB, E5080B, M980xA

(Write-Only) Sets a port mapping for a single port. The test set RF measurement path is switched at the specified channel. If this command creates a conflict with an existing port, the VNA will resolve the conflict.

**Note:** To use this command with the M980xA PXI VNA, the firmware version must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required. This command is effective only in the standard class.

**Parameters**

- `<cnum>`: Channel number of the measurement. If unspecified, value is set to 1.
- `<id>`: Id of the external test set. If unspecified, Id is assumed to be 1.
- `<pnum>`: The VNA connection port number (1 to 4).

  If the E5092A’s configuration is E5092_28, this parameter specifies the individual switch number (1 to 4: SP4T switches, 5 to 10: SPDT switches).

- `<string>`: The label of the E5092A’s measurement port.

  If the E5092A’s configuration is E5092_28, this parameter specifies the label of individual switch number.
### Labels of E5092A’s measurement ports in E5092_13 / 16 / 22 / X10 modes

<table>
<thead>
<tr>
<th>Mode</th>
<th>Port 1</th>
<th>Port 2</th>
<th>Port 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>E5092_13</strong></td>
<td>A,T1,T2,T3</td>
<td>T1,T2,T3,T4</td>
<td>R1,R2,R3,R4</td>
</tr>
<tr>
<td></td>
<td>(1A,8COM,9COM,10COM)</td>
<td>(8COM,9COM,10COM,2D)</td>
<td>(3A,3B,3C,3D)</td>
</tr>
<tr>
<td><strong>E5092_16</strong></td>
<td>A1,A2,A3,A4</td>
<td>B1,B2,B3,B4</td>
<td>R1,R2,R3,R4</td>
</tr>
<tr>
<td></td>
<td>(1A,1B,1C,1D)</td>
<td>(2D,2A,2B,2C)</td>
<td>(3A,3B,3C,3D)</td>
</tr>
<tr>
<td><strong>E5092_22</strong></td>
<td>A1,A2,A3,A4,A5,56</td>
<td>A7,A8,A9,A10,A11</td>
<td>B1,B2,B3,B4,B5,B</td>
</tr>
<tr>
<td></td>
<td>(5A,5B,6A,6B,1C,1D)</td>
<td>(8A,8B,2B,2C,2D)</td>
<td>(3A,9A,9B,10A,10</td>
</tr>
<tr>
<td><strong>E5092_X10</strong></td>
<td>1,3,5,7</td>
<td>2,4,6,8</td>
<td>2,4,6,10</td>
</tr>
<tr>
<td></td>
<td>(5COM,6COM,7COM,1D)</td>
<td>(8COM,9COM,10COM,2D)</td>
<td>(8COM,9COM,10C</td>
</tr>
</tbody>
</table>

### Labels of E5092A’s individual switch connections in E5092_28 mode

<table>
<thead>
<tr>
<th>PORT1 (switch1)</th>
<th>PORT2 (switch2)</th>
<th>PORT3 (switch3)</th>
<th>PORT4 (switch4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1A,1B,1C,1D)</td>
<td>(2A,2B,2C,2D)</td>
<td>(3A,3B,3C,3D)</td>
<td>(4A,4B,4C,4D)</td>
</tr>
<tr>
<td><strong>PORT5 (switch5)</strong></td>
<td><strong>PORT6 (switch6)</strong></td>
<td><strong>PORT7 (switch7)</strong></td>
<td><strong>PORT8 (switch8)</strong></td>
</tr>
<tr>
<td>A,B</td>
<td>A,B</td>
<td>A,B</td>
<td>A,B</td>
</tr>
<tr>
<td>(5A,5B)</td>
<td>(6A,6B)</td>
<td>(7A,7B)</td>
<td>(8A,8B)</td>
</tr>
<tr>
<td><strong>PORT9 (switch9)</strong></td>
<td><strong>PORT10 (switch10)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A,B</td>
<td>A,B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(9A,9B)</td>
<td>(10A,10B)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Examples

```plaintext
SENS1:STAT ON
SENS1:MULT1:TYP 'E5092_22'
SENS1:MULT1:PORT1:SEL 'A2'
' set E5092A to 22-port config and let port-1 be connected to 'A2' at the beginning of channel-1 sweep.
```
SENSe:MULTiplexer<id>:STATe <bool>

**Applicable Models:** N522xB, N523xB, N524xB, E5080B, M980xA

*(Read-Write)* Enables and disables (ON/OFF) the port mapping and control line output of the specified test set. Turning this state on will let the E5092A’s paths be switched at the beginning of the sweep for the channel specified by `<cnum>` of `SENS<cnum>:MULT<id>:PORT<pnum>:SEL <string>.

If the specified test set is not connected or not ON, then setting State ON will report an error. All other properties can be set when the test set is not connected.

When this command is set to ON, then the display of the test set status bar (SENS:MULT:DISP) is also set to ON.

**Note:** To use this command with the M980xA PXI VNA, the firmware version must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required.
This command is effective only in the standard class.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;id&gt;</td>
<td>Id of the external test set. If unspecified, Id is assumed to be 1.</td>
</tr>
<tr>
<td>&lt;bool&gt;</td>
<td>ON(1) Enables test set control. OFF (0) Disables test set control.</td>
</tr>
</tbody>
</table>

**Examples**

- `SENS:MULT1:STAT 1`
- `sense2:multiplexer2:state on`

**Query Syntax**

`SENSe<cnum>:MULTiplexer<id>:STATe?`

**Return Type**

Boolean

**Default**

OFF (0)

---

`SENSe<cnum>:MULTiplexer<id>:TSET9:OUTPut[:DATA] <data>` Superseded

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

**Note:** This command is replaced with SENS:MULT:OUTP

**(Read-Write)** Sets the control lines of the specified E5091A. Control lines, provided through a E5091A front panel connector, are used to control external equipment such as a part handler. See your E5091A documentation to learn more about control lines.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;id&gt;</td>
<td>Id of the E5091A test set. Choose from 1 or 2. Learn how to set ID value.</td>
</tr>
<tr>
<td>&lt;data&gt;</td>
<td>Data value used to set control lines. Values are obtained by adding weights from the following table that correspond to individual lines. HIGH =1; LOW=0.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Line</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td>8</td>
<td>128</td>
</tr>
</tbody>
</table>
0 - Sets all lines low

255 - Sets all lines high

**Examples**

```
'St The following sets line 3 and 4 high. All other lines low.
SENS:MULT1:TSET9:OUTP 12
```

**Query Syntax**

```
SEnSe<cnum>:MULTiplexer<id>:TSET9:OUTPut[:DATA]?
```

**Return Type**

Numeric

**Default**

0

**SENSe<cnum>:MULTiplexer<id>:TSET9:PORT1 <char> Superseded**

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

**Note:** This command is replaced with SENS:MULT:ALLPorts which sets ALL ports to the specified outputs.

**(Read-Write)** Switches Port 1 of the specified E5091A to one of the available outputs.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<id>` Id of the E5091A test set. Choose from 1 or 2. Learn how to set ID value.
- `<char>` Output port to be switched to. Choose from:
  - A

**Examples**

```
SENS:MULT1:TSET9:PORT1 A
```

**Query Syntax**

```
SEnSe<cnum>:MULTiplexer<id>:TSET9:PORT1?
```

**Return Type**

Character

**Default**

A

**SENSe<cnum>:MULTiplexer<id>:TSET9:PORT2 <char> Superseded**

---

4468
**Applicable Models:** N522xB, N523xB, N524xB, E5080A

**Note:** This command is replaced with SENS: MULT: ALLPorts which sets ALL ports to the specified outputs.

*(Read-Write)* Switches Port 2 of the specified E5091A to one of the available outputs.

**Parameters**
- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<id>`: Id of the E5091A test set. Choose from 1 or 2. Learn how to set ID value.
- `<char>`: Output port to be switched to. Choose from:
  - **T1**: If Port 1 already is connected to T1, then Port 1 will be switched to A.
  - **T2**

**Examples**
```
SENS: MULT1: TSET9: PORT2 T2
```

**Query Syntax**
```
SENSe<cnum>: MULTiplexer<id>: TSET9: PORT2?
```

**Return Type**
Character

**Default**
T1

---

**SENSe<cnum>: MULTiplexer<id>: TSET9: PORT3 <char> Superseded**

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

**Note:** This command is replaced with SENS: MULT: ALLPorts which sets ALL ports to the specified outputs.

*(Read-Write)* Switches Port 3 of the specified E5091A to one of the available outputs.

**Parameters**
- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<id>`: Id of the E5091A test set. Choose from 1 or 2. Learn how to set ID value.
- `<char>`: Output port to be switched to. Choose from:
  - **R1** (R1+)
  - **R2** (R2+)
  - **R3** (R3+) If option 007 (7port), R2 is selected.

**Examples**
```
SENS: MULT1: TSET9: PORT3 R2
```
SENSe<cnum>:MULTiplexer<id>:TSET9:PORT4 <char> **Superseded**

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

**Note:** This command is replaced with SENS:MULT:ALLPorts which sets ALL ports to the specified outputs.

**Parameters**
- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<id>`: Id of the E5091A test set. Choose from 1 or 2. Learn how to set ID value.
- `<char>`: Output port to be switched to. Choose from:
  - **R1** (R1-)
  - **R2** (R2-)
  - **R3** (R3-) If option 007 (7port), R2 is selected.

**Examples**

```
SENSe:MULT1:TSET9:PORT4 R2
```

**Query Syntax**

SENSe<cnum>:MULTiplexer<id>:TSET9:PORT4?  
**Return Type** Character  
**Default** R1

---

**SENSe:.MULTiplexer<id>:TYPE <name>**
Applicable Models: N522xB, N523xB, N524xB, E5080B, M980xA

(Read-Write) Specifies the configuration of the E5092A test set.

Note: To use this command with the M980xA PXI VNA, the firmware version must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required.

Parameters

- `<name>` String  The name of the type of test set. Must be one of the items in the list returned by the SENSe:MULT:CATalog? query – E5092_13, E5092_16, E5092_22, E5092_28, E5092_X10
- `<id>` Id of the external test set. Set by this command. Use consecutive values starting at 1.

Examples

```
SENS:_MULT1:TYPE 'E5092_22' ' set E5092A to 22-port config.
```

Query Syntax

`SENSe:MULTiplexer<id>:TYPe?`

Return Type

String

Default

Not Applicable
Controls the Noise Figure / NFX configuration and calibration.

<table>
<thead>
<tr>
<th>SENSe:NOISe:</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVERage &lt;num&gt;</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>BWIDth &lt;num&gt;</td>
</tr>
<tr>
<td>CALibration:</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>ENR</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>EXDC:NAME</td>
</tr>
<tr>
<td>GAIN &lt;num&gt;</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>IMPedance:COUNt &lt;num&gt;</td>
</tr>
<tr>
<td>NARRowband[:STATe] &lt;bool&gt;</td>
</tr>
<tr>
<td>PMAP &lt;in&gt;,&lt;out&gt;</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>PULL[:STATe] &lt;bool&gt;</td>
</tr>
<tr>
<td>RECeiver &lt;char&gt;</td>
</tr>
<tr>
<td>SNP? &lt;string&gt;</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>SOURce:</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>CONNector &lt;string&gt;</td>
</tr>
<tr>
<td>SWEep</td>
</tr>
<tr>
<td>TIMe?</td>
</tr>
<tr>
<td>TEMPerature:AMBient &lt;num&gt;</td>
</tr>
<tr>
<td>TUNer:</td>
</tr>
<tr>
<td>FILE:NAME &lt;string&gt;</td>
</tr>
<tr>
<td>FILE[:STATE]</td>
</tr>
<tr>
<td>ID &lt;string&gt;</td>
</tr>
<tr>
<td>INPut &lt;string&gt;</td>
</tr>
<tr>
<td>ORlent[:STATE]</td>
</tr>
<tr>
<td>OUTPut &lt;string&gt;</td>
</tr>
<tr>
<td>USBSource:</td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>ENR</td>
</tr>
<tr>
<td>SAVE</td>
</tr>
<tr>
<td>[:SELect]</td>
</tr>
<tr>
<td>TEMPerature?</td>
</tr>
</tbody>
</table>

Click on a keyword to view the command details.

**Other Noise Figure SCPI commands**

The calibration commands listed in this topic are supplemental to the Guided Cal commands.

- **CALCulate:MEASure:DEFine** - creates a noise figure measurement.
- **CONTroll:NOISe:SOURce** or **OUTPut:MANual:NOISe[:STATE]** - turns the Noise Source ON and OFF.
- **SENSe:PATH:CONF:ELEMent[:STATE]** - sets the port 1 and port 2 noise switches.
- **SENS:CORR:ENR:CAL**: manage ENR data - usually not necessary.
- **SYST:PREF:ITEM:SWIT:DEF**: Sets the default setting of the Noise Tuner switch
- **SENS:CORR:NOISe** commands - noise calibration
- **SENS:CORR:Guided** commands - performs most of noise cal.

See Also
- **Examples**:
  - Create and Cal a Noise Figure Measurement
  - Create and Cal an NFX Measurement
- Learn about Noise Figure Application
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

### SENSE<ch>:NOISe:AVERage[:COUNt] <num>

**Applicable Models**: All with with Noise Figure Option (S9x029A/B, 028, 029)

*(Read-Write)* Set and read the averaging factor for the noise receiver. Learn more

**Parameters**

- **<ch>** Noise Figure channel number. If unspecified, value is set to 1.
- **<num>** Averaging value. Choose any number from 1 to 16000.

**Examples**

```
SENS:NOIS:AVER 20
sense:noise:average:count 10
```

**Query Syntax**

SENSe:NOISe:AVERage[:COUNt]?

**Return Type**

Numeric

**Default**

1

### SENSE<ch>:NOISe:AVERage:STATe <bool>
Applicable Models: VNAs with Noise Figure Option (S9x029A/B, 028, 029)

(Read-Write) Turns noise averaging ON and OFF.

Parameters

- `<ch>` Noise Figure channel number. If unspecified, value is set to 1.
- `<bool>` Averaging state. Choose from
  - 0 - Noise averaging OFF
  - 1 - Noise averaging ON

Examples

```
SENS:NOIS:AVER:STAT 0
sense:noise:average:state 1
```

Query Syntax `SENSe:NOISe:AVERage:STATe?`

Return Type Boolean

Default O - OFF

---

SENSe[ch]>NOISe:BWIDth[:RESolution] <num>

Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-Write) Set and read the bandwidth of the noise receiver. Learn more

Parameters

- `<ch>` Noise Figure channel number. If unspecified, value is set to 1.
- `<num>` Bandwidth value. Choose from:
  - For `Sens:Noise:Receiver` = NOISe (Opt 029) choose from: 800 KHz, 2 MHz, 4 MHz, 8 MHz, or 24 MHz or the numerical equivalent, such as 8e6 and so forth.
  - For `Sens:Noise:Receiver` = NORMal (Opt 028) choose from: 720 kHz or 1.2 MHz
  - If the value does not match one of these, it is rounded up to the next valid bandwidth value.

Examples

```
SENS:NOIS:BWID 2e6
sense:noise:bwidth:resolution 8mhz
```

Query Syntax `SENSe:NOISe:BWIDth[:RESolution]?`

Return Type Numeric
SENSe<ch>:NOISe:CALibration:METHod <string>

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-Write) Set and read the method for performing a calibration on a noise channel.

**Parameters**
- `<ch>` Noise Figure channel number. If unspecified, value is set to 1.
- `<string>` Calibration method. NOT case-sensitive. Choose from:
  - "VectorFull" or "Vector"
  - "SParameter" (Not available for NFX measurements)
  - "ScalarFull" or "Scalar"

**Examples**
- `SENSe:NOISe:CAL:METH "Vector"`
- `SENSe:NOISe:CAL:METH "SParameter"

**Query Syntax** `SENSe:NOISe:CALibration:METHod?`

**Return Type** String

**Default** "VectorFull"

SENSe<ch>:NOISe:CALibration:RMETHod <string>
Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-Write) Set and read the method used to characterize the noise receivers. Learn more.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Noise Figure channel number. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;string&gt;</td>
<td>Receiver characterization method. NOT case-sensitive. Choose from:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• &quot;NoiseSource&quot; - Use a noise source. This selection is NOT allowed when a standard VNA receiver is used as the noise receiver (SENS:NOIS:RECNORM).</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• &quot;PowerMeter&quot; - Use a power meter. <strong>NOTE:</strong> This selection is NOT allowed when the Noise Bandwidth is 8 MHz or 24 MHz.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:NOIS:CAL:RMET "PowerMeter"
sense:noise:calibration:rmethod "noisesource"
```

**Query Syntax**

```
SENSe:NOISe:CALibration:RMEThod?
```

**Return Type**

String

**Default**

"NoiseSource"

SENSe<ch>:NOISe:ENR:FILename <string>

Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-Write) Set and read the path and name of the ENR file associated with the noise source.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Noise Figure channel number. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;string&gt;</td>
<td>Full path, filename, and extension of the ENR file.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:NOIS:ENR:FIL "c:\Program Files(x86)\Keysight\Network Analyzer\Documents\ENR\346C.enr"
sense:noise:enr:filename "c:\Program Files(x86)\Keysight\Network Analyzer\Documents\ENR\346C.enr"
```

**Query Syntax**

```
SENSe:NOISe:ENR:FILename?
```

**Return Type**

String

**Default**

Not applicable

SENSe<ch>:NOISe:ENR <INTernal | FILE >
Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-Write) Specifies whether to use the ENR file stored internally on the USB sensor or to use specified .enr file (located on the VNA).

**Note:** 1) If FILE is used, the .enr file should be specified using `SENSe<ch>:NOISe:ENR:FILEnname <string>`
2) If INT is used, then "SENSe:NOISe:ENR:FILEnname?" returns “Internal”

**Parameters**

- **<ch>** Noise Figure channel number. If unspecified, value is set to 1.
- **<INTernal | FILE>** Choose from:
  - INTernal = use internal file on the USB
  - FILE = use separate .enr file.

**Examples**

```
SENS:NOIS:ENR INT
sense:noise:enr file
```

**Query Syntax**

SENSe:NOISe:ENR?

**Return Type**

Characters

**Default**

Not applicable

---

SENSe<ch>:NOIS:EXDC:NAME <string>

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-Write) Set and read the external DC source name in order to control the noise source.

**Parameters**

- **<ch>** Noise Figure channel number but this number is ignored for this command.
- **<string>** DC source name. The DC source should be defined in the external device dialog in advance.

**Examples**

Add DC-biased Noise Source named "NoiseSource1". Open NF channel before sending following commands.

```
SYSTem:CONFigure:EDEVice:ADD "NoiseSource1"
SYSTem:CONFigure:EDEVice:DTYPE "NoiseSource1","DC Source"
SYSTem:CONFigure:EDEVice:DRIVer "NoiseSource1", "DC Source"
SYSTem:CONFigure:EDEVice:IOConfig "NoiseSource1", "GPIB0::23::INSTR"
```
SYSTEM:CONFigure:EDEVice:LOAD "NoiseSourceSettings.xml", "NoiseSource1"

SENSe:NOISe:EXDC:NAME "NoiseSource1"

NOTE: NoiseSourceSettings.xml has setting information about waiting time and SCPI commands to DC Source. Create this xml file before sending SCPI commands.

---

Query Syntax  SENSe:NOISe:EXDC:NAME?
Return Type  String
Default  ""

---

SENSe<ch>:NOISe:GAIN <num>

Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-Write) Set and read the amount of gain for the noise receiver. This setting is NOT used when Sens:Noise:Receiver = NORMal (Opt 028)

Parameters
- <ch>  Noise Figure channel number. If unspecified, value is set to 1.
- <num>  Gain value. Choose from:
  - 0 - Low gain; select if the gain of your DUT is relatively high (>35 dB).
  - 15 - Medium gain; select if the gain of your DUT is about average (20 dB to 45 dB)
  - 30 - High gain; select if the gain of your DUT is relatively low (<30 dB).

Learn more about Noise Receiver Gain setting.

If the value does not match one of these, it is rounded up to the next legal value.

Examples
- SENS:NOIS:GAIN 15
- sense:noise:gain 0

Query Syntax  SENSe:NOIS:GAIN?
Return Type  Numeric
Default  30

---

SENSe<ch>:NOISe:GAIN:CTCheck <bool>

---
Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-Write) Turns noise threshold checking ON and OFF.

**Parameters**

- `<ch>`  Noise Figure channel number. If unspecified, value is set to 1.
- `<bool>`  Threshold checking state. Choose from
  
  - 0 - Noise threshold checking OFF
  - 1 - Noise threshold checking ON

**Examples**

```plaintext
SENS:NOIS:GAIN:CTC 0
sense:noise:gain:ctcheck 1
```

**Query Syntax**  
SENSe:NOISe:GAIN:CTCheck?

**Return Type**  
Boolean

**Default**  
O - OFF

SENSe<ch>:NOISe:IMPedance:COUNt <num>

Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-Write) Sets the number of impedance states to use during calibrated measurements.

**Parameters**

- `<ch>`  Noise Figure channel number. If unspecified, value is set to 1.
- `<num>`  Number of impedance states to use. Choose between 4 and the maximum number allowed by the noise tuner device. The more states that are used, the more accurate, and slower, the measurement. If the specified number exceeds the capability of the device, the measurement will use the maximum number of states the device allows.

**Examples**

```plaintext
SENS:NOIS:IMP:COUN 5
sense:noise:impedance:count 7
```

**Query Syntax**  
SENSe:NOISe:IMPedance:COUNt?

**Return Type**  
Numeric

**Default**  
4

SENSe<ch>:NOISe:NARrowband[:STATe] <bool>
Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029) (Excepts M9485A, M980xA, P50xxA)

(Read-Write) Turns narrowband noise figure compensation ON and OFF

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <bool> Compensation state. Choose from

  0 or OFF - Narrowband noise compensation OFF

  1 or ON - Narrowband noise compensation ON

Examples

- `SENS:NOIS:NARR 0`
- `sense:noise:narrowband:state 1`

Query Syntax

`SENSe:NOISe:NARRowband[:STATe]?`

Return Type

Boolean

Default

O - OFF

SENSe<ch>:NOISe:PMAP <in>,<out>

Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029)

(Write-only) Set the DUT-to-VNA port mapping for Noise Figure. Port mapping is allowed without restriction when the standard PNA receiver is used (`SENSe:NOISe:RECeiver` is set to `NORMal`). When the low-noise receiver is selected (`SENSe:NOISe:RECeiver` is set to `NOISe`) the following restrictions apply:

- If the low-noise receiver is selected, the DUT output port must be port 2.
- On high-frequency PNAs that have an internal tuner on port 1, the input port must be port 1 if the internal tuner is selected as the noise tuner. Conversely, if the input port is something other than 1, the internal tuner cannot be selected.
- For PNAs that have a maximum frequency of 26.5 GHz or less, any port can be selected as the DUT input port.
- If a vector calibration is desired, the tuner must be connected to the selected input port.

Use the `SENSe:NOISe:PMAP:INPut?` and `SENSe:NOISe:PMAP:OUTPut?` commands to read the DUT input and output ports.

Parameters

- <ch> Any existing NF or NFX channel. If unspecified, value is set to 1.
<in> VNA port which is connected to the DUT input.
<out> VNA port which is connected to the DUT output.

**Examples**

SENS:NOIS:PMAP 1,2  
*sense:noise:pmap 2,1*

See example program

**Query Syntax**

Not Applicable

**Default**

1,2

---

**SENSe<ch>:NOISe:PMAP:INPut?**

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-only) Read the VNA port number to be connected to the DUT Input.

Use SENS:NOISE:PMAP to set the port mapping.

**Parameters**

<ch> Any existing NF or NFX channel. If unspecified, value is set to 1.

**Examples**

SENS:NOIS:PMAP:INP?  
sense:noise:pmap:input?

**Return Type**

Numeric

**Default**

1

---

**SENSe<ch>:NOISe:PMAP:OUTPut?**

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-only) Read the VNA port number to be connected to the DUT Output.

**Parameters**

<ch> Any existing NF or NFX channel. If unspecified, value is set to 1.

**Examples**

SENS:NOIS:PMAP:OUTP?  
sense:noise:pmap:output?

**Return Type**

Numeric

**Default**

2
SENSe<ch>:NOISe:PULL[:STATe] <bool>

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

*(Read-Write)* Enables and disables the use of source pull technique to compute S22. [Learn more.](#)

**Parameters**

- `<ch>` Noise Figure channel number. If unspecified, value is set to 1.
- `<bool>` Source pull technique state. Choose from:
  - **OFF** or **0** - Disable use of source pull technique.
  - **ON** or **1** - Enable use of source pull technique.

**Examples**

```
SENS:NOIS:PULL 0
sense2:noise:pull:state ON
```

**Query Syntax** SENSe:NOISe:PULL[:STATe]?

**Return Type** Boolean

**Default** 0 - OFF

---

SENSe<ch>:NOISe:RECeiver <char>

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029) (Excepts M9485A, M980xA, P50xxA)

*(Read-Write)* Sets and returns the noise receiver to use for noise measurements.

**Parameters**

- `<ch>` Noise Figure channel number. If unspecified, value is set to 1.
- `<char>` Noise receiver. Choose from:
  - **NORMal** The standard VNA receiver. (Opt 028 and Opt 029)
  - **NOISe** The low-noise receivers. (Opt 029 only)

**Examples**

```
SENS:NOIS:REC NORM
sense2:noise:receiver noise
```

**Query Syntax** SENSe:NOISe:RECeiver?

**Return Type** Character

**Default** NOISe
SENSe<ch>:NOISe:SNP? [string]

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029) (Except M9485A)

*(Read-Only)* Returns S-parameter and noise parameter data for vector noise figure measurements.

Noise parameters are NOT valid for NFX or Scalar noise figure measurements. Learn more about [noise parameters](#).

**Parameters**

<ch> Noise Figure channel number. If unspecified, value is set to 1.


If unspecified, only S-parameter data is saved.

**Examples**

SENSe:NOISe:SNP?

sense2:noise:snp? "NoiseParameter"

**Return Type** Comma-separated values

Data is returned in this order:

1. <all frequencies | point number>

2. <real S11> <imag S11> <real S21> <imag S21> <real S12> <imag S12> <real S22> <imag S22>

Then if noise parameters are specified:

3. <NFMin dB> <mag GammaOpt> <phase GammaOpt> <Rn/Z0>

The data display format depends on MMEM:STOR:TRAC:FORM:SNP

**Default** Not Applicable

SENSe<ch>:NOISe:SNP:SAVE <filename>, [data]

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

*(Write-only)* Saves the S-parameters and vector noise parameters to an S2P file. For NFX channels, mixer setup information is included as comments at the beginning of the file.

Learn more about [noise parameters](#).

The format of the snp data is set with MMEM:STOR:TRAC:FORM:SNP.
The following is sample data for two data points from a Noise Figure measurement:

```
! Keysight Technologies,N5242A,USxxxxxxx,A.09.85
! pnan-nn Thu Nov 01 12:26:27 2012
# HZ S MA R 50
!freq (Hz)   S11M  S11A  S21M  S21A  S12M  S12A  S22M  S22A
2000000000  9.038147e-001  6.241193e+001  5.855965e+000 -6.116778e+001  2.232653e-002 -1.475392e+002  5.275644e-001  1.750775e+002
8000000000  6.951366e-001 -1.458202e+002  5.307699e+000 -7.055212e+001  7.838612e-002 -1.460951e+002  3.986142e-001 -2.226317e+001

! Noise Parameters
!freq (Hz)   NFMin(dB)  Rho_opt(Mag)  Rho_opt(deg)  Rn/Z0
2000000000  1.251697e+000  2.172018e-001 -8.765875e+001  1.806663e-001
8000000000  1.583849e+000  2.015185e-001  1.029875e+001  1.320403e-001
```

**Parameters**

- `<ch>`: Noise Figure channel number. If unspecified, value is set to 1.
- `<filename>`: Path (optional), filename and suffix of location to store SNP data. If path is not specified, the current path is used.
  
  If unspecified, only S-parameter data is saved.

**Examples**

```
SENS:NOIS:SNP:SAVE "MySparams.s2p"

sense2:noise:snp:save "C:\Program Files(x86)\Keysight\Network Analyzer\Documents\MyNoiseParams.s2p", "NoiseParameter"
```

**Query Syntax**

- **Default**: Not Applicable

```
SENS<ch>:NOISe:SOURce:CKIT <string>
```
**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

*(Read-Write)* Set and read the Cal Kit that will be used for the Noise Source adapter.

An adapter is always necessary to connect a 346C Noise Source to the VNA port 2. Select a Cal Kit that is the same type and gender as the noise source connector.

If the Noise Source mates directly to VNA port 2, then set this type to "None".

**Parameters**

- `<ch>` Noise Figure channel number. If unspecified, value is set to 1.
- `<string>` Cal Kit. Case sensitive.

To read possible cal kit strings for the adapter:

- Change the port connector type to that of the noise source using:
  ```
  SENS:CORR:COLL:GUID:CONN:PORT<n>
  ```
- Then read the possible cal kit strings for that port using:
  ```
  ```

**Examples**

```bash
SENS:NOIS:SOUR:CKIT "N4691-60004 ECal"
```

**Query Syntax**

```
SENSe:NOISe:SOURce:CKIT?
```

**Return Type**

String

**Default**

Not applicable

```
SENSe<ch>:NOISe:SOURce:CONNector <string>
```
Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-Write) Set and read the Noise Source connector type and gender. The Keysight 346C has an "APC 3.5 male" connector.

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <string> Noise source connector type and gender. Case sensitive.


Examples

```
SENS:NOIS:SOUR:CONN "APC 3.5 male"
sense:noise:source:connector "APC 3.5 female"
```

Query Syntax

`SENS:NOIS:SOUR:CONN?`

Return Type

String

Default

Not applicable

---

SENSe<ch>:NOIS:SWEep:TlMe?

Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029)

(Read-only) Returns the APPROXIMATE time the channel will take to make one noise receiver sweep given the current setup. This, along with the sweep time for a standard receiver measurement and the following calculations, can tell you how long a “single” sweep would take so that you can set an appropriate "timeout" in your program.

To calculate the total sweep time:

Noise Figure on amplifiers (Vector Correction ON):

- \( 2 \times SSwpTime + X \times NoiseReceiverSweepTime \)
- Where \( X \) = the number of noise receiver impedance state sweeps. (Default is 4).

Noise Figure on converters (NFX) correction on - increased number of sweeps due to extra mixer sweeps and source pulling:

- \( 4 \times SSwpTime + X \times NoiseReceiverSweepTime \) (without source pulling)
- \( 8 \times SSwpTime + X \times NoiseReceiverSweepTime \) (with source pulling)
- Where \( X \) = the number of noise receiver impedance state sweeps. (Default is 4).
**Note:** The number of sweeps to perform a noise measurement is annotated at the bottom of the Noise Figure screen.

### Parameters

- `<ch>`  
  Noise Figure channel number. If unspecified, value is set to 1.

### Examples

- `SENS:NOIS:SWE:TIM?`
- `sense:noise:sweep:time?`

### Return Type

- **Double**
- **Default:** Not applicable

---

**SENSe<ch>:NOISe:TEMPerature:AMBient <num>**

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

*(Read-Write)* Sets the temperature at which the current noise measurement is occurring. [Learn more](#)

### Parameters

- `<ch>`  
  Noise Figure channel number. If unspecified, value is set to 1.
- `<num>`  
  Ambient temperature in Kelvin.

### Examples

- `SENS:NOIS:TEMP:AMB 292`
- `sense:noise:temperature 289`

### Query Syntax

- `SENSe:NOISe:TEMPerature:AMBient?`

### Return Type

- **Numeric**
- **Default:** 295

---

**SENSe<ch>:NOISe:TUNer:FILE:NAME <string>**
Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029) (Except M9485A)

(Read-Write) Set and read a custom noise tuner file to be used instead of the one generated automatically based on the state. This is considered an "expert user" feature and is therefore the responsibility of the user to provide a calibration file that yields proper results. Specify the full path of the file. If the full path is not specified, the noise tuner file must be stored in the C:\Users\Public\Documents\Network Analyzer directory. To enable the custom noise tuner file, the SENSE:NOISE:TUNer:FILE:STATE must be set to ON.

Parameters

- `<ch>` Noise Figure channel number. If unspecified, value is set to 1.
- `<string>` Noise tuner file name.

Examples

```
SENS:NOIS:TUN:FILE:NAME "MyNoiseFile.tm"
sense:noise:tuner:file:name "MyNoiseFile.tm"
```

Query Syntax

SENSe:NOIs:TUNer:FILE:NAME?

Return Type

String

Default

Not applicable

SENSe<ch>:NOIS:TUNer:FILE[:STATe] <bool>

Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029) (Except M9485A)

(Read-Write) Sets the state of the custom noise tuner file specified using the SENSE:NOISE:TUNer:FILE:NAME command.

Parameters

- `<ch>` Noise Figure channel number. If unspecified, value is set to 1.
- `<bool>` Custom noise tuner file state. Choose from:
  - OFF or 0 - Disable custom noise tuner file
  - ON or 1 - Enable custom noise tuner file

Examples

```
SENS:NOIS:TUN:FILE 1
sense2:noise:tuner:file:state ON
```

Query Syntax

SENSe:NOIS:TUNer:FILE[:STATe]?

Return Type

Boolean

Default

0 - OFF
SENSe<ch>:NOISe:TUNer:ID <string>

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029) (Except M9485A)

*(Read-Write)* Set and read the identity of the noise tuner. This is an ECal model and serial number string. To read the identities of the connected ECal modules, use **SENSe:CORRection:CKIT:ECAL:LIST?** and **SENSe:CORRection:CKIT:ECAL<mod>:INFormation?**

**Parameters**
- `<ch>` Noise Figure channel number. If unspecified, value is set to 1.
- `<string>` ECal model and serial number string. The ECal module must be connected when this command is sent.

**Examples**
- `SENSe:NOIS:TUN:ID "N4691-60004 ECal 02822`n
- `sense:noise:tuner:id "N4691-60004 ECal 02822"

**Query Syntax** **SENSe:NOISe:TUNer:ID?**

**Return Type** String

**Default** Not applicable

SENSe<ch>:NOISe:TUNer:INPut <string>

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029) (Except M9485A)

*(Read-Write)* Sets and reads the port of the ECal noise tuner that is connected to the VNA SOURCE OUT.

**Parameters**
- `<ch>` Noise Figure channel number. If unspecified, value is set to 1.
- `<string>` ECal port identifier. Case sensitive.

**Examples**
- `SENSe:NOIS:TUN:INP "B"

- `sense:noise:tuner:input "A"

**Query Syntax** **SENSe:NOISe:TUNer:INPut?**

**Return Type** String

**Default** "B"

SENSe<ch>:NOISe:TUNer:ORIent[:STATe] <bool>
Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029) (Except M9485A)

(Read-Write) Sets the state of auto orientation for a noise tuner during Noise Figure for NFX.

Parameters

<ch> Noise Figure channel number. If unspecified, value is set to 1.
<bool> Auto-orientation state. Choose from:

OFF or 0 - Disable Auto-orientation
ON or 1 - Enable Auto-orientation

Examples

SENSe:NOISe:TUNer:ORIent 0
sense2:noise:tuner:orient:state ON

Query Syntax

SENSe:NOISe:TUNer:ORIent[:STATe]?

Return Type

Boolean

Default

1 - ON

SENSe<ch>:NOISe:TUNer:OUTPut <string>

Applicable Models: All with Noise Figure Option (S9x029A/B, 028, 029) (Except M9485A)

(Read-Write) Sets and reads the port of the ECal noise tuner that is connected to the CPLR THRU.

Parameters

<ch> Noise Figure channel number. If unspecified, value is set to 1.
<string> ECal port identifier. Case sensitive.

Examples

SENSe:NOISe:TUNer:OUTP "B"
sense:noise:tuner:output "A"

Query Syntax

SENSe:NOISe:TUNer:OUTPut?

Return Type

String

Default

"A"

SENSe<ch>:NOISe:USBSource:CATalog?
**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

*(Read only)* Returns a comma separated list of connected USB Noise Sources, identified by their `usbNsSrcID`: “ModelNumber SerialNumber.”

**Parameters**
- `<ch>` Noise Figure channel number. If unspecified, value is set to 1.

**Examples**
- `SENS:NOIS:USB:S:CAT?`
- `sense:noise:usbsource:catalog?`

**Return Type** String
- **Default** Not applicable

---

```plaintext
SENS<ch>:NOISe:USBSource:ENR:SAVE " usbNsSrcID ","filename"
```

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

*(Write only)* This writes the ENR file stored in USB Noise Sensor to the filename specified by “filename”

**Note:** When a full path is not given, the .enr file will be written to the default folder. Change the default folder name using `MMemory:CDIrectory`.

**Parameters**
- `<ch>` Noise Figure channel number. If unspecified, value is set to 1.

**Examples**
- `SENS:NOIS:ENR:SAVE "U1831C MY12345678","C:\myUSBsensor.enr"
- `sense:noise:enr:save "U1231C MY12345678", "my.enr"

**Query Syntax** None
- **Default** Not applicable

---

```plaintext
SENS<ch>:NOISe:USBSource[:SELect] "usbNsSrcID"
```
**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

**Parameters**

- `<ch>`: Noise Figure channel number. If unspecified, value is set to 1.

**Examples**

```plaintext
SENS:NOIS:USBSEL "U1831C MY12345678"

sense:noise:usbsource:select "U1831C MY12345678"
```

**Query Syntax**

SENS:NOISE:USBSource?

**Return Type**

String

**Default**

Not applicable

---

**SENSe<ch>:NOISe:USBSource:TEMPerature? "usbNsSrcID"**

**Applicable Models:** All with Noise Figure Option (S9x029A/B, 028, 029)

**Parameters**

- `<ch>`: Noise Figure channel number. If unspecified, value is set to 1.

**Examples**

```plaintext
SENS:NOIS:USBSTEMP? "U1831C MY12345678"

sense:noise:usbsource:temperature? "U1831C MY12345678"
```

**Return Type**

Numeric

**Default**

Not applicable

---

**Note:** This can be used to help determine what temperature to use for the noise source temperature during calibration. Learn more about ConfigureNSDiag.

The commands to set the noise source connector temperature for calibration remotely are:

- **Smart Cal:** `SENSe<ch>:CORRection:TCOLd:USER:VALue<num>`
- **Cal All:** `SYSTem:CALibrate:ALL[1-250]:MCLass:PROPerty:VALue[:STATE] "Noise Source Temperature",<value>`
Sense:Offset Commands **Superseded**

**Note:** These commands are replaced by the Sense:FOM commands which include the features of the new FOM dialog. Although these old commands will continue to work, they can NOT be mixed with the new commands.

Sets the offset frequency functions, causing the stimulus and response frequencies to be different.

Click on a keyword to view the command details.

**See Also**

- Example Programs
- Learn about Frequency Offset
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

SENSe<cnum>:OFFSet:CW <bool>

**Applicable Models:** N522xB, N523xB, N524xB

**(Read-Write)** Turns stimulus CW Override mode ON or OFF. Use this setting to establish a fixed (CW) stimulus frequency while measuring the Response over a swept frequency range.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` ON (or 1) - turns CW override ON.
  
  OFF (or 0) - turns CW override OFF.

**Examples**

```
SENS:OFFS:CW ON
sense2:offset:off
```

**Query Syntax**

SENSe<cnum>:OFFSet:CW?

**Return Type**

Boolean

**Default**

OFF
SENSe<cnum>:OFFSet:DIVisor <num>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Specifies (along with the multiplier) the value to multiply by the stimulus.

Parameters
<cnun> Any existing channel number. If unspecified, value is set to 1
<num> Divisor value. Range is 1 to 1000

Examples
SENSe:OFFS:DIV 3
sense2:offset:divisor 2

Query Syntax SENSe<cnum>:OFFSet:DIVisor?
Return Type Numeric
Default 1

SENSe<cnum>:OFFSet:MULTiplier <num>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Specifies (along with the divisor) the value to multiply by the stimulus.

Parameters
<cnun> Any existing channel number. If unspecified, value is set to 1
<num> Multiplier value. Range is +/- 1000. Negative multipliers cause the stimulus to sweep in decreasing direction. For mixer measurements, this would be for setups requiring the RF frequency to be less than LO frequency

Examples
SENSe:OFFS:MULT 2
sense2:offset:multiplier 4

Query Syntax SENSe<cnum>:OFFSet:MULTiplier?
Return Type Numeric
Default 1

SENSe<cnum>:OFFSet:OFFSet <num>
**Applicable Models:** N522xB, N523xB, N524xB

(Read-Write) Specifies an absolute offset frequency in Hz. For mixer measurements, this would be the LO frequency.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Offset frequency. Range is +/- 1000 GHz. Offsets can be positive or negative

**Examples**

```
SENS:OFFS:OFFS 1GHz
sense2:offset:offset 1e9
```

**Query Syntax**

`SENSe<cnum>:OFFSet:OFFSet?`

**Return Type** Numeric

**Default** 0 Hz

---

`SENSe<cnum>:OFFSet:STARt?`

**Applicable Models:** N522xB, N523xB, N524xB

(Read-Only) Returns the response start frequency

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Examples**

```
SENS:OFFS:STAR?
sense2:offset:start?
```

**Return Type** Numeric

**Default** Not applicable

---

`SENSe<cnum>:OFFSet:[STATe] <bool>`
**Applicable Models:** N522xB, N523xB, N524xB

(Read-Write) Enables Frequency Offset Mode on ALL measurements that are present on the active channel. This immediately causes the source and receiver to tune to separate frequencies. The receiver frequencies are specified with the other SENS:OFFSet commands. To make the stimulus settings use the SENS:FREQ commands.

Tip: To avoid unnecessary errors, first make other offset frequency settings, then set Frequency Offset ON.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` ON (or 1) - turns Frequency Offset ON.
  
  OFF (or 0) - turns Frequency Offset OFF.

**Examples**

```latex
SENS:OFFS ON
sense2:offset:state off
```

**Query Syntax**

SENSe<cnun>:OFFSet:[STATe]?

**Return Type** Boolean

**Default** OFF (0)

---

**SENSe<cnun>:OFFSet:STOP?**

**Applicable Models:** N522xB, N523xB, N524xB

(Read-Only) Returns the response stop frequency.

**Parameters**

- `<cnun>` Any existing channel number. If unspecified, value is set to 1

**Examples**

```latex
SENS:OFFS:STOP
sense2:offset:stop
```

**Return Type** Numeric

**Default** Not applicable
Controls the path configuration settings.

<table>
<thead>
<tr>
<th>SENSE:PATH:CONFig</th>
</tr>
</thead>
<tbody>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>COPY</td>
</tr>
<tr>
<td>DELete</td>
</tr>
<tr>
<td>DTEXt</td>
</tr>
<tr>
<td>ELEMent</td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>[STATE]</td>
</tr>
<tr>
<td>VALUE:CATalog</td>
</tr>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>SELect</td>
</tr>
<tr>
<td>STORe</td>
</tr>
</tbody>
</table>

Click on a keyword to view the command details.

The 'ELEMent' commands are used for both RF path and IF path configuration.

- RF Configuration elements and values
- IF Configuration elements and values

See Also

- Example Programs
- Learn about RF Path Configuration
- Learn about IF Path Configuration
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

SENSe:PATH:CONFig:CATalog?
Applicable Models: N522xB, N523xB, N524xB

(Read-only) Returns a list of configuration names stored in the VNA.

<table>
<thead>
<tr>
<th><strong>Examples</strong></th>
<th><strong>SENSe:PATH:CONF:CAT?</strong></th>
</tr>
</thead>
</table>

**Return Type** Comma-separated list of double-quoted strings

**Default** Not Applicable

SENSe<ch>:PATH:CONF:COPY <num>

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Copies the mechanical switch and attenuator settings from the specified channel <num> to channel <ch>.

To avoid potential conflicts, all port couplings in the calling channel will be turned OFF and all port attenuator settings will be set to manual before copying the switch or attenuator settings. The two channels CAN be of different measurement classes.

Use SYSTem:MACRo:COPY:CHANnel to copy ALL settings from one channel to another.

**Parameters**

- `<ch>` Channel number to copy mechanical settings to. If unspecified, value is set to 1.
- `<num>` Channel number to copy mechanical settings from.

**Examples**

'Copies mechanical settings from chan 2 to chan 1.'

`SENS1:PATH:CONF:COPY 2`

**Return Type** Not Applicable

**Default** Not Applicable

SENSe:PATH:CONF:DELe<te <string>
**Applicable Models:** N522xB, N523xB, N524xB

**(Write-only)** Deletes the specified configuration name from the VNA. The factory configurations cannot be deleted. This is the only method of distinguishing a factory configuration from a user-named configuration.

**Parameters**

- `<string>` Configuration name to be deleted.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:PATH:CONF:DEL &quot;MyMixer&quot;</td>
<td>Delete configuration &quot;MyMixer&quot;</td>
</tr>
</tbody>
</table>

**Return Type** Not Applicable

**Default** Not Applicable

---

**SENS:**PATH:CONF:DTEx <string>

**Applicable Models:** N522xB, N523xB, N524xB

**(Read-Write)** Write and read descriptive text associated with the configuration. This text is displayed in the path configuration dialog. Text is generally used to describe external connections that must be made manually to complete the configuration setup.

**Parameters**

- `<string>` Descriptive text enclosed in quotes. Double quotes are not allowed within the descriptive text.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:PATH:CONF:DTEX &quot;Connect J1 jumper on the rear panel.&quot;</td>
<td>Set descriptive text</td>
</tr>
</tbody>
</table>

**Query Syntax**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS&lt;ch&gt;:PATH:CONF:DTEx?</td>
<td>Query descriptive text</td>
</tr>
</tbody>
</table>

**Return Type** String

**Default** Not Applicable

---

**SENS<ch>:PATH:CONFig:ELEMent:CATalog?**
**Applicable Models:** N522xB, N523xB, N524xB

*(Read-only)* Returns the names of configurable elements as a comma-delimited list of strings.

See a list of configurable elements and settings for various VNA models.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.

**Examples**

```
SENS:PATH:CONF:ELEM:CAT?
returns
"Combiner", "Src1", "Src2"
```

**Default** Not Applicable

**SENSe<ch>:PATH:CONFig:ELEMent[:STATE] <elem>, <setting>**

**Applicable Models:** N522xB, N523xB, N524xB, M9485A with M9341B, M937xA with M9341B, P937xA, M980xA and P500xA, E5080B

*(Read-Write)* Write or read the setting of a specified element in the current configuration.

This command is used for both RF path and IF path configuration.

- RF Configuration elements and values
- IF Configuration elements and values (Includes IF Path, Pulse, and DSP elements)

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<elem>` Name of the element for which a setting is to be made.

  "Disable" or "Enable"

**Examples**

```
SENS:PATH:CONF:ELEM "Combiner","Normal"
```

**Query Syntax**

```
SENSe<ch>:PATH:CONFig:ELEMent? "Combiner"
```

Returns the current state of the Combiner element.

**Return Type** String
SENSe<ch>:PATH:CONFig:ELEMent:VALue:CATalog? <element>

**Applicable Models:** N522xB, N523xB, N524xB

*(Read-only)* Returns the list of valid settings that can be used with the specified element.

See a list of configurable elements and settings for various VNA models.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<element>` String. Element name for which to return valid settings.

**Examples**

```plaintext
returns "Normal", "Reversed"
```

**Default** Not Applicable

---

SENSe<ch>:PATH:CONFig:NAME?

**Applicable Models:** N522xB, N523xB, N524xB

*(Read-only)* Returns the name of the current configuration only if NO individual element settings had been changed since selecting or storing a configuration. When element settings change, the path configuration name is cleared.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.

**Examples**

```plaintext
SENS:PATH:CONF:NAME?
returns "Default"
```

**Return Type** String

**Default** Not Applicable

---

SENSe<ch>:PATH:CONFig:SELect <string>
Applicable Models: N522xB, N523xB, N524xB

(Write only) Loads the named configuration onto the specified channel.

**Note:** Loading a stored configuration will over-write MANY RF and IF path configuration settings. Make your measurement settings AFTER recalling a stored configuration, NOT before.

Use `SENS:PATH:CONF:CAT?` to return the configuration names that are stored on the VNA.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<string>` Configuration name. "Default" is the default factory configuration.

**Examples**

```
SENS:PATH:CONF:SEL 'default'
sense2:path:CONF:select "MyMixer"
```

**Query Syntax**

Not Applicable

**Default**

"Default"

---

`SENSe<ch>:PATH:CONFig:STORe <name>`

Applicable Models: N522xB, N523xB, N524xB

(Write only) Saves the path configuration currently associated with channel `<ch>` to the specified configuration name.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<name>` String. Configuration name. Factory configurations can NOT be overwritten. Specifying the name of a pre-defined factory configuration will result in an error.

**Examples**

```
SENS:PATH:CONF:STOR "MyMixer"
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable
Phase Noise Commands

Defines the settings for phase noise measurements. Option S93031xB is required and applies ONLY to instruments with serial prefix 6021 and above.

```
SENSe:PN
ADJ ust
    | CONFigure
    | FREQuency
    | CHECK
    | LIMit
    | HIGH
    | LOW
    | SEARch
    | [:STATe]
        | LEVel
        | THReshold
BWIDth
    | [:RESolution]
        | RATio
FAVerage
    | FACTor
NTYPe
```
Click a keyword to view the command details.

See Also

- Calibrating with SCPI
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

```
SENSe<ch>:PN:ADJust:CONFigure:FREQuency:CHECK <bool>
```
Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Enables and disables check for carrier. This is a narrow band search that expects the carrier to exist around the user-specified carrier frequency.

Parameters
<ch> Channel number of the measurement (optional).
<bool> Choose from:

0 - OFF - Disable check for carrier.
1 - ON - Enable check for carrier.

Examples
SENS:PN:ADJ:CONF:FREQ:CHEC ON
sense2:pn:adjust:configure:frequency:check ON

Query Syntax
SENSe<ch>:PN:ADJust:CONFigure:FREQuency:CHECk?
Return Type
Boolean
Default
ON

SENSe<ch>:PN:ADJust:CONFigure:FREQuency:LIMit:HIGH <value>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the high frequency limit to use during a broadband carrier search. The SENSe:PN:ADJust:CONFigure:FREQuency:SEARch[:STATe] command enables/disables the carrier search.

Parameters
<ch> Channel number of the measurement (optional).
(value> Carrier search high frequency limit.

Examples
sense2:pn:adjust:configure:frequency:limit:high 1e9

Query Syntax
SENSe<ch>:PN:ADJust:CONFigure:FREQuency:LIMit:HIGH?
Return Type
Numeric
Default
1 GHz

SENSe<ch>:PN:ADJust:CONFigure:FREQuency:LIMit:LOW <value>
**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

*(Read-Write)* Sets and returns the low frequency limit to use during a broadband carrier search. The `SENSe:PN:ADJust:CONFigure:FREQuency:SEARch[:STATe]` command enables/disables the carrier search.

**Parameters**

- `<ch>` Channel number of the measurement (optional).
- `<value>` Carrier search low frequency limit.

**Examples**

- `sense2:pn:adjust:configure:frequency:limit:high 1e6`

**Query Syntax**

`SENSe<ch>:PN:ADJust:CONFigure:FREQuency:LIMit:LOW?`

**Return Type**

Numeric

**Default**

1 MHz

---

**SENSe<ch>:PN:ADJust:CONFigure:FREQuency:SEARch[:STATe] <bool>**

*(Read-Write)* Enables and disables a broadband carrier search within the range specified using the `SENSe:PN:ADJust:CONFigure:FREQuency:LIMit:LOW` and `SENSe:PN:ADJust:CONFigure:FREQuency:LIMit:HIGH` commands.

**Parameters**

- `<ch>` Channel number of the measurement (optional).
- `<bool>` Choose from:
  - 0 - OFF - Disable search.
  - 1 - ON - Enable search.

**Examples**

- `SENSe:PN:ADJ:CONF:FREQ:SEAR ON`
- `sense2:pn:adjust:configure:frequency:search:state ON`

**Query Syntax**

`SENSe<ch>:PN:ADJust:CONFigure:FREQuency:SEARch[:STATe]?`

**Return Type**

Boolean

**Default**

ON

---

**SENSe<ch>:PN:ADJust:CONFigure:LEVel:THReshold <value>**

4507
Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the threshold to use during a carrier search.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<value>` Carrier search threshold in dBm.

Examples

```
SENS:PN:ADJ:CONF:LEV:THR -20
sense2:pn:adjust:configure:level:threshold -20
```

Query Syntax

```
SENSe<ch>:PN:ADJust:CONFigure:LEVel:THReshold?
```

Return Type

Numeric

Default

-20 dBm

SENSe<ch>:PN:BWIDth[:RESolution]:RATio <value>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the resolution bandwidth ratio, which is the specified resolution bandwidth percentage of every half decade offset frequency.

Example:

Start Offset = 1 kHz

Stop Offset = 100 kHz

RBW Ratio = 10%

1 kHz - 3 kHz: RBW = 100 Hz (10% of 1 kHz)

3 kHz - 10 kHz: RBW = 300 Hz (10% of 3 kHz)

10 kHz - 30 kHz: RBW = 1 kHz (10% of 10 kHz)

30 kHz - 100 kHz: RBW = 3 kHz (10 % of 30 kHz)

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<value>` Resolution bandwidth ratio in %.
SENSe<ch>:PN:FAVerage:FACTor <value>

**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

*(Read-Write)* Sets and returns the FFT average factor number. The average factor is multiplied by the default average count for each frequency range. The default average count of the lower frequency range is 1 and at the higher offset frequency range is a larger count.

**Parameters**

- `<ch>`: Channel number of the measurement (optional).
- `<value>`: FFT average factor number from 1 to 10,000.

**Examples**

```
SENS:PN:FAV:FACT 10
sense2:pn:faverage:factor 10
```

**Query Syntax**

SENSe<ch>:PN:FAVerage:FACTor?

**Return Type**

Numeric

**Default**

1

SENSe<ch>:PN:NTYPE <enum>
**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the noise type to phase or residual noise.

**Parameters**

- `<ch>`  Channel number of the measurement (optional).
- `<enum>`  (Enumeration) Choose from:

  - **PNOise** - Phase noise measurement.
  - **RESidual** - Residual (additive) noise measurement.

**Examples**

```
SENS:PN:NTYP PNO
sense2:pn:ntype residual
```

**Query Syntax**  
`SENSe<ch>:PN:NTYPe?`

**Return Type**  
Enumeration

**Default**  
PNOise

---

**SENSe<ch>:PN:RECeiver <rcvr>**

**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the receiver for the phase noise measurement or receiver ratios for residual phase noise measurements.

**Parameters**

- `<ch>`  Channel number of the measurement (optional).
- `<rcvr>`  (String) Receiver string.

Choose from:

- a1, a2, a3, a4, b1, b2, b3, b4, b2/a1, b1/a2.

**Note:** The b2/a1 and b1/a2 ratios are for residual phase noise measurements. For example, b2/a1 measures the additive phase noise at b2 relative to a1.

**Examples**

```
SENS:PN:REC "b2"
sense:pn:receiver "b2"
```

**Query Syntax**  
`SENSe<ch>:PN:RECeiver?`

**Return Type**  
String
### SENSe<ch>:PN:RESidual:INPut <dutInput>

**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns receiver at the DUT input for residual phase noise measurements.

**Parameters**

- `<ch>` Channel number of the measurement (optional).
- `<dutInput>` *(String)* Receiver string for DUT input.

Choose from:

a1, a2, a3, a4, b1, b2, b3, b4.

**Examples**

```
SENS:PN:RES:INP "a1"
sense:pn:residual:input "a1"
```

**Query Syntax**

`SENSe<ch>:PN:RESidual:INPut?`

**Return Type** String

**Default** a1

### SENSe<ch>:PN:RESidual:OUTPut <dutOutput>

**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns receiver at the DUT output for residual phase noise measurements.

**Parameters**

- `<ch>` Channel number of the measurement (optional).
- `<dutOutput>` *(String)* Receiver string for DUT output.

Choose from:

a1, a2, a3, a4, b1, b2, b3, b4.

**Examples**

```
SENS:PN:RES:OUT "b2"
sense:pn:residual:output "b2"
```

**Query Syntax**

`SENSe<ch>:PN:RESidual:OUTPut?`

**Return Type** String

**Default** b2
SENSe<ch>:PN:SWEep:CARRier:FREQuency <value>

**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

*(Read-Write)* Sets and returns the carrier frequency.

**Parameters**

- **<ch>** Channel number of the measurement (optional).
- **<value>** Carrier frequency.

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```plaintext
SENSe<ch>:PN:SWEep:CARRier:FREQ 1 GHz
sense2:pn:sweep:carrier:frequency max
```

**Query Syntax**

SENSe<ch>:PN:SWEep:CARRier:FREQuency?

**Return Type**

Numeric

**Default** 1 GHz

---

SENSe<ch>:PN:SWEep:NOISe:MODE<enum>

**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

*(Read-Write)* Sets and returns the sweep noise mode.

**Parameters**

- **<ch>** Channel number of the measurement (optional).
- **<enum>** *(Enumeration)* Choose from:

  - **FAST** - Fastest measurement speed with lowest accuracy.
  - **NORMal** - Between fastest and slowest measurement speed and accuracy.
  - **BEST** - Slowest measurement speed with highest accuracy.
### Examples

- `SENS:PN:SWE:NOIS:MODE NORM`
- `sense2:pn:sweep:noise:mode fast`

### Query Syntax

SENSe<ch>:PN:SWEep:NOISe:MODE?

### Return Type

Enumeration

### Default

NORMal
This command is superseded by the `SOURce:POWer:ATTenuation:RECeiver:TEST` command.

Learn about Receiver Attenuation

`SENS<cnm>:POWer:ATTenuator <rcvr>,<num>`

**Applicable Models:** N522xB, N523xB, N524xB

*(Read-Write)* Sets the attenuation level for the specified receiver.

**Note:** Attenuation cannot be set with Sweep Type set to Power

**Note:** For M980xA/P50xxA/E5080B Receiver Gain, use `SENS:SOUR:REC:GAIN`.

**Parameters**

- `<cnm>` Any existing channel number. If unspecified, value is set to 1
- `<rcvr>` Receiver to get attenuation. Choose from:
  - `ARECeiver` - receiver A
  - `BRECeiver` - receiver B
  - `CRECeiver` - receiver C
  - `DRECeiver` - receiver D

Receiver attenuation can NOT be set using logical receiver notation.

- `<num>` Attenuation value in dB. To determine how many receiver attenuators, the maximum receiver attenuation, and attenuation step size, for a VNA model, see [VNA Models and Options](#). If a number other than these is entered, the analyzer will select the next lower valid value. For example, if 19 is entered for the E8361A/C, then 10 dB attenuation will be selected.

**Examples**

```
SENS:POW:ATT AREC,10
sense2:power:attenuator breceiver,30
```

**Query Syntax**

```
SENS<cnm>:POWer:ATTenuator? <rec>
```

**Return Type**

Numeric

**Default**

0
Sense: Pulse Commands

Configures the 5 pulse generators in the PNA-X.

Beginning with A.09.50, these commands can also be used to control an external Pulse Generator.

Learn more.

SENSe:PULSe
| CATalog?
| DELay
| DINCrement
| HDELay
| ADC?
| MODulator
| [:STATe]
| INVert
| MTIMing
| DEVice
| PULSe4:
| MODE
| OPTion
| PERiod
| STATE
| SUBPointtrig
| TPOLarity
| TTYPE
| WIDTh

Click on a keyword to view the command details.
To make other Pulse settings, such as enabling the internal pulse modulators, use this command:

- `SENSe<ch>:PATH:CONFig:ELEMent[:STATe] <elem>, <setting>`
- At that command help topic, click the **IF Configuration elements and values** link to see the Pulse element and setting (in the middle box) to configure.

**See Also**

- `SENS:SWEep:PULSE` - configures the channel for pulse measurements
- External Pulse Generator configuration commands
- `SENS:IF` configuration commands
- Example Programs
- Integrated Pulse Application
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

### Pulse Definitions

![Pulse Definitions](image)

- $D =$ Delay; the time before each pulse begins
- $W =$ Width; the time the pulse is ON
- $P =$ Period; one complete pulse cycle
- Duty Cycle $= \frac{W}{P}$

**Important:** If $D + W$ is greater than $P$, then undefined VNA behavior results. There is NO error message or warning.

**SENSe:PULSe:CATalog?**
**Applicable Models:** All with Pulsed RF Measurement Option

*(Read-only)* Returns the string names of internal and configured external pulse generators.

**Parameters**

| None |

**Examples**

| SENS:PULS:CAT? |

**Default**

| Not Applicable |

---

**SENSe<ch>:PULSe<n>:DELay <value>[,<name>]**

**Applicable Models:** All with Pulsed RF Measurement Option

*(Read-Write)* Sets the pulse delay. The amount of time before a new pulse begins.

*See Pulse Definition diagram.*

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Internal pulse generator number. Choose from 0 to 4.
  
  0 is the generator that pulses the ADC.
- `<value>` Delay value in seconds. Choose a value from about 33ns to about 70 seconds.
- `<name>` Optional. String name of the pulse generator.
  
  Required for use with *external pulse generators*.

Use **SENSe:PULSe:CAT?** to return the names of configured pulse generators.

If specified, `<n>` is ignored.

If unspecified, `<n>` is required for internal pulse generators.

**Examples**

| SENS:PULS1:DEL .5 |
| SENS:PULS:DEL .5, "My81110" |

**Query Syntax**

| SENSE<ch>:PULSe<n>:DELay? [,<name>] |

**Return Type**

| Numeric |

**Default**

| 0 |

---

**SENSe<ch>:PULSe<n>:DINCrement <value>[,<name>]**
**Applicable Models:** All with Pulsed RF Measurement Option (Except M980xA, P50xxA, E5080B)

(Read-Write) Sets the pulse delay increment. The delay increments with each pulse by the `<value>` amount.

For example, in this diagram the delay starts as 1. On the second pulse, delay=2. On the third pulse, delay=3.

**Important:** If \( D + W \) is greater than \( P \), then undefined VNA behavior results. There is NO error message or warning. Delay includes the incremented value.

This is useful for pulse profiling.

![Diagram](image)

See Pulse Definition diagram.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Pulse generator number. Choose from 0 to 4.
  - 0 is the generator that pulses the ADC.
- `<value>` Delay increment value in seconds.
- `<name>` Optional. String name of the pulse generator.
  - Required for use with external pulse generators.
  - Use `SENSe:PULSe:CAT?` to return the names of configured pulse generators.

If specified, `<n>` is ignored.

If unspecified, `<n>` is required for internal pulse generators.

**Examples**

```
SENS:PULS1:DINC .5
SENS:PULS:DINC .5, "My81110"
```

**Query Syntax**

`SENSe<ch>:PULSe<n>:DINCrement? [<name>]`

**Return Type**

Numeric

**Default**

0
SENSe:PULSe<n>:HDELay:ADC?

**Applicable Models:** All with Pulsed RF Measurement Option

(Read-only) Returns the ADC delay for pulse measurements.

**Parameters**

<n> Internal pulse generator number. Choose from 1 to 4.

0 is the generator that pulses the ADC.

**Examples**

```
SENS : PULS : HDEL : ADC?
```

**Default** Not Applicable

SENSe<ch>:PULSe<n>:HDELay:MODulator <value>

**Applicable Models:** All with Pulsed RF Measurement Option

(Read-Write) Sets the time lag between the pulse drive signal and the actual RF output.

The following diagram shows Pulse1 used as the pulse drive signal.
Parameters

chw> Any existing channel number; if unspecified, value is set to 1.

<\n> Internal pulse generator number. Choose from 1 to 4.

0 is the generator that pulses the ADC.

<\text{value}> Delay value in seconds.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{SENSe}&lt;\text{ch}&gt;\text{:PULSe}&lt;\text{n}&gt;\text{:HDELay:MODulator}\text{?}$</td>
<td>Numeric</td>
<td>50 ns</td>
</tr>
</tbody>
</table>

SENSe<ch>:PULSe<n>:HDELay[:STATe] <value>

Applicable Models: All with Pulsed RF Measurement Option

(Read-Write) Enables/disables modulator and ADC delays for pulse measurements.

Parameters

chw> Any existing channel number; if unspecified, value is set to 1.

<\n> Pulse generator number. Choose from 0 to 4.

0 is the generator that pulses the ADC.

<\text{value}> Boolean

ON (or 1) - turns delays ON.

OFF (or 0) - turns delays OFF.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{SENSe}&lt;\text{ch}&gt;\text{:PULSe}:\text{HDELay[:STATe]}? [&lt;\text{name}&gt;]$</td>
<td>Boolean</td>
<td>OFF</td>
</tr>
</tbody>
</table>

SENSe<ch>:PULSe<n>:INVert <value>[,<name>]

4521
**Applicable Models:** All with Pulsed RF Measurement Option

*(Read-Write)* Sets whether to invert the polarity of the pulse.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Pulse generator number. Choose from 0 to 4. 0 is the generator that pulses the ADC.
- `<value>` Boolean
  - ON (or 1) - Invert the pulse generator polarity. This causes the pulse ON time to be active low and OFF be active high.
  - OFF (or 0) - Do NOT Invert the pulse generator polarity.
- `<name>` Optional. String name of the pulse generator. Required for use with external pulse generators.

Use `SENSe:PULSe:CAT?` to return the names of configured pulse generators.

If specified, `<n>` is ignored.

If unspecified, `<n>` is required for internal pulse generators.

**Examples**

```
SENSe:PULS1:INV 1
SENSe:PULS:INV 1, "My81110"
```

**Query Syntax**

`SENSe<ch>:PULSe<name>:INVert?` [optional `<name>`]

**Return Type**

Boolean

**Default**

OFF (0)

```
SENSe<ch>:PULSe<name>:MTIMing:DEVice <device>[,<name>]
```
**Applicable Models:** All with Pulsed RF Measurement Option and Modulation Distortion Option

*(Read-Write)* Sets and reads the device being controlled by the pulse generator output.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Pulse generator number. Choose from 1 to 4.
  
  0 is the generator that pulses the ADC.
- `<device>` Pulse Device. Choose from:
  
  - **ADCTrigger** - (Pulse0 only) Pulse 0 is used to trigger the ADC.
  - **RFMOdul** - Indicates that the pulse signal is used to drive the RF modulator. Only one pulse generator output can be used to drive an RF source. If you try to set more than one pulse generator output to **RFMOdul**, then the other one will be set to **UserN** (where "N" is the pulse generator number).
  - **ADCActivity** - (Pulse4 only) Pulse4 can also be set to monitor ADC activity. This selection outputs a signal on Pulse4 when the ADC is active. This is the same as **SENSe:PULSe4:OPTion**.
  - **USR1** through **USR4** - Labels for user convenience. These labels do not connect the pulse generator to any specific hardware. These selections may be used to control a DUT, DC biases, or other signals.
- `<name>` Optional. String name of the pulse generator.
  
  Required for use with external pulse generators.
  
  Use **SENSe:PULSe:CAT?** to return the names of configured pulse generators.

If specified, `<n>` is ignored.

If unspecified, `<n>` is required for internal pulse generators.

**Examples**

```
SENSe:PULS:MTIM:DEV RFMOdul
```

**Query Syntax**

```
SENSe<ch>:PULSe:MTIMing:DEVice?
```

**Return Type**

Character

**Default**

RFMOdul
Applicable Models: N522xB, N524xB

(Read-Write) Sets the specific PULSe4 behavior to either “ALL” ADC ACTivity or “TRACe" ADC Activity, WHEN Pulse4 has been set to display ADC activity using the SENSE:PULSe4:OPTion command.

“ALL” ADC Activity enables PULSe4 to indicate all ADC activity, even ADC measurements that are not displayed as final trace data. An example are background measurements used for receiver leveling, but that are never displayed on a trace.

“TRACe” ADC Activity enables PULSe4 to indicate only the ADC activity that ends up displayed in a VNA trace.

Parameters

<ch> Any existing channel number; if unspecified, value is set to 1.
<char> ADC activity. Choose from:

ALL - Pulse 4 output pin indicates all ADC activity.

TRACe - Pulse 4 output pin indicates ADC being used for final measurements.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:PULS4:MODE TRACe</td>
<td>Character</td>
</tr>
</tbody>
</table>

SENSe<ch>:PULSe4:OPTion <bool>

Applicable Models: All with Pulsed RF Measurement Option

(Read-Write) Turns pulse4 output ON and OFF. Enable pulse4 to use an oscilloscope connected to pin 13 of the PULSE I/O connector on the rear panel of the VNA to display when the ADC is making measurements.

Note: The pulse output must be on using SENSE:PULSe4[:STATE] ON to view ADC activity.

Parameters

<ch> Any existing channel number; if unspecified, value is set to 1.
<bool> Choose from:

ON (or 1) - Pulse 4 output pin indicates ADC activity.

OFF (or 0) - Pulse 4 output pin indicates legacy behavior (pulse generator number 4 output).

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:PULS4:OPT 1</td>
<td>Character</td>
</tr>
</tbody>
</table>
SENSe<ch>:PULSe:PERiod <value>[,<name>]

**Applicable Models:** All with Pulsed RF Measurement Option

(Read-Write) Sets the pulse-period (1/PRF) for ALL pulse generators.

The resolution of the period is:

DSP version: **4.0** = 16.667nS.

DSP version: **5.0** = 10nS

Learn more about DSP version.

See Pulse Definition diagram.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<value>` Pulse period in seconds. Choose a value from about 33ns to about 70 seconds.
- `<name>` Required for use with an external pulse generator.
  
  String name of the external pulse generator.

  If unspecified, the period for the internal pulse generators are set.

  Use `SENSe:PULSe:CAT?` to return the names of configured pulse generators.

**Examples**

```
SENS:PULS:PERiod .05
SENS:PULS:PER .01, "My81110"
```

**Query Syntax**

`SENSe<ch>:PULSe<ch>:PULSe:PERiod? [,<name>]`

**Return Type** Numeric

**Default** 1e-3 sec
**Applicable Models:** All with Pulsed RF Measurement Option

*(Read-Write)* Turns the pulse output ON and OFF.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Pulse generator number. Choose from 0 to 4.
  
  0 is the generator that pulses the ADC.
- `<value>` Boolean
  
  ON (or 1) - turns pulse output ON.
  
  OFF (or 0) - turns pulse output OFF.
- `<name>` Optional. String name of the pulse generator.
  
  Required for use with external pulse generators.
  
  Use `SENSe:PULSe:CAT?` to return the names of configured pulse generators.

  If specified, `<n>` is ignored.
  
  If unspecified, `<n>` is required for internal pulse generators.

**Examples**

```
SENSe:PULS1 1
SENSe:PULS 1, "My81110"
```

**Query Syntax**

`SENSe<ch>:PULSe<name>:STATe`?

**Return Type**

Boolean

**Default**

OFF

`SENSe<ch>:PULSe<n>:SUBPointtrig` <bool>
**Applicable Models:** All with Pulsed RF Measurement Option (Except M980xA, P50xxA, E5080B)

(Read-Write) Enables / Disables subpoint triggering. When enabled and performing **Point Averaging**, each rising edge of P0 triggers a subpoint (one of N acquisitions in an N point average). Must also enable the P0 generator using **SENS:PULS0:STAT**.

Learn more about the PNA-X pulse generators.

### Parameters

- `<ch>`: Any existing channel number; if unspecified, value is set to 1.
- `<n>`: Pulse generator number. **Must be 0** as this is the generator that triggers the ADC.
- `<bool>`: ON (or 1) - turns subpoint triggering ON.
  
  OFF (or 0) - turns subpoint triggering OFF.

### Examples

```
SENS:PULS0:SUBP 1
```

### Query Syntax

```
SENSe<ch>:PULSe0:SUBPointtrig?
```

### Return Type

Boolean

### Default

OFF

---

**SENS<ch>:PULSe:TPOLarity `<char>`**

**Applicable Models:** All with Pulsed RF Measurement Option

(Read-Write) Sets the polarity of the trigger signal to which the internal pulse generators will respond when being externally triggered at the PulseSyncIn pin.

**Note:** This feature requires DSP version: **4.0 FPGA: 34** or higher. [Learn more.](#)

Learn more about the PNA-X pulse generators.

### Parameters

- `<ch>`: Any existing channel number; if unspecified, value is set to 1.
- `<char>`: Pulse polarity. Choose from:
  
  **POSitive** - VNA responds to rising edge or HIGH level

  **NEGative** - VNA responds to falling edge or LOW level.

  Set Edge or Level triggering using **SENS:PULS:TTYPe**.

### Examples

```
SENS:PULS:TPOL NEG
```
SENSe<ch>:PULSe<n>:TTYPe <char>

Applicable Models: All with Pulsed RF Measurement Option (Except M980xA, P50xxA, E5080B)

(Read-Write) Sets the type of trigger signal to which the internal pulse generators will respond when being externally triggered at the PulseSyncIn pin.

Note: This feature requires DSP version: **4.0 FPGA: 34** or higher. [Learn more](#).

Learn more about the PNA-X pulse generators.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<char>` Trigger type. Choose from:
  - **EDGE** - VNA responds to the edge (rising or falling) of a signal
  - **LEVEL** - VNA responds to the level (HIGH or LOW) of a signal

Set polarity using `SENS:PULS:TPOL`

**Examples**

```
SENS:PULS:TTYP EDGE
```

**Query Syntax**

SENSe<ch>:PULSe<n>:TTYPe?

**Return Type** Character

**Default** LEVel - Also the type used when the PNA-X does not have the required DSP hardware.

**SENSe<ch>:PULSe<n>:WIDTh <value> [,.<name>]**
**Applicable Models:** All with Pulsed RF Measurement Option

*(Read-Write)* Sets the pulse width. The amount of time that the pulse is ON.

See Pulse Definition diagram.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Pulse generator number. Choose from 0 to 4.
  
  0 is the generator that pulses the ADC.
- `<value>` Pulse width in seconds. Choose a value from about 33ns to about 70 seconds.
- `<name>` Optional. String name of the pulse generator.

Required for use with external pulse generators.

Use `SENSe:PULSe:CAT?` to return the names of configured pulse generators.

If specified, `<n>` is ignored.

If unspecified, `<n>` is required for internal pulse generators.

**Examples**

- `SENS:PULS.WIDT .5`
- `SENS:PULS.WIDT .5, "My81110"`

**Query Syntax**

- `SENSe<ch>:PULSe<n>:WIDTH? [<name>]`

**Return Type**

- Numeric

**Default**

- `1e-4 sec`
SENSe:ROSCillator:CONTrol:AUTO <bool>

Applicable Models: N522xB, N524xB

(Read-Write) Enables selecting between locking to an external reference without the ability to change the reference frequency, or select internal or external reference with the ability to change the reference frequency.

Parameters

<bool>

ON (or 1) This is the default behavior which locks to an external reference without the ability to change the reference frequency.

OFF (or 0) Leaves the reference in its current state and all changes to the reference behavior must be made through the SCPI commands in this topic or by using the Reference dialog.

Examples

SENSe:ROSCillator:EXTernal:FREQuency
Applicable Models: M980xA, P50xxA/B, P93xxB, E5080B, N522xB, N524xB

(Read-Write) Set and read Reference Oscillator frequency at the reference input connector. 100 MHz reference is typical for M980xA, P50xxA/B, P93xxB E5080B.

**Parameters**

<num> 1E+7 or 1E+8

1E+7: accepts 10 MHZ reference
2E+7: accepts 20 MHZ reference
1E+8: accepts 100 MHz reference

**Examples**

SENSe:ROSCillator:EXT:FREQ 1E7
sense:roscillator:external:frequency 1E7

**Query Syntax**

SENSe:ROSCillator:EXT:FREQ?

**Return Type** Numeric

**Default** 1E+7

---

SENSe:ROSCillator:OUTPut:FREQuency

Applicable Models: M980xA, P50xxA/B, P93xxB, E5080B, N522xB, N524xB

(Read-Write) Set and read Reference Oscillator frequency outputted from the reference output connector. 100 MHz reference is typical for M980xA, P50xxA, E5080B.

**Parameters**

<num> 1E+7 or 1E+8

1E+7: outputs 10 MHz reference.
1E+8: outputs 100 MHz reference

**Examples**

SENSe:ROSCillator:OUTPut:FREQuency 1E8
sense:roscillator:output:frequency 1E8

**Query Syntax**

SENSe:ROSCillator:OUTPut:FREQuency?

**Return Type** Numeric

**Default** 1E+7
**SENSe:ROSCillator:SOURce?**

Applicable Models: M980xA, P50xxA/B, P93xxB, N522xB, N523xB, N524xB, E5080A/B

(Read-only) Applying a signal to the Reference Oscillator connector automatically sets the Reference Oscillator to EXTernal. This command allows you to check that it worked.

- **EXTERNal** is returned when a signal is present at the Reference Oscillator connector.
- **INTERNAL** is returned when NO signal is present at the Reference Oscillator connector.

<table>
<thead>
<tr>
<th>Examples</th>
<th>SENS:ROSC:SOUR?</th>
<th>sense:roscillator:source?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Character</td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
<td></td>
</tr>
</tbody>
</table>

**SENSe:ROSCillator:SOURce <state>**

Applicable Models: M937xA, P93xxA/B, M980xA, P50xxA/B

(Write-only) Set and read the Reference Oscillator state.

**Note:** This setting is NOT cleared with Preset. However, it does clear when the M937xA software is restarted.

**Note:** Using the signal from PXI Chassis backplane degrades the measurement performance especially absolute measurement. This is due to worse phase noise on the reference signal from backplane.

**Parameters**

- **<state>** Choose from the following:
  - INTernal - Use the internal Reference Oscillator.
  - EXTernal - Use an external Reference Oscillator. Use SENSE:ROSCillator:SOURce:CONDition? to determine if the M937xA is locked to the external oscillator.
  - PXIBackplane - Use an external Reference Oscillator via PXI Chassis backplane (PXI only)

| Examples  | SENS:ROSC:SOUR INT | sense:roscillator:source external |
SENSe:ROSCillator:SOURce:CONDition?

**Applicable Models:** M937xA, P93xxB, M980xA, P50xxA/B, N522xB, N524xB

(Read-only) Reads the Reference Oscillator 'locked' condition.

When SENS:ROSC:SOUR is set to Internal, this command will always return "LOCKed".

When SENS:ROSC:SOUR is set to External, then this function takes about 100 usec to read the state of the hardware.

**Examples**

```
SENSe:ROSCillator:COND?
sense:roscillator:source:condition?
```

**Return Type** Character

**Default** Not applicable
Spectrum Analyzer Commands

Controls the Spectrum Analyzer Application.

SA Application - SA Setup tab

Source Setup tab
Coherence Setup tab (Not applicable for M98x0A, P50xxA)
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:MULTitone</td>
<td>Enables multi-tone mode</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:MULTitone</td>
<td>Enables multi-tone mode</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:VECTor:AVERage[:STATE]</td>
<td>Enables vector averaging</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:VECTor:AVERage:VALUE</td>
<td>Enables vector averaging</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:MULTitone:VALID</td>
<td>Enables multi-tone validation</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PHASe:DISPLAY:LEVEL</td>
<td>Enables phase display level</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:DISTortion:FREQuency:TUNE:IMMediate</td>
<td>Enables distortion frequency tuning immediately</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PHASe[:STATE]</td>
<td>Enables phase state</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:COUNt</td>
<td>Enables pulse search count</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:DUTY:TOLerance</td>
<td>Enables pulse search duty tolerance</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:EXECute</td>
<td>Enables pulse search execution</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:INITialize</td>
<td>Enables pulse search initialization</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:ITEM:COUNt?</td>
<td>Enables pulse search item count query</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:ITEM:SELect</td>
<td>Enables pulse search item selection</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:PERiod:TOLerance</td>
<td>Enables pulse search period tolerance</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:PRIority</td>
<td>Enables pulse search priority</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:WIDTh:TOLerance</td>
<td>Enables pulse search width tolerance</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:PERiod[:VALUE]</td>
<td>Enables pulse search period value</td>
</tr>
</tbody>
</table>

**Advanced Setup tab**
SENSe:SA:BANDwidth:SHAPE
SENSe:SA:IMAGE:REJect
SENSe:SA:IMAGE:STRENgth
SENSe:SA:BANDwidth:VIDeo:RATio
SENSe:SA:FREQuency:SPAN:BAN
SENSe:SA:BANDwidth:SEARch:OC
SENSe:SA:SOURce:DC:SWEep[:STATe]
SENSe:SA:SOURce:DC:SWEep:POINt
SENSe:SA:FREQuency:TUNE:IMMediate
SENSe:SA:SOURce:DC:SWEep:FIRst[:DIMension]

SENSe:SA:ADC:FILTer
SENSe:SA:ADC:FILTer:AUTO
SENSe:SA:DFT:BANDwidth:AUTO
SENSe:SA:DFT:BANDwidth:NARRow
SENSe:SA:DFT:BANDwidth:NARRow
SENSe:SA:DFT:BANDwidth:WIDE
SENSe:SA:DFT:BANDwidth:WIDE
Trigger Setup tab

SENSe:SA:TRIGer:LEVel[:STATe]

SENSe:SA:TRIGer:LEVel:VALue

SENSe:SA:TRIGer:PERCounter[:STATe]

SENSe:SA:TRIGer:PERCounter:VALue

Processing Setup tab

SENSe:SA:DFT:TYPE

SENSe:SA:DATA:KEEP:FREQuencies

(see End of Sweep Processing)
SENSe:SA:DATA:KEEP:RECeivers (see End of Sweep Processing)
SENSe:SA:DATA:KEEP:RECeivers (see End of Sweep Processing)
SENSe:SA:DATA:KEEP:RECeivers
SENSe:SA:DATA:KEEP:SCALar:Gi
SENSe:SA:DATA:KEEP:[STATE] (see End of Sweep Processing)
SENSe:SA:DATA:KEEP:VECTor:Gi
SENSe:SA:TRACe:IMAGe:[STATE]
SENSe:SA:ADC:ACQTime?
SENSe:SA:LO:COUNt?
SENSe:SA:SPAN:BINS:COUNt?
SENSe:SA:DFT:RESolution?
SENSe:SA:DFT:RECord:SIZE?
SENSe:SA:ADC:RECord:SIZE:VAL

ADC & LO Setup tab

SENSe:SA:ADC:OVERload:COUNt
SENSe:SA:ADC:OVERload:LIST?
SENSe:SA:ADC:OVERload:PERCent
SENSe:SA:ADC:RANGe:PERCent[:MAXimum]
SENSe:SA:ADC:RANGe:PERCent:MINimum
SENSe:SA:ADC:RANGe:PERCent:RECeiver?
SENSe:SA:ADC:SAMPle:RATE
Other SA SCPI commands

SA Application - Setup tab
SENS:FREQ:STAR
SENS:FREQ:STOP
SENS:FREQ:CENT
SENS:FREQ:SPAN
SENS:FREQ:SPAN:FULL
SENS:FREQ:CENT:STEP:SIZE
SENS:FREQ:CENT:STEP:AUTO
SENS:FREQ:CW
SENS:SWE:POIN
SENS:POWer:ATT
SENS:PATH:CONFig:ELEM

IF Gain Settings

Modification Distortion App, LO Feedthru Monitor SCPI commands
**SA Marker Settings**

Sets and reads the bandwidth of the band density marker.

CALCulate:MEASure:SA:MARKer:BDENsity:BW
CALCulate:SA:MARKer:BDENsity:BW

Returns the band density level in dBm/Hz from the band density marker.

CALCulate:MEASure:SA:MARKer:BDENsity:DATA?
CALCulate:SA:MARKer:BDENsity:DATA?

Marker to SA
CALC:MEAS:MARKer:SET SA

Read Band Power
CALC:MEAS:SA:MARK:BPOWer:DATA?

Set and read Band Power
CALC:MEAS:SA:MARK:BPOWer:SPAN

Set Band Power State
CALC:MEAS:SA:MARK:BPOWer[:STATe]

Read occupied bandwidth center frequency
CALC:MEAS:SA:MARK:OCCBand:CENTer?

Set and read occupied bandwidth percentage of span
CALC:MEAS:SA:MARK:OCCBand:PERCent

Read the occupied bandwidth power.
CALC:MEAS:SA:MARK:OCCBand:POWer?
Read occupied bandwidth span
CALC:MEAS:SA:MARK:OCCBand:SPAN?

Set occupied bandwidth state
CALC:MEAS:SA:MARK:OCCBand[:STATE]

Sets and reads the state of the band density noise marker.
CALCulate:MEASure:SA:MARKer:BDENsity:NOISe[:STATE]
CALCulate:SA:MARKer:BDENsity:NOISe[:STATE]

Sets and reads the bandwidth of the band power density marker.
CALCulate:MEASure:SA:MARKer:BDENsity:POWer:BW
CALCulate:SA:MARKer:BDENsity:POWer:BW

Sets and reads the state of the band power density marker.
CALCulate:MEASure:SA:MARKer:BDENsity:POWer[:STATE]
CALCulate:SA:MARKer:BDENsity:POWer[:STATE]

Sets and reads the bandwidth of the band tone density marker.
CALCulate:SA:MARKer:BDENsity:TONE:BW

Sets and reads the state of the band tone
CALCulate:MEASure:SA:MARKer:BDENsity:TONE[:STATE]
CALCulate:SA:MARKer:BDENsity:TONE[:STATE]
density marker.
Sets and reads the spacing of the band tone density marker.

CALCulate:SA:MARKer:BDENsity:TONE:TSPacing

Sets and reads the frequency span used by Power Density to normalize the power.

CALCulate:MEASure:SA:MARKer:BDENsity:EQSPan
CALCulate:SA:MARKer:BDENsity:EQSPan

Other SA commands

- CALCulate:MEASure:DEFine - creates an SA measurement.
- SA Calibration uses the Guided Calibration commands.

See Also

- Example Program: Create an SA Measurement
- Learn about SA Application
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

End of Sweep Processing

The SENS:SA:DATA:KEEP commands keep the memory buffer of the last full sweep in memory for further processing. This is not done by default because keeping the whole data in memory requires large amounts of memory and processing (for example, in the case of wide span or low RBW).

Before the implementation of this function, raw data could be sent to a file (ascii or binary) or the fifo, as this can be done while sweeping with no need to keep the whole data in memory.
This option is still available.

With each sweep, the data buffers are filled, and are erased if a new speed is started. In other words, this feature works well in the logic of “Single” sweeps with a wait for OPC: (operation complete). You must ensure that a full sweep is in the buffers before pulling out raw data.

The receiver list can be either ALL, or a comma separated list of valid receivers.

One command defines a list of receivers and another command reads which receivers are currently kept in memory. Only the receivers currently defined for measurement traces can be kept in memory.

---

**SENS<ch>:SA:ADC:ACQTime?**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Returns the LO acquisition time which is the ADC Record Size x ADC Sampling Frequency (10 nsec or 40 nsec) x (1 + Stacking) x (Video Average.Coefficient).

**Parameters**

- `<ch>`: Channel number of the measurement. If unspecified, value is set to 1.

**Examples**

- `SENS:SA:ADC:ACQT?`
- `sense2:sa:adc:acqtime?`

**Default**

Not applicable

---

**SENS<ch>:SA:ADC:DITHer[:STATe] <bool>**
(Read-Write) Set and read the ON/OFF state of the dither setting

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Dither OFF.
  - **1 - ON** - Dither ON.

Learn about these settings.

**Examples**

```
SENS:SA:ADC:DITH 1
```

**Query Syntax**

```
SENSe<ch>:SA:ADC:DITH?
```

**Return Type** Boolean

**Default** 0

---

**SENSe<ch>:SA:ADC:FILTer <num>**

(Read-Write) Set and read the ADC filter cutoff frequency. The entered frequency value is rounded to the closest value supported by the VNA (11 MHz or 38 MHz).

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Choose 11 MHz or 38 MHz.

Learn about these settings.

**Examples**

```
SENS:SA:ADC:FILTer 11MHz
```

**Query Syntax**

```
SENSe<ch>:SA:ADC:FILTer?
```

**Return Type** Numeric

**Default** 11 MHz
SENSe<ch>:SA:ADC:FILTer:AUTO <bool>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read how the ADC filter is set. When ON, the ADC filter is set based on the start and stop frequencies and the ADC sampling frequency.

Parameters

<ch> Any existing SA channel. If unspecified, value is set to 0.
<bool> Choose from:

0 - OFF - ADC filter is set manually using SENSe:SA:ADC:FILT.

1 - ON - ADC filter is set automatically.

Learn about these settings.

Examples SENSe<ch>:SA:ADC:FILTer:AUTO ON

Query Syntax SENSe<ch>:SA:ADC:FILTer:AUTO?

Return Type Boolean

Default 1

SENSe<ch>:SA:ADC:MREC:PERiod <value>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the period to wait between ADC record chunks.

Parameters

<ch> Any existing SA channel. If unspecified, value is set to 1.
,value> Choose a value between 64 and 33554432

Learn about these settings.

Examples SENSe<ch>:SA:ADC:MREC:PER 256

Query Syntax SENSe<ch>:SA:ADC:MREC:PERiod?

Return Type Integer

Default 64
SENSe<ch>:SA:ADC:MREC:SIZE <value>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the size of the ADC record chunks.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<value>` Choose a value between 1 and (ADC record size / 2).

Learn about these settings.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:SA:ADC:MREC:SIZE 256</td>
<td>Integer</td>
<td>32</td>
</tr>
</tbody>
</table>

**SENSe:SA:ADC:MREC[:STATe] <bool>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the ON/OFF state of the multiple recording function. Multiple recording allows the ADC Record Size to be divided and acquired in smaller "chunks" and also to specify a wait period between these acquisitions.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - ADC record size "chunking" OFF.
  - **1 - ON** - ADC record size "chunking" ON.

Learn about these settings.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:SA:ADC:MREC 0</td>
<td>Boolean</td>
<td>0</td>
</tr>
</tbody>
</table>
SENSe<ch>:SA:ADC:OVERload:COUNt?

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the count of ADCs/receivers in overload status for the previous sweep. The count is cleared if a new sweep is started. Zero indicates no overload.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```
SENS:SA:ADC:OVER:COUN? 0
```

**Return Type** Integer

**Default** Not applicable

SENSe<ch>:SA:ADC:OVERload:LIST?

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the list of measured receivers in overload status for the previous sweep. The list is cleared if a new sweep is started.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```
SENS:SA:ADC:OVER:LIST? "b2,b4"
```

**Return Type** String

**Default** Not applicable

SENSe<ch>:SA:ADC:OVERload:PERCent <num>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

**(Read-Write)** Set and read the ADC overload threshold percentage.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` ADC overload threshold percentage.

**Examples**

```
SENS:SA:ADC:OVER:PERC 80
```

**Query Syntax**

```
SENSe<ch>:SA:ADC:OVERload:PERCent?
```

**Return Type** Numeric

**Default** 80

---

```
SENSe<ch>:SA:ADC:RANGe:PERCent[:MAXimum]?
```

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

**(Read-only)** Read the maximum ADC percentage range of all receivers in use. For example, if the ADC range is -8190 to +8190 and the current min and max ADC values are (for this sweep) -4000 and +4195, then the percentage range is 50%. An incorrect request will return 0.0 or 100.0 depending on the request.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```
SENS:SA:ADC:RANG:PERC:MAX?
```

**Return Type** Integer

**Default** Not applicable

---

```
SENSe<ch>:SA:ADC:RANGe:PERCent:MINimum?
```
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09x/A/B)

*(Read-only)* Read the minimum ADC percentage range of all receivers in use. For example, if the ADC range is -8190 to +8190 and the current min and max ADC values are (for this sweep) -4000 and +4195, then the percentage range is 50%. An incorrect request will return 0.0 or 100.0 depending on the request.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```
SENS:SA:ADC:RANG:PERC:MIN?
```

**Return Type**

Integer

**Default**

Not applicable

**SENSe<ch>:SA:ADC:RANGe:PERCent:RECeiver? <recName>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09x/A/B)

*(Read-only)* Read the ADC percentage range for the specified receiver. For example, if the ADC range is -8190 to +8190 and the current min and max ADC values are (for this sweep) -4000 and +4195, then the percentage range is 50%. An incorrect request will return 0.0 or 100.0 depending on the request.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<recName>` (String) Receiver name

**Examples**

```
SENS:SA:ADC:RANG:PERC:REC? "b2"
```

**Return Type**

Integer

**Default**

Not applicable

**SENSe<ch>:SA:ADC:RECord:SIZE:FORCe[:STATe] <bool>**
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the ADC record size mode.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:

  0 - OFF - ADC record size set automatically.

  1 - ON - Manually set ADC record to specified size.

Examples

SENS:SA:ADC:REC:SIZE:FORC:STAT 1

Query Syntax

SENSe<ch>:SA:ADC:REC:SIZE:FORC[:STATe]?  

Return Type  

Boolean

Default  

0

SENSe<ch>:SA:ADC:RECord:SIZE:FORCE:VALue <num>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the integer size value for the force ADC record size feature.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` ADC record size.

Examples

SENS:SA:ADC:REC:SIZE:FORCE:VAL 64

Query Syntax

SENSe<ch>:SA:ADC:REC:SIZE:FORCE:VAL?

Return Type  

Numeric

Default  

Not applicable
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the current maximum available ADC record size.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```
SENS:SA:ADC:REC:SIZE:MAX?
```

**Return Type** Integer

**Default** Not applicable

---

```
SENS<ch>:SA:ADC:RECord:SIZE:MIN?
```

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the current minimum available ADC record size.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```
SENS<ch>:SA:ADC:REC:SIZE:MIN?
```

**Return Type** Integer

**Default** Not applicable

---

```
SENS<ch>:SA:ADC:RECord:SIZE:VALue?
```

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read ADC record size value.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```
```

**Return Type** Integer

**Default** Not applicable

---

```
SENS<ch>:SA:ADC:SAMPle:DECimation:FIR <bool>
```

---
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the FIR filter for 25 MHz decimation.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable 25 MHz FIR filter.
  - **1 - ON** - Enable 25 MHz FIR filter.

**Examples**

```
SENS:SA:ADC:SAMP:DECimation:FIR 1
```

**Query Syntax**

```
SENSe<ch>:SA:ADC:SAMP:DECimation:FIR?
```

**Return Type**

Boolean

**Default**

0

---

**SENSe<ch>:SA:ADC:SAMP:RATE <num>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the ADC sampling frequency. The entered frequency is rounded to the closest value supported by the VNA (25 MHz or 100 MHz).

**Parameters**

- `<ch>` Any existing SA channel.
- `<num>` Choose from 100 MHz or 25 MHz.

Learn about these settings.

**Examples**

```
SENS:SA:ADC:SAMP:RATE 100MHz
```

**Query Syntax**

```
SENSe<ch>:SA:ADC:SAMP:RATE?
```

**Return Type**

Numeric

**Default**

100 MHz

---

**SENSe<ch>:SA:ADC:SAMP:RATE:AUTO <bool>**
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the ADC sample rate mode.

Parameters

<ch>  Any existing SA channel. If unspecified, value is set to 1.

<bool>  Choose from:

0 - OFF - Manually set ADC sampling rate using SENSE:SA:ADC:SAMPLE:RATE.

1 - ON - ADC sampling rate set automatically.

Learn about these settings.

Examples

```
SENS:SA:ADC:SAMP:RATE:AUTO 1
```

Query Syntax

SENSe<ch>:SA:ADC:SAMPle:RATE:AUTO?

Return Type  Boolean

Default  1

SENSe:SA:ADC:STAcking:STATe <bool>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the ON/OFF state of the ADC sample stacking.

Parameters

<ch>  Any existing SA channel. If unspecified, value is set to 1.

<bool>  Choose from:

0 - OFF - ADC sample stacking OFF.

1 - ON - ADC sample stacking ON.

Learn about these settings.

Examples

```
SENS:SA:ADC:STAC:STAT 0
```

Query Syntax

SENSe<ch>:SA:ADC:STACking:STATe?

Return Type  Boolean
**SENSe<ch>:SA:ADC:STACking:VALue [<MAX>,] <value>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the ADC stacking value.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 0 (no stacking).
- `[<MAX>]` Sets and reads the current maximum available stacking size.
- `<value>` Choose a value between 0 and 65535.

Learn about these settings.

**Examples**

```plaintext
SENSe:SA:ADC:STAC:VAL 1
```

**Query Syntax**

```
SENSe<ch>:SA:ADC:STACking:VALue? MAX
```

**Return Type**

Integer

**Default**

0

---

**SENSe<ch>:SA:BANDwidth[:RESolution] <num>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the resolution bandwidth. Also set SENSe:SA:BAND:AUTO OFF.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Choose a value between 6 Hz and 3 MHz. Attempting to set the bandwidth outside these bounds will force the bandwidth to the nearest bound.

Learn about these settings.

**Examples**

```plaintext
SENSe<ch>:SA:BAND 1e3
```

**Query Syntax**

```
SENSe<ch>:SA:BANDwidth[:RESolution]?
```

**Return Type**

Numeric

**Default**

100 kHz
SENSe<ch>:SA:BANDwidth:RESolution <enum>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the minimum and maximum resolution bandwidth.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - **MAX**
  - **MIN**

**Examples**

- `SENS:SA:BAND:RES MAX`

**Query Syntax**

- `SENSe<ch>:SA:BANDwidth:RESolution? MIN`

**Return Type**

- Double

**Default**

- N/A

SENSe<ch>:SA:BANDwidth[:RESolution]:AUTO <bool>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read how the resolution bandwidth is set. When ON, the resolution bandwidth is set based on Span/RBW ratio.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0** - **OFF** - Res. BW is set manually using SENS:SA:BAND.
  - **1** - **ON** - Res. BW is set automatically.

**Examples**

- `SENS:SA:BAND:AUTO 1`

**Query Syntax**

- `SENSe<ch>:SA:BANDwidth[:RESolution]:AUTO?`

**Return Type**

- Boolean

**Default**

- 1
SENSe<ch>:SA:BANDwidth:RESolution:CATalog?

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the list of all Resolution Bandwidth values currently supported with spectrum analyzer current settings.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**
- `SENS:SA:BAND:RES:CAT?`

**Default** Not applicable

SENSe<ch>:SA:BANDwidth:SEARch:OCCupied:MIN <num>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the minimum search frequency to use during an Occupied BW search measurement. Power below this frequency is ignored.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Minimum search frequency value.

**Examples**

**Query Syntax**
- `SENSe<ch>:SA:BANDwidth:SEARch:OCCupied:MIN?`

**Return Type** Numeric

**Default** 250 MHz

SENSe<ch>:SA:BANDwidth:SHAPe <enum>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the resolution bandwidth shape.

**Parameters**

- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
- `<enum>`: Choose from:
  - GAUSsian
  - FLATtop
  - KAISeR
  - BLACkman
  - NONE

Learn about these settings.

**Examples**

SENS:SA:BAND:SHAPE GAUS

**Query Syntax**

SENSe<ch>:SA:BANDwidth:SHAPE?

**Return Type**

Enumeration

**Default**

GAUSsian

---

**SENS<ch>:SA:BANDwidth:VIDeo <enum>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the minimum and maximum video bandwidth.

**Parameters**

- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
- `<enum>`: Choose from:
  - MAX
  - MIN

**Examples**

SENS:SA:BAND:VID MAX

**Query Syntax**

SENSe<ch>:SA:BANDwidth:VIDeo? MIN

**Return Type**

Double
**SENS<ch>:SA:BANDwidth:VIDeo <num>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the Video bandwidth. Also set SENS:SA:BAND:VID:AUTO OFF.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Choose a value between 3 Hz and 3 MHz. Going outside this range places the trace into a hold mode.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS&lt;ch&gt;:SA:BANDvid 1e5</td>
<td>Numeric</td>
<td>100 kHz</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENS<ch>:SA:BANDwidth:VIDeo?

**Return Type**

Numeric

**Default**

N/A

---

**SENS<ch>:SA:BANDwidth:VIDeo:AUTO <bool>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read how the video bandwidth is set. When ON, video bandwidth is set based on RBW/VBW ratio.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Video BW is set manually using SENS:SA:BAND.
  - **1 - ON** - Video BW is set automatically.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS&lt;ch&gt;:SA:BANDvid:AUTO 1</td>
<td>Boolean</td>
<td></td>
</tr>
</tbody>
</table>

**Query Syntax**

SENS<ch>:SA:BANDwidth:VIDeo:AUTO?

**Return Type**

Boolean
SENSe<ch>:SA:BANDwidth:VIDeo:AVERage:COUNt?

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Reads the number of Video bandwidth sweeps that are averaged together. This readout is displayed on the SA setup page.

**Parameters**
- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.

**Examples**
- `SENSe<ch>:SA:BANDwidth:VIDeo:AVERage:COUNt?`

**Query Syntax**
- `SENSe<ch>:SA:BANDwidth:VIDeo:AVERage:COUNt?`

**Return Type**
- Numeric

**Default**
- 1

SENSe<ch>:SA:BANDwidth:VIDeo:AVER:TYPE <enum>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the averaging type.

**Parameters**
- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
- `<enum>`: Choose from:
  - VOLTage
  - POWer
  - LOG
  - VMAX (Voltage Max)
  - VMIN (Voltage Min)

Learn about these settings.

**Examples**
- `SENSe<ch>:SA:BANDwidth:VIDeo:AVER:TYPE VOLT`

**Query Syntax**
- `SENSe<ch>:SA:BANDwidth:VIDeo:AVER:TYPE?`
**Return Type**  Enumeration
**Default**  POWe

**SENS<ch>:SA:BANDwidth:VIDeo:RATio num**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the RBW / VBW ratio.

**Parameters**
- <ch>  Any existing SA channel. If unspecified, value is set to 1.
- <num>  RBW / VBW ratio.

Learn about these settings.

**Examples**  
SENS:SA:BAND:VID:RAT

**Query Syntax**  SENS<ch>:SA:BANDwidth:VIDeo:RATio?

**Return Type**  Numeric
**Default**  1.0


**Applicable Models:** All with Modulation Distortion (S9x070B) and Spectrum Analysis (S9x09xxA/B, S9x09xA/B)

(Write-only) Auto tunes all measurement bands for the modulation distortion channel.

**Parameters**
- <cnum>  Any existing Modulation Distortion channel. If unspecified, value is set to 1.

**Examples**  SENS:SA:COH:DIST:FREQ:TUNE:IMM

**Return Type**  Not applicable
**Default**  Not applicable

**Applicable Models:** All with Modulation Distortion (S9x070B) and Spectrum Analysis (S9x09xxA/B, S9x09xA/B, multitone only)

*(Read Only)* Return the last monitor result of LO feedthru monitor.

**Parameters**
- `<ch>` Any existing MOD channel. If unspecified, value is set to 1.

**Examples**

**Query Syntax**

**Return Type**
- Boolean (1 or 0) A "1" is pass and a "0" is fail.

**Default**
- Not applicable

---


**Applicable Models:** All with Modulation Distortion (S9x070B) and Spectrum Analysis (S9x09xxA/B, S9x09xA/B, multitone only)

*(Read Only)* Return the last monitor value of LO feedthru monitor.

**Parameters**
- `<ch>` Any existing MOD channel. If unspecified, value is set to 1.

**Examples**

**Query Syntax**

**Return Type**
- real (dBm for ABSolute type, dBc for BAND or ATONe types)

**Default**
- Not applicable

---

**SENSe<ch>:SA:COHerence:LO:FTHRu:MONitor:RECeiver <string>**
**Applicable Models:** All with Modulation Distortion (S9x070B) and Spectrum Analysis (S9x09xxA/B, S9x09xA/B, multitone only)

*(Read-Write)* Set and read the target receiver for LO Feedthru monitor. Select the reference receiver in most cases.

**Parameters**
- `<ch>` Any existing MOD channel. If unspecified, value is set to 1.
- `<String>` Specify the receiver port for LO Feedthru monitor.

**Examples**
```
SENS:SA:COH:LO:FTHR:MON:REC "a1"
```

**Query Syntax**
```
SENSe<ch>:SA:COHerence:LO:FTHRu:MONitor:RECeiver?
```

**Return Type** String

**Default** "a1"

---


**Applicable Models:** All with Modulation Distortion (S9x070B) and Spectrum Analysis (S9x09xxA/B, S9x09xA/B, multitone only)

*(Read-Write)* Set and read the LO feedthru monitoring state. Use SENS:SA:COH:LO:FTHR:MON:LAST:FAIL? to query the result. SYST:ERR? does not include the monitor fail result.

**Parameters**
- `<ch>` Any existing MOD channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - LO feedthru monitor disabled.
  - 1 - ON - LO feedthru monitor enabled.

**Examples**
```
SENS:SA:COH:LO:FTHR:MON 1
```

**Query Syntax**
```
SENSe<ch>:SA:COHerence:LO:FTHRu:MONitor[:STATe]?
```

**Return Type** Boolean

**Default** OFF

---

```SENSe<ch>:SA:COHerence:LO:FTHRu:MONitor:TOLerance <real>`
**Applicable Models:** All with Modulation Distortion (S9x070B) and Spectrum Analysis (S9x09xxA/B, S9x09xA/B, multitone only)


**Parameters**

- `<ch>` Any existing MOD channel. If unspecified, value is set to 1.
- `<num>` Specify the threshold level. For ABSolute type: absolute tone level in dBm. For BAND or ATONe types: relative level in dBc.

**Examples**

```
```

**Query Syntax**

```
SENSe<ch>:SA:COHerence:LO::FTHRu:MONitor:TOLerance?
```

**Return Type**

real

**Default**

-40

---

**SENSe<ch>:SA:COHerence:LO:FTHRu:MONitor:TYPE <enum>**

*Applicable Models:* All with Modulation Distortion (S9x070B) and Spectrum Analysis (S9x09xxA/B, S9x09xA/B, multitone only)

*(Read-Write)* Set and read the monitoring method for LO feedthru monitor.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - **BAND:** relative to total band power
  - **ATONe:** relative to average tone power
  - **ABSolute:** absolute tone power

**Examples**

```
SENS:SA:COH:MULT:DATA ALL
```

**Query Syntax**

```
SENSe<ch>:SA:COHerence:LO:FTHRu:MONitor:TYPE?
```

**Return Type**

Enum

**Default**

BAND

---

**SENSe<ch>:SA:COHerence:MULTitone:DATA <enum>**
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the data display mode.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - **ALL**: Legacy SA mode - all frequency points are displayed.
  - **ZNTones**: All the frequencies that are not on the multi-tone coherence grid are set to -200 dBm before being displayed. This setting only has an effect if the coherence multitone mode is enabled.
  - **DNTones**: Deletes span frequencies that are not on the multitone grid.

**Examples**

```
SENS : SA : COHerence : MULT : DATA  ALL
```

**Query Syntax**

```SEnSe<ch>:SA:COHerence:MULTitone:DATA?```

**Return Type**

- Enum

**Default**

**SENSe<ch>:SA:COHerence:MULTitone:HREJect <num>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the number of test signal harmonics you want to be protected against. This adds constraints to the list of LOs used to cover the span.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Number of test signal harmonics to be protected. The more this number is increased, the more constraints are added on the span LOs setting.

**Examples**

```
SENS : SA : COH : MULT : HREJ  0
```

**Query Syntax**

```SEnSe<ch>:SA:COHerence:MULTitone:HREJect?```

**Return Type**

- Integer

**Default**

0
SENSe<ch>:SA:COHerence:MULTitone:NYQReJect <num>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Sets and returns the Nyquist protection level. Avoids Nyquist images of the IF higher order signal to fall back on multitone frequencies. This setting can only be set > 1 if the tone spacing of the multitone is not an integer divider of 100 MHz.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Nyquist protection level.

**Examples**
```
SENS:SA:COH:MULT:NYQR 2
```

**Query Syntax**
```
SENSe<ch>:SA:COHerence:MULTitone:NYQReject?
```

**Return Type**
Integer

**Default**
0

SENSe<ch>:SA:COHerence:MULTitone:PERiod <num>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the test signal repetition rate (in seconds). This value is 1/SA:COH:MULT:SPAC.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Test signal repetition rate (in seconds).

**Examples**
```
SENS:SA:COH:MULT:PER 1E6
```

**Query Syntax**
```
SENSe<ch>:SA:COHerence:MULTitone:PERiod?
```

**Return Type**
Double

**Default**
1E6

SENSe<ch>:SA:COHerence:MULTitone:REFerence <num>
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the multitone image rejection offset frequency. If the multitone grid does not start from 0 Hz, this command is used to set its offset. To make this more convenient, this command accepts as well the frequency of any tone of the multitone grid (Hz).

Parameters
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Offset frequency (in Hz).

Examples
- `SENSe:SA:COHerence:MULTitone:REF 0`

Query Syntax
- `SENSe<ch>:SA:COHerence:MULTitone:REFerence?`

Return Type
- Double

Default
- 0

SENSe<ch>:SA:COHerence:MULTitone:SPACing <num>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the tone spacing of the multitone signal (in Hz). This value is 1/SA:COH:MULT:PER.

Parameters
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Frequency spacing of multitone signal (in Hz).

Examples
- `SENSe:SA:COHerence:MULTitone:SPAC 1E6`

Query Syntax
- `SENSe<ch>:SA:COHerence:MULTitone:SPACing?`

Return Type
- Double

Default
- 1000000

SENSe<ch>:SA:COHerence:MULTitone[:STATe] <bool>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Enables/disables multitone image rejection. When enabled, the window type is set to No Window and the list of RBW possible values is recomputed according to the multitone spacing. When disabled, the window type is set back to what it was before enabling and the RBW list is also set to the previous setting.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - Multitone image rejection disabled.
  - 1 - ON - Multitone image rejection enabled.

**Examples**
- `SENS:SA:COH:MULT 1`

**Query Syntax**
- `SENSe<ch>:SA:COHerence:MULTitone[:STATe]?`

**Return Type**
- Boolean

**Default**
- OFF

**SENSe<ch>:SA:COHerence:MULTitone:VALid?**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-only)* Read the current multitone settings and determine if they are valid or not.

**Parameters**
- `<ch>` Channel number of the measurement. If unspecified, value is set to 1.

**Examples**
- `SENS:SA:COH:MULT:VAL?`
- `sense2:sa:coherence:multitone:valid?`

**Return Type**
- Boolean (ON, OFF, 1, 0) A "1" is valid and a "0" is invalid.

**Default**
- Not applicable

**SENSe<ch>:SA:COHerence:PHASe:DISPlay:LEVel <num>**
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the phase display minimum level.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Phase display minimum level (in dBm).

Learn about these settings.

**Examples**
```
SENS:SA:COH:PHAS:DISP:LEV -60 dBm
```

**Query Syntax**
```
SENSe<ch>:SA:COHerence:PHASe:DISPlay:LEVel?
```

**Return Type** Double

**Default** -60 dBm

**SENSe<ch>:SA:COHerence:PHASe[:STATe] <bool>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Enables/disables phase computing.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Phase computing disabled.
  - **1 - ON** - Phase computing enabled.

Learn about these settings.

**Examples**
```
SENS:SA:COH:PHAS:STAT 1
```

**Query Syntax**
```
SENSe<ch>:SA:COHerence:PHASe[:STATe]?
```

**Return Type** Boolean

**Default** OFF

**SENSe<ch>:SA:COHerence:PULSe:SEARch:COUNt <num>**
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xAX/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the maximum pulse search count for the next SENSe:SA:COHerence:PULSe:SEARch:EXECute.

Parameters
- <ch> Any existing SA channel. If unspecified, value is set to 1.
- <num> Any number between 10 and 500.

Examples
```
SENS:SA:COH:PULS:SEAR:COUN 100
```

Query Syntax
SENSe<ch>:SA:COHerence:PULSe:SEARch:COUNt?

Return Type
Integer

Default
100

SENSe<ch>:SA:COHerence:PULSe:SEARch:DUTY:TOLerance <num>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xAX/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the duty cycle tolerance. Only solutions that ensure this tolerance will be accepted when performing SENSe:SA:COHerence:PULSe:SEARch:EXECute.

Parameters
- <ch> Any existing SA channel. If unspecified, value is set to 1.
- <num> The tolerance value as a percentage, any number between 2 and 100.

Examples
```
```

Query Syntax
SENSe<ch>:SA:COHerence:PULSe:SEARch:DUTY:TOLerance?

Return Type
Double

Default
40
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Write-only)* Executes the pulse search.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```
SENS:SA:COH:PULS:SEAR:EXEC
```

**Return Type** Not applicable

**Default** Not applicable

---

**SENSe<ch>:SA:COHerence:PULSe:SEARch:INITialize**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Write-only)* Sets all the search parameters to their default values:

- Count is set to 100.
- Tolerances for Duty cycle, period, width are all set to 40%.
- Search mode is set to WIDTH.
- Pulse period starting search point is set to the current pulse period.
- Pulse width starting search point is set to the current pulse width.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```
SENS:SA:COH:PULS:SEAR:INIT
```

**Return Type** Not applicable

**Default** Not applicable

---

**SENSe<ch>:SA:COHerence:PULSe:SEARch:ITEM:COUNt?**

```
4573
```

4573
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-only) Returns the number of possible pulse configurations found after a SENSe:SA:COHerence:PULSe:SEARch:EXECute.

Parameters

[ch] Any existing SA channel. If unspecified, value is set to 1.

Examples

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Any existing SA channel. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;num&gt;</td>
<td>Item number from 1 to 500. If unspecified, value is set to 1.</td>
</tr>
</tbody>
</table>

Examples

<table>
<thead>
<tr>
<th>Examples</th>
<th>Description</th>
</tr>
</thead>
</table>

Return Type

Integer

Default

Not applicable

SENSe<ch>:SA:COHerence:PULSe:SEARch:ITEM<num>:PERiod?

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-only) Returns the pulse period for found pulse configuration number num.

Parameters

[ch] Any existing SA channel. If unspecified, value is set to 1.
<br><br><br><br>[<num>] Item number from 1 to 500. If unspecified, value is set to 1.

Examples

<table>
<thead>
<tr>
<th>Examples</th>
<th>Description</th>
</tr>
</thead>
</table>

Return Type

Double

Default

Not applicable
(Write-only) Selects the found configuration number num, and applies it to the measurements.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Item number from 1 to 500. If unspecified, value is set to 1.

**Examples**

```
```

**Return Type**

Not applicable

**Default**

Not applicable

---

**SENSe<ch>:SA:COHerence:PULSe:SEARch:ITEM<num>:WIDTh?**

(Read-only) Returns the pulse width for found pulse configuration number num.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Item number from 1 to 500. If unspecified, value is set to 1.

**Examples**

```
```

**Return Type**

Double

**Default**

Not applicable

---

**SENSe<ch>:SA:COHerence:PULSe:SEARch:PERiod:TOLerance <num>**
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09x(A/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the period tolerance. Only solutions that ensure this tolerance will be accepted when performing SENSe:SA:COHerence:PULSe:SEARch:EXECute.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` The tolerance value as a percentage, any number between 2 and 100.

Examples

```
```

Query Syntax

SENSe<ch>:SA:COHerence:PULSe:SEARch:PERiod:TOLerance?

Return Type

Double

Default

40

SENSe<ch>:SA:COHerence:PULSe:SEARch:PERiod[:VALue] <num>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09x(A/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the period starting point for the next SENSe:SA:COHerence:PULSe:SEARch:EXECute.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Period value (unit seconds).

Examples

```
SENS:SA:COH:PULS:SEAR:PER:VAL 0.001
```

Query Syntax

SENSe<ch>:SA:COHerence:PULSe:SEARch:PERiod:VALue?

Return Type

Double

Default

Not applicable

SENSe<ch>:SA:COHerence:PULSe:SEARch:PRIority <enum>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the priority choice for the search algorithm.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - **PERiod**: Set the search priority to period.
  - **WIDTH**: Set the search priority to width.

**Examples**
```
SENS:SA:COH:PULS:SEAR:PRI PER
```

**Query Syntax**
```
SENSe<ch>:SA:COHerence:PULSe:SEARch:PRIority?
```

**Return Type**
Enumeration

**Default** WIDTH

---

**SENS<ch>:SA:COHerence:PULSe:SEARch:WIDTH:TOLerance <num>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the width tolerance. Only solutions that ensure this tolerance will be accepted when performing SENSe:SA:COHerence:PULSe:SEARch:EXECute.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` The tolerance value as a percentage, any number between 2 and 100.

**Examples**
```
```

**Query Syntax**
```
SENSe<ch>:SA:COHerence:PULSe:SEARch:WIDTH:TOLerance?
```

**Return Type**
Double

**Default** 40

---

**SENS<ch>:SA:COHerence:PULSe:SEARch:WIDTH[:VALue] <num>**
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the width starting point for the next SENSe:SA:COHerence:PULSe:SEARch:EXECute.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Width value (unit: second)

Examples

```
```

Query Syntax

SENSe<ch>:SA:COHerence:PULSe:SEARch:WIDTh:VALue?

Return Type Double

Default Not applicable

SENSe<ch>:SA:COHerence:VECTor:AVERage[:STATe] <bool>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the ON/OFF state of the vector averaging.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:

  0 - OFF - Vector averaging OFF.

  1 - ON - Vector averaging ON.

Examples

```
SENS:SA:COH:VECT:AVER:STAT 1
```

Query Syntax

SENSe<ch>:SA:COHerence:VECTor:AVERage:STATe?

Return Type Boolean

Default 0

SENSe<ch>:SA:COHerence:VECTor:AVERage:VALue <value>[,<enum>]

4578
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the vector averaging value. In addition, read the minimum and maximum values using MIN/MAX.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<value>` Choose a value between 0 and 65536.

Learn about these settings.

- `<enum>`
  - **MIN** - Read the minimum value.
  - **MAX** - Read the maximum value.

**Examples**

- `SENS:SA:COH:VECT:AVER:VAL 1`
- `SENS:SA:COH:VECT:AVER:VAL MAX`

**Query Syntax**

`SENSe<ch>:SA:COHerence:VECTor:AVERage:VALue?`

**Return Type**

Integer

**Default**

1 (no averaging)

---

**SENSe<ch>:SA:DATA:BINs:COUNt?**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the number of DFT points to be exported across the total RF span. Note that this number is modified by the SENSe:SA:COHerence:MULTitone:DATa ZNT setting: If coherent multitone mode is enabled, only the frequency points that are on the multitone grid are exported.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**

- `SENS:SA:DATA:BIN:COUN?`

**Return Type**

Integer

**Default**

Not applicable

---

**SENSe<ch>:SA:DATA:FIFO[:STATe] <bool>**
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Enables/disables exporting data to the FIFO (First-IN, First-OUT) data buffer. FIFO is a circular buffer that allows very fast Read-Write access.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Export data to FIFO disabled.
  - **1 - ON** - Export data to FIFO enabled.

**Examples**
```
SENS:SA:DATA:FIFO 1
```

**Query Syntax**
```
SENSe<ch>:SA:DATA:FIFO[:STATe]
```

**Return Type** Boolean

**Default** 0

**Note:** FIFO commands are under SYTSem:FIFO, and a new set of commands has been added here for binary data.

---

**SENSe<ch>:SA:DATA:FILE:BINary[:STATe] <bool>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Enables/disables binary file (*.bin) output. Data is not exported until the next new sweep occurs.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable binary file output.
  - **1 - ON** - Enable binary file output.

**Examples**
```
SENS:SA:DATA:FILE:BIN 1
```

**Query Syntax**
```
SENSe<ch>:SA:DATA:FILE:BINary[:STATe]
```

**Return Type** Boolean

**Default** 0
SENSe<ch>:SA:DATA:FILE:ERASSe[:STATe] <bool>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Enables/disables erasing output data files after each sweep. When disabled, data is appended to the output file after each sweep which could lead to very large files sizes (and eventually fill the disk).

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - Erase data files after each sweep disabled.
  - 1 - ON - Erase data files after each sweep enabled.

**Examples**
```
SP:SA:DATA:FILE:ERA 1
```

**Query Syntax**
SENSe<ch>:SA:DATA:FILE:ERASSe[:STATe]?

**Return Type**
Boolean

**Default**
1

SENSe<ch>:SA:DATA:FILE:PREFix <string>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and return the file name prefix for the data file. The receivers selected in SENSe:SA:DATA:RECeivers:LIST will be appended to the specified prefix name with either "_X.txt" if a text file is exported (SENSe:SA:DATA:FILE:TEXT) or "_X.bin" if a binary file is exported (SENSe:SA:DATA:FILE:BINary). X is the receiver name.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<string>` String. Specified prefix.

**Examples**
```
SP:SA:DATA:FILE:PREF "C:\TEMP\SA_DATA_OUT"
```

**Query Syntax**
SENSe<ch>:SA:DATA:FILE:PREFix?

**Return Type**
String

**Default**
"C:\TEMP\SA_DATA_OUT"

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Enables/disables adding marker data to the text file (*.txt) output.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Do not add marker data to the text file output.
  - **1 - ON** - Add marker data to the text file output.

**Examples**

```
SENS:SA:DATA:FILE:TEXT:MARK:STAT 1
```

**Query Syntax**

```
SENSe<ch>:SA:DATA:FILE:TEXT:MARKers[:STATE]?
```

**Return Type**

Boolean

**Default**

0

---

### SENSE<ch>:SA:DATA:FILE:TEXT[:STATE] <bool>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Enables/disables text file (*.txt) output. Data is not exported until the next new sweep occurs.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable text file output.
  - **1 - ON** - Enable text file output.

**Examples**

```
SENS:SA:DATA:FILE:TEXT 1
```

**Query Syntax**

```
SENSe<ch>:SA:DATA:FILE:TEXT[:STATE]?
```

**Return Type**

Boolean

**Default**

0
SENSe<ch>:SA:DATA:FILE:TEXT:VERBose[:STATe] <bool>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Enables/disables exporting frequency and data for text files. Data is not exported until the next new sweep occurs.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable text file output.
  - **1 - ON** - Enable text file output.

**Examples**

```
SENS:SA:DATA:FILE:TEXT:VERB:STAT 1
```

**Query Syntax**

SENSe<ch>:SA:DATA:FILE:TEXT:VERBose[:STATe]?

**Return Type** Boolean

**Default** 0

SENSe<ch>:SA:DATA:KEEP:FREQuencies:COUNt?

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the number of frequency domain points in memory. This command should be called only after a Single sweep, and after making sure the sweep is finished. Learn about these settings.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```
SENS:SA:DATA:KEEP:FREQ:COUN?
```

**Return Type** Integer

**Default** Not applicable

SENSe<ch>:SA:DATA:KEEP:RECeivers:COUNt?
**Applicable Models:** NAll with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the number of receivers that have been stored in memory. Learn about these settings.

**Parameters**
- `<ch>`  Any existing SA channel. If unspecified, value is set to 1.

**Examples**
- `SENS:SA:DATA:KEEP:REC:COUN?`

**Return Type**  Integer
**Default**  Not applicable

SENSe<ch>:SA:DATA:KEEP:RECeivers[:CURRent]?

**Applicable Models:** NAll with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the current list of receivers that have been stored in memory. Learn about these settings.

**Parameters**
- `<ch>`  Any existing SA channel. If unspecified, value is set to 1.

**Examples**
- `SENS:SA:DATA:KEEP:REC?  "b2,a1"

**Return Type**  String
**Default**  Not applicable

SENSe<ch>:SA:DATA:KEEP:RECeivers:LIST <string>
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the list of receivers to export. The order set using this command will determine the order in which data will be exported. Learn about these settings.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<string>` String. Comma separated list of receivers. The “All” option will select all the currently working receivers of the current channel. If the user sets an empty list, this is equivalent to sending “ALL”.

Examples

```
SENS:SA:DATA:KEEP:RECeivers:LIST "A,B,a1,a2"
```

Query Syntax

```
SENSe<ch>:SA:DATA:KEEP:RECeivers:LIST?
```

Return Type String

Default Not applicable

**Note:** This list can contain receivers that are not currently measured in the channel. However, this is not an issue. To get the current list of receivers that export data, query SENS:SA:DATA:KEEP:REC?

---


Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-only) Read the scalar data. Scalar data is amplitude only values, when the Compute phase mode is not turned on. The resulting values are float numbers, as Volts RMS in a 50 Ohm load. The float number is the number that can be queried with the SENS:SA:DATA:KEEP:FREQuencies:COUNt? command. Learn about these settings.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<recName>` (String) Receiver name
- `[position,Length]` Starting position of data and quantity. The default is 0, full span (the value is queried using the SENS:SA:DATA:KEEP:FREQuencies:COUNt? command).

Examples

```
SENS:SA:DATA:KEEP:SCAL:GET? "b2"
```

Return Type String

Default Not applicable

---

4585
SENSe<ch>:SA:DATA:KEEP[:STATe] <bool>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the ON/OFF state of the end of sweep processing. Learn about these settings.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - End of sweep processing OFF.
  - **1 - ON** - End of sweep processing ON.

**Examples**
```
SENS:SA:DATA:KEEP 1
```

**Query Syntax**
SENSe<ch>:SA:DATA:KEEP[:STATe]?

**Return Type** Boolean

**Default** 0

SENSe<ch>:SA:DATA:KEEP:VECTor:GET? <recName> [,position,Length]

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the vector data. Vector data is amplitude and phase, when the Compute phase mode is not turned on. The resulting values are 2 float numbers per frequency. The float number is the number that can be queried with the SENSe:SA:DATA:KEEP:FREQuencies:COUNt? command. Learn about these settings.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<recName>` (String) Receiver name
- `[,.position,Length]` Starting position of data and quantity. The default is 0, full span (the value is queried using the SENSe:SA:DATA:KEEP:FREQuencies:COUNt? command).

**Examples**
```
```

**Return Type** String

**Default** Not applicable
SENSe<ch>:SA:DATA:RECeivers?

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-only) Read the list of receivers that will be exported to a data file.

Parameters

<ch> Any existing SA channel. If unspecified, value is set to 1.

Examples

Examples

SENS:SA:DATA:RECeivers?

Return Type String

Default Not applicable

Note: The list is set with SENS:SA:DATA:REC:LIST can contain more receivers, this query will only return the ones that are currently measured and that are in the receiver list.

SENSe<ch>:SA:DATA:RECeivers:COUNt?

Applicable Models: NAll with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-only) Read the number of currently exported receivers.

Parameters

<ch> Any existing SA channel. If unspecified, value is set to 1.

Examples

SENS:SA:DATA:REC:COUN?

Return Type Integer

Default Not applicable

SENSe<ch>:SA:DATA:RECeivers:LIST <string>
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the list of receivers to export. The order set using this command will determine the order in which data will be exported to the FIFO data buffer.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<string>` String. Comma separated list of receivers. The “All” option will select all the currently working receivers of the current channel. If the user sets an empty list, this is equivalent to sending “ALL”.

Examples

```
SENS:SA:DATA:REC:LIST "A,B,a1,a2"
```

Query Syntax

```
SENSe<ch>:SA:DATA:RECeivers:LIST?
```

Return Type

String

Default

Not applicable

Note: This list can contain receivers that are not currently measured in the channel. However, this is not an issue. To get the current list of receivers that export data, query SENS:SA:DATA:REC?

SENSe<ch>:SA:DATA:SHARed:NAME <string>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Assigns a specified name to the Microsoft Windows shared data mechanism when SENSe:SA:DATA:SHARed:STATe is enabled.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<string>` String. Memory file name.

Examples

```
SENS:SA:DATA:SHAR:NAM "Mem_Share"
```

Query Syntax

```
SENSe<ch>:SA:DATA:SHARed:NAME?
```

Return Type

String

Default

Not applicable

SENSe<ch>:SA:DATA:SHARed[:STATe] <bool>
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Enables/disables exporting data to shared memory, which is the fastest way to transfer data between applications.

Parameters

- <ch> Any existing SA channel. If unspecified, value is set to 1.
- <bool> Choose from:
  - 0 - OFF - Disable memory sharing.
  - 1 - ON - Enable memory sharing.

Examples

```
SENS:SA:DATA:SHAR 1
```

Query Syntax

```
SENSe<ch>:SA:DATA:SHARed[:STATe]?
```

Return Type Boolean

Default 0

---

**SENSe<ch>:SA:DATA:SIZE?**

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-only) Read the byte size of the data to be exported in binary mode.

Parameters

- <ch> Any existing SA channel. If unspecified, value is set to 1.

Examples

```
SENS:SA:DATA:SIZE?
```

Return Type Integer

Default Not applicable

**Note:** Returned number can exceed the maximum integer number size. In that case, an error will be raised. For that reason, we provide an access to larger numbers with the same query and LSB or MSB suffixes.

---

**SENSe<ch>:SA:DATA:SIZE:BIN?**
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-only) Read the byte size of one data bin in binary mode.

Parameters
<ch> Any existing SA channel. If unspecified, value is set to 1.

Examples SENS:SA:DATA:SIZE:BIN?

Return Type Integer
Default Not applicable

SENSe<ch>:SA:DATA:SIZE:HIGH?

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-only) Read the most significant bytes (MSB) of the byte size of the data to be exported.

Parameters
<ch> Any existing SA channel. If unspecified, value is set to 1.

Examples SENS:SA:DATA:SIZE:HIGH?

Return Type Integer
Default Not applicable

SENSe<ch>:SA:DATA:SIZE:LOW?

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-only) Read the least significant bytes (LSB) of the byte size of the data to be exported.

Parameters
<ch> Any existing SA channel. If unspecified, value is set to 1.

Examples SENS:SA:DATA:SIZE:LOW?

Return Type Integer
Default Not applicable

SENSe<ch>:SA:DATA:STARt?
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-only) Read the frequency of the first RF bin.

Parameters
- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.

Examples
- `SENS:SA:DATA:STAR?`

Return Type: Double
Default: Not applicable

Note: This value can differ slightly from the SA Sweep start frequency, the frequency of the first RF bin is aligned with the current DFT grid.

SENSe<ch>:SA:DATA:THReshold[:STATe] <bool>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Enables/disables data level threshold mode. Set the threshold level using the SENSe:SA:DATA:THReshold:VALue command.

Parameters
- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
- `<bool>`: Choose from:
  - 0 - OFF: Disable threshold mode.
  - 1 - ON: Enable threshold mode.

Examples
- `SENS:SA:DATA:TH:STAT 1`

Query Syntax
- `SENSe<ch>:SA:DATA:THReshold[:STATe]?`

Return Type: Boolean
Default: 0

SENSe<ch>:SA:DATA:THReshold:VALue <num>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the threshold value (dBm). For text file output with verbose mode, only the frequencies with power greater than this threshold setting will be written to the file. This command can be used as a kind of simple spurious search.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Threshold setting in dBm.

**Examples**

**Query Syntax**

SENSe<ch>:SA:DATA:THR:VALue?

**Return Type**

Real

**Default**

-60 dBm

---

**SENSe<ch>:SA:DATA:TYPE <enum>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the data format.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - **MAGDb** : Set data format to log magnitude in dBm.
  - **AMPVolt** : Set data format to linear magnitude in volts.
  - **PINT** : Set data format to Packed Integers: a more compact (2 bytes) numeric representation for dBm. Each set of 2 bytes is a short number `s`, to get the dBm value compute `(s/200.0 -36.165)`.

**Examples**

**Query Syntax**

SENSe<ch>:SA:DATA:TYPE?

**Return Type**

Enumeration

**Default**

MAGDB

---

**SENSe<ch>:SA:DATA:WFACtor?**
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-only) Read the windowing factor for band power computation. This factor is derived from the window type (Gaussian, flat top, etc.). When doing the sum of linear power over a band, use this factor to compensate the side lobe effect of windowing to get an accurate band power value.

**Parameters**

<ch> Any existing SA channel. If unspecified, value is set to 1.

**Examples**

SENS:SA:DATA:WFACtor?

**Return Type** Real

**Default** Not applicable

SENSe:SA:DETector:BYPass:[STATe] <bool>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the ON/OFF state of the detector bypass setting.

**Parameters**

<ch> Any existing SA channel. If unspecified, value is set to 1.

<bool> Choose from:

0 - OFF - Detector bypass OFF.

1 - ON - Detector bypass ON.

Learn about these settings.

**Examples**

SENS:SA:DET:BYP 0

**Query Syntax**

SENSe<ch>:SA:DETector:BYPass:[STATe]?

**Return Type** Boolean

**Default** 0

SENSe<ch>:SA:DETector:FUNCtion <enum>
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the detector type.

Parameters

  <ch> Any existing SA channel. If unspecified, value is set to 1.
  <enum> Choose from:

       PEAK
       AVERage
       SAMPlе
       NORMal
       NEGPeak
       PSAMple (Peak Sample)
       PAVerage (Peak Average)

Learn about these settings.

Examples

Query Syntax

SENSe<ch>:SA:DETector:FUNCtion?

Return Type

Enumeration

Default

PEAK

SENSe<ch>:SA:DFT:BANDwidth:AUTO <bool>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the default values for DFT bandwidth.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:

  0 - **OFF** - DFT minimum and maximum values are set manually:
  
  - **Narrow** - 500 kHz to 11 MHz
  - **Wide** - 500 kHz to 44 MHz

  1 - **ON** - DFT minimum and maximum values are set to their default values:
  
  - **Narrow** - 1 MHz to 10 MHz
  - **Wide** - 1 MHz to 34 MHz

**Examples**

```
SENS:SA:DFT:BAND:_AUTO 0
```

**Query Syntax**

```
SENSe<ch>:SA:DFT:BANDwidth:AUTO?
```

**Return Type** Boolean

**Default** 1

---

**SENS<ch>:SA:DFT:BANDwidth:NARRow:MAX <num>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the maximum value for narrow DFT bandwidth. The maximum narrow DFT bandwidth setting is 11 MHz. The SENSe:SA:DFT:BANDwidth:AUTO must be set to OFF to set this value manually.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Max narrow DFT bandwidth.

**Examples**

```
```

**Query Syntax**

```
SENSe<ch>:SA:DFT:BANDwidth:NARRow:MAX?
```

**Return Type** Double

---

4595
**SENSe<ch>:SA:DFT:BANDwidth:NARRow:MIN <num>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the minimum value for narrow DFT bandwidth. The minimum narrow DFT bandwidth setting is 500 kHz. The SENSe:SA:DFT:BANDwidth:AUTO must be set to OFF to set this value manually.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Minimum narrow DFT bandwidth.

**Examples**
```
SENSe<ch>:SA:DFT:BANDwidth:NARRow:MIN 5e5
```

**Query Syntax**
```
SENSe<ch>:SA:DFT:BANDwidth:NARRow:MIN?
```

**Return Type**
Double

**Default**
1e6

---

**SENSe<ch>:SA:DFT:BANDwidth:WIDE:MAX <num>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the maximum value for wide DFT bandwidth. The maximum wide DFT bandwidth setting is 44 MHz. The SENSe:SA:DFT:BANDwidth:AUTO must be set to OFF to set this value manually.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Max wide DFT bandwidth.

**Examples**
```
SENSe<ch>:SA:DFT:BANDwidth:WIDE:MAX 44e6
```

**Query Syntax**
```
SENSe<ch>:SA:DFT:BANDwidth:WIDE:MAX?
```

**Return Type**
Double

**Default**
34e6
SENSe<ch>:SA:DFT:BANDwidth:WIDE:MIN <num>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the minimum value for wide DFT bandwidth. The minimum wide DFT bandwidth setting is 500 kHz. The SENSe:SA:DFT:BANDwidth:AUTO must be set to OFF to set this value manually.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Minimum wide DFT bandwidth.

**Examples**
```
SENS:SA:DFT:BAND:WIDE:MIN 5e5
```

**Query Syntax**
SENSe<ch>:SA:DFT:BANDwidth:WIDE:MIN?

**Return Type**
Double

**Default**
1e6

SENSe<ch>:SA:DFT:RECord:SIZE?

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the current DFT record size. This value is based on the SENSe:SA:ADC:RECord:SIZE:VALue and SENSe:SA:DFT:TYPE settings.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**
```
SENS:SA:DFT:REC:SIZE?
```

**Return Type**
Integer

**Default**
Not applicable

SENSe<ch>:SA:DFT:RESolution?
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-only) Read the DFT resolution.

**Parameters**

<ch>  Any existing SA channel. If unspecified, value is set to 1.

**Examples**

SENS:SA:DFT:RES?

**Default** Not applicable

**SENS<ch>:SA:DFT:TYPE <enum>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and return the DFT record size type. The DFT SENSE:SA:DFT:RECORD:SIZE is based on the SENSE:SA:ADC:RECORD:SIZE:VALUE and the DFT record size type.

**Parameters**

<ch>  Any existing SA channel. If unspecified, value is set to 1.

<enum>  Choose from:

- **POW2** - Sets the DFT record size to the next power of 2 greater than or equal to the current ADC record size.

- **RADix** - Sets the DFT to the nearest equal or larger integer that can be decomposed with 2, 3, 5, 7, 11, 13 radixes.

- **ARBitrary** - Sets DFT record size equal to the ADC record size. If the current ADC record size is a large prime number, then the DFT can be very slow.

- **FASTest** - Sets the DFT record size as close as possible to the ADC record size (larger or equal) while optimizing processing speed.

**Examples**

SENS:SA:DFT:TYPE ARB

**Query Syntax**

SENS<ch>:SA:DFT:TYPE?

**Return Type**  Enumeration

**Default** RADix

**Note:** In previous releases the default was POW2.
**SENSe<ch>:SA:FFT:DITHer[:STATE] <bool>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the ON/OFF state of the FFT grid dither setting

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Dither OFF.
  - **1 - ON** - Dither ON.

**Examples**

```
SENS:SA:FFT:DITH ON
```

**Query Syntax**

```
SENSe<ch>:SA:FFT:DITH?
```

**Return Type**

Boolean

**Default**

0

---

**SENSe<ch>:SA:FFT:RES?**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the FFT resolution.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```
SENS:SA:FFT:RES?
```

**Default**

Not applicable

---

**SENSe<ch>:SA:FREQuency:SPAN:BANDwidth[:RESolution]:RATio <value>**
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Red-Write) Set and read the Frequency Span / RBW ratio.

Parameters

<ch> Any existing SA channel. If unspecified, value is set to 1.
[value> Frequency Span / RBW ratio. Choose a value between 1 and 200e9.

Learn about these settings.

Examples SENS:SA:FREQ:SPAN:BAND:RAT 100

Query Syntax SENSE<ch>:SA:FREQuency:SPAN:BANDwidth[:RESolution]:RATio ?

Return Type Integer

Default 106

SENSe<ch>:SA:FREQuency:TUNE:IMMediate [sync]

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Write-only) Auto tunes and zooms in on a signal within a SA sweep.

Parameters

<ch> Any existing SA channel. If unspecified, value is set to 1.
[sync] SYNChronous - blocks SCPI commands during auto tune process (default behavior). When no sync is specified, sync is set to SYNChronous.

Note: The auto tune feature can take seconds to process, so a timeout error is likely to occur when sending other commands/queries before the completion of the auto tune process.

ASYNchronous - does NOT block SCPI commands during auto tune process.

Note: The ASYNchronous mode provides a method to check/wait remotely for completion of the auto tune process while allowing other commands/queries to be sent.

Note: In ASYNchronous mode, *OPC? will freeze the instrument and should not be used. Instead, use *OPC and *ESR? to check for completion.

Examples SENS:SA:FREQ:TUNE:IMM
SENSe<ch>:SA:IMAGe:REJect <enum>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the image reject mode.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - NHIGH
  - NLOW
  - MIN
  - MHIGH
  - MLOW
  - NORMAL
  - BETTER
  - MAX

Learn about these settings.

**Examples**

```
SENS : SA : IMAG : REJ NLOW
```

**Query Syntax**

SENSe<ch>:SA:IMAGe:REJect?

**Return Type**

Enumeration

**Default**

NORMal

SENSe<ch>:SA:IMAGe:STRENgth <enum>
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Sets and returns the image rejection strength. During the image rejection process, several LO acquisitions overlap at the same RF frequency. As a result, different RF signal values can be returned. This command sets the acceptable power differences between LOs in determining actual signals.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - **WEAK**: 3 dB (approximate number, depends on RBW)
  - **NORMAL**: 1 dB (approximate number, depends on RBW)
  - **STROng**: 0.5 dB (approximate number, depends on RBW)

Examples

```
SENS:SA:IMAG:STREN STROng
```

Query Syntax

```
SENSe<ch>:SA:IMAGe:STRENgth?
```

Return Type

Enumeration

Default

MEDium

SENSe<ch>:SA:LO:BASeband[:STATe] <bool>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Enables and disables baseband sweep independent of the LO sweep to allow signals down to 1 Hz to be analyzed. Learn more.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF**: Disable SA baseband.
  - **1 - ON**: Enable SA baseband.

Examples

```
SENS:SA:LO:BA
```

Query Syntax

```
SENSe<ch>:SA:LO:BASeband[:STATe]?
```

Return Type

Boolean
SENSe<ch>:SA:LO:COUNt?

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Returns the number of LO acquisitions determined by the **Image Reject** selection and the span.

**Parameters**

- `<ch>` Channel number of the measurement. If unspecified, value is set to 1.

**Examples**

```
SENSe:SA:LO:COUNt?
sense2:sa:lo:count?
```

**Default** Not applicable

SENSe<ch>:SA:LO:FORCe:FREQuency <num>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read enable force LO to the specified frequency.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Frequency in Hz. Choose a value within the frequency range of the analyzer.

Learn about these settings.

**Examples**

```
SENSe:SA:LO:FORC:FREQ 1e9
```

**Query Syntax** SENSe<ch>:SA:LO:FORCe:FREQuency?

**Return Type** Numeric

**Default** 1 GHz

SENSe<ch>:SA:LO:FORCe:OFFSet:DIVider <num>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the divider in the equation for setting LO frequency. Learn more.

### Parameters
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Divider in the equation for setting LO frequency.

### Examples
```
SENS:SA:LO:FORC:OFFS:DIV 1
```

### Query Syntax
```
SENSe<ch>:SA:LO:FORCe:OFFSet:DIVider?
```

### Return Type
Numeric

### Default
1

---

**SENSe<ch>:SA:LO:FORCe:OFFSet:MULtiplier <num>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the multiplier in the equation for setting LO frequency. Learn more.

### Parameters
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<num>` Multiplier in the equation for setting LO frequency.

### Examples
```
SENS:SA:LO:FORC:OFFS:MUL 1
```

### Query Syntax
```
SENSe<ch>:SA:LO:FORCe:OFFSet:MULtiplier?
```

### Return Type
Numeric

### Default
1
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the source name in the equation for setting LO frequency. Learn more.

Parameters

- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.

Examples

```
SENS:SA:LO:FORC:OFFS:SOUR "Port 1"
sense2:sa:lo:force:offset:source "Port 1 Src2"
```


Return Type: String

Default: Not applicable

---

**SENS<ch>:SA:LO:FORCE[:STATe] <bool>**

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read enable force LO to frequency mode.

Parameters

- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
- `<bool>`: Choose from:
  - **0** - OFF - Force LO to frequency is disabled.
  - **1** - ON - Manually set LO to specified frequency using SENS:SA:LO:FORCE:FREQuency. Only applied if Image Reject is set to None, LO High or None, LO Low.

Examples

```
SENS:SA:LO:FORCE 1
```

Query Syntax: `SENS<ch>:SA:LO:FORCE[:STATe]?`

Return Type: Boolean

Default: 0
SENSe<ch>:SA:LO:FREQ:FORCe <bool> *(Obsolete)*

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read enable force LO to frequency mode. This command is replaced by SENSe:SA:LO:FORCe[:STATe] .

Parameters

<ch> Any existing SA channel. If unspecified, value is set to 1.

<bool> Choose from:

0 - OFF - Force LO to frequency is disabled.

1 - ON - Manually set LO to specified frequency using SENSe:SA:LO:FREQ:VALue . Only applied if Image Reject is set to None, LO High or None, LO Low.

Examples SENSe:SA:LO:FREQ:FORC 1

Query Syntax SENSe<ch>:SA:LO:FREQ:FORC?

Return Type Boolean

Default 0

SENSe<ch>:SA:LO:FREQ:VALue <num> *(Obsolete)*

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read enable force LO to frequency. This command is replaced by SENSe:SA:LO:FORCe:FREQuency .

Parameters

<ch> Any existing SA channel. If unspecified, value is set to 1.

<num> Frequency in Hz. Choose a value within the frequency range of the analyzer.

Examples SENSe:SA:LO:FREQ:VAL 1e9

Query Syntax SENSe<ch>:SA:LO:FREQ:VAL?

Return Type Numeric

Default 1 GHz
SENSe<ch>:SA:LO:RANDom[:STATe] <bool>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the LO randomize state.

Parameters

<ch> Any existing SA channel. If unspecified, value is set to 1.
<char> Choose from:

0 - OFF - LO Randomize is set to OFF.
1 - ON - LO Randomize is set to ON.

Examples

Query Syntax
SENSe<ch>:SA:LO:RANDom[:STATe]?

Return Type
Boolean

Default
1


Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the DC sweep order. The SA may be programmed to loop through a series of spectrum measurements at multiple RF source frequencies, multiple RF source powers, and multiple DC voltages. These settings determine whether the DC sources are swept before the RF power and frequencies are swept, or whether the DC sources are swept after the RF power and frequencies are swept.

Parameters

<ch> Any existing SA channel. If unspecified, value is set to 1.
<enum> DC sweep order. Choose from:

DC - Sweep through each DC voltage step first then sweep through the next frequency.
RF - Sweep through each frequency step first then sweep through the next DC voltage.
**SENS**:SA:SOUR:DC:SWEep:FIRD <value>

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the number of steps the source will make across the specified source DC range. This setting is common to all sources.

**Parameters**

- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
- `<value>`: Point count. Choose an integer value of 1 or higher.

**Examples**

- **SENSe**:SA:SOUR:DC:SWEep:POInT 100

**Query Syntax**: SENSe<ch>:SA:SOUR:DC:SWEep:POInT?

**Return Type**: Integer

**Default**: 1

---

**SENSe**:SA:SOURce:DC:SWEep[:STATe] <bool>

---
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the ON/OFF state of the DC sources. If ON, the DC sources sweep between their start and stop voltages. If OFF, the DC sources are set to their start voltages.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - DC sweep OFF.
  - 1 - ON - DC sweep ON.

Learn about these settings.

Examples

```
SENS:SA:SOUR:DC:SWE 1
```

Query Syntax

```
SENSe<ch>:SA:SOURce:DC:SWEep[:STATe]?
```

Return Type

Boolean

Default

0

---

SENS<ch>:SA:SOURce<port>:FREQuency:CW <num>[,src]

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the source CW frequency.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<num>` CW frequency in Hz. Choose a value within the frequency range of the analyzer.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.
SENSe<ch>:SA:SOURce<port>:FREQuency:STARt <num>[,src]

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the source start frequency.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<num>` Start frequency in Hz. Choose a value within the frequency range of the analyzer.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
SENSe:SA:SOUR:FREQ:STAR 1e9
sense2:sa:source:frequency:star 1e9,"Port 1 Src2"
```

**Query Syntax**

`SENSe<ch>:SA:SOURce<port>:FREQuency:STARt?`

**Return Type** Numeric

**Default** Start frequency of the analyzer.

SENSe<ch>:SA:SOURce<port>:FREQuency:STOP <num>[,src]
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the source stop frequency.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<num>` Stop frequency in Hz. Choose a value within the frequency range of the analyzer.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

SENS:SA:SOUR:FREQ:STOP 1e10
sense2:sa:source:frequency:stop 1e9, "Port 1 Src2"

Query Syntax

SENSe<ch>:SA:SOURce<port>:FREQuency:STOP?

Return Type

Numeric

Default

Stop frequency of the analyzer.

SENSe<ch>:SA:SOURce:POW:SWEep:POINT:COUNt <value>
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the number of steps the source will make across the specified source power range. This setting is common to all sources.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<value>` Point count. Choose an integer value of 1 or higher.

**Examples**

```
SENS:SA:SOUR:POW:SWE:POIN:COUNT 100
```

**Query Syntax**

```
SENSe<ch>:SA:SOUR:POW:SWEep:POINt:COUNt?
```

**Return Type**

Integer

**Default**

1

**SENSe<ch>:SA:SOURce:POW:SWEep:REPeat:COUNt <value>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the number of SA (receiver) sweeps for each Source Step. This setting is common to all sources.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<value>` Repeat count. Choose an integer value of 1 or higher.

**Examples**

```
```

**Query Syntax**

```
SENSe<ch>:SA:SOUR:POW:SWEep:REPeat:COUNt?
```

**Return Type**

Integer

**Default**

1

**SENSe<ch>:SA:SOURce<port>:POWer:STARt <dBm> [.src]**
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the source start power level. This command applies to Power or LFPower sweep types.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.
- `<dBm>` Start power level in dBm. Choose a value within the power range of the source.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

**Examples**
```
SENS:SA:SOUR:POW:STAR 0
```

**Query Syntax**
```
SENS<ch>:SA:SOURce<port>:POWer:STARt?
```

**Return Type** Numeric

**Default** Default of source

---

**SENSe<ch>:SA:SOURce<port>:POWer:STOP <dBm>[,.src]**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the source stop power level. This command applies to Power or LFPower sweep types.

**Parameters**
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.
- `<dBm>` Stop power level in dBm. Choose a value within the power range of the source.
While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
```

**Query Syntax**

```
SENSe<ch>:SA:SOURce<port>:POWer:STOP?
```

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Numeric</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>Default of source</td>
</tr>
</tbody>
</table>

---

SENSe<ch>:SA:SOURce<port>:POWer[:VALue] <dBm>[,src]

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<dBm>` Source output power level in dBm. Choose a value within the power range of the source.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
SENS:SA:SOUR:POW -5
```

**Query Syntax**

```
SENSe<ch>:SA:SOURce<port>:POWer?
```

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Numeric</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>Default of source</td>
</tr>
</tbody>
</table>
Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the sweep order. This command applies whenever frequency and power are being swept (LFPower sweep type). Otherwise, this setting is ignored. For example, if all the active sources are set to CW and/or LINear sweep type, or if all the active sources are set to CW and/or POWer sweep type, the sweep order is ignored. If any active source is set to LFPower sweep type, or if an active source is set to LINear sweep type and another active source is set to POWer sweep type, then the sweep order setting will be used.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<enum>` Sweep order. Choose from:
  - `FREQ` - Sweep from Start to Stop frequency first followed by a power sweep.
  - `POWer` - Sweep power first then sweep from Start to Stop frequency.

Examples

```
SENS:SA:SOUR:SWE:FIR POW
```

Query Syntax

```
SENSe<ch>:SA:SOUR:SWEep:FIRst?
```

Return Type

Enumeration

Default

FREQ

SENSe<ch>:SA:SOURce<port>:SWEep:POINt:COUNt <value>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

(Read-Write) Set and read the number of steps the source will make across the specified source frequency range.

Parameters

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<value>` Point count. Choose a value between 1 and 2e9.

Examples

```
SENS:SA:SOUR:SWE:POIN:COUN 100
```

Query Syntax

```
SENSe<ch>:SA:SOUR:SWEep:POINt:COUNt?
```

Return Type

Integer
**SENSe<ch>:SA:SOURce<port>:SWEep:REPeat:COUNt <value>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the number of SA (receiver) sweeps for each Source Step. This setting is common to all sources.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.
- `<value>` Repeat count. Choose a value between 1 and 2e9.

**Examples**

```
```

**Query Syntax**

`SENSe<ch>:SA:SOUR:SWEep:REPeat:COUNt?`

**Return Type**

Integer

**Default**

1

---

**SENSe<ch>:SA:SOURce<port>:SWEep:TYPE <enum>[,src]**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the source sweep type.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.
- `<enum>` Sweep type. Choose from:
  - **CW** - SA source is at a single frequency, set with `SENS:SA:SOUR:FREQ:CW`.
  - **LINear** - SA source sweeps from Start to Stop in linear steps.
  - **POWer** - SA source is set to a power sweep.
  - **LFPower** - SA source is set to sweep from the Start to Stop frequency and...
power sweep. The order is determined by the SENS:SA:SOUR:SWEep:FIRst command.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```plaintext
SENS:SA:SOUR:SWE:TYPE CW
sense2:sa:source:sweep:type linear,"Port 1 Src2"
```

**Query Syntax**

SE NSSe<ch>:SA:SOURce<port>:SWEep:TYPE?

**Return Type**

Enumeration

**Default**

CW

**SENSes<ch>:SA:SPAN:BINS:COUNt?**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-only)* Read the current span DFT bin count.

**Parameters**

- **<ch>** Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```plaintext
SENS:SA:SPAN:BINS:COUN?
```

**Return Type**

Integer

**Default**

Not applicable

**SENSes<ch>:SA:TRACe:IMAGe[:STATe] <bool>**
**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B)

*(Read-Write)* Set and read the show / hide state of the image reject traces in the measurement parameters dialog.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable image reject traces.

Learn about these settings.

**Examples**

```
SENS:SA:TRAC:IMAG ON
```

**Query Syntax**

```
SENSe<ch>:SA:TRACe:IMAGe[:STATe]?
```

**Return Type** Boolean

**Default** 0

---

**SENSe:SA:TRIGer:LEVel[:STATe] <bool>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the ON/OFF state of a measurement trigger event that will occur whenever the ADC level is greater than the value specified using the `SENSe:SA:TRIGer:LEVel:VALue` command.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - ADC measurement trigger OFF.
  - **1 - ON** - ADC measurement trigger ON.

Learn about these settings.

**Examples**

```
SENSe<ch>:SA:TRIG:LEV 0
```

**Query Syntax**

```
SENSe<ch>:SA:TRIGer:LEVel[:STATe]?
```
SENSe<ch>:SA:TRIGer:LEVel:VALue <value>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the ADC trigger level.

Parameters

<ch> Any existing SA channel. If unspecified, value is set to 1.
<value> Choose a value between 0 and 16383.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Examples</th>
</tr>
</thead>
</table>

SENSe:SA:TRIGer:PERCounter[:STATe] <bool>

Applicable Models: All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

(Read-Write) Set and read the ON/OFF state of a measurement trigger event based on the specified period set using the SENS:SA:TRIGer:PERCounter:VALue command.

Parameters

<ch> Any existing SA channel. If unspecified, value is set to 1.
<bool> Choose from:

| 0 - OFF | Periodic counter OFF. |
| 1 - ON  | Periodic counter ON. |

Examples

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:SA:TRIG:PERC 0</td>
</tr>
<tr>
<td><strong>Query Syntax</strong></td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
</tr>
<tr>
<td><strong>Default</strong></td>
</tr>
</tbody>
</table>

**SENSe<ch>:SA:TRIGer:PERCounter:VALue <value>**

**Applicable Models:** All with Spectrum Analysis Options (S9x09xxA/B, S9x09xA/B) (Except M98x0A, P50xxA)

*(Read-Write)* Set and read the periodic counter value. This command initiates a measurement trigger event based on the specified period.

**Parameters**

- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<value>` Choose a value between 0 and 2147483647.

Learn about these settings.

**Examples**

SENSe:SA:TRIG:PERC:VAL 64

**Query Syntax**

SENSe<ch>:SA:TRIGer:PERCounter:VALue?

**Return Type**

Integer

**Default**

256
Defines the segment sweep settings.

Enable segment sweep with SENS:SWE:TYPE SEGment.

```
SENSe:Segment
  | ADD
  | ARBitrary
  | BWIDth
    | PORT
      | [:RESolution]
        | CONTrol
  | [RESolution]
    | CONTrol
  | COUNt
  | DELete
    | ALL
  | FREQuency
    | CENTER
    | SPAN
    | STARt
    | STOP
  | LIST
  | NFBW
    | CONTrol
  | POWer
    | ATTenuation
    | RECEiver
```
| CONTrol
| REFerence
| TEST
| [LEVEL]
| CONTrol
| SA
| DTHReshold
| CONTrol
| MTRference
| CONTrol
| MAX?
| MIN?
| VAVerage
| CONTrol
| VIDeobw
| CONTrol
| SHLO
| CONTrol
| SOURCE
| RECEiver:GAIN
| ALL
| CONTrol
| [STATE]
| SWEep
| DELay
| CONTrol
| DWELI
| CONTrol
SENSe<cnum>:SEGMenť<snum>:ADD

**Applicable Models:** All

*(Write-only)* Adds a segment.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<snum>`: Segment number to add. If unspecified, value is set to 1. Segment numbers must be sequential. If a new number is added where one currently exists, the existing segment and those following are incremented by one.

**Examples**

Two Segments exist (1 and 2). The following command will add a new segment (1). The existing (1 and 2) will become (2 and 3) respectively.

```plaintext
SENS:SEG1:ADD
sense2:segment1:add
```

**Query Syntax**

Not applicable. Use Sense:Segment:Count to determine the number of segments in a trace.

---

**See Also**

- Example: Upload and Download a Segment List
- Learn about Segment Sweep
- Synchronizing the Analyzer and Controller
- SCPI Command Tree
SENSe<cnum>:SEGMent:ARBitrary <ON | OFF>

Applicable Models: All

(Read-Write) Enables you to setup a segment sweep with arbitrary frequencies. The start and stop frequencies of each segment can overlap other segments. Also, each segment can have a start frequency that is greater than its stop frequency which causes a reverse sweep over that segment. Learn more about Arbitrary Segment Sweep.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<ON | OFF>`: ON (or 1) - Allows the setup of arbitrary segment sweep. OFF (or 0) - Prevents the setup of arbitrary segment sweep.

Examples

- SENS:SEG:M:ARB ON
- sense2:SEG:ARB OFF

Query Syntax

SENSe<cnum>:SEGMent:ARBitrary?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF

---

SENSe<cnum>:SEGMent<snum>:BWIDth:PORT<pnum>[:RESolution] <num>

Applicable Models: All

(Read-Write) Specifies whether the IF Bandwidth resolution can be set independently for each segment for the selected port and channel.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<snum>`: Segment number to modify. Choose any existing segment number.
- `<pnum>`: Individual port number of the source: Port 1 to Port 2/Port 4. If unspecified, value is set to 1.
- `<num>`: IF Bandwidth of each segment in Hz. The list of valid IF Bandwidths is different depending on the VNA model. (Click to see the lists.) If an invalid number is specified, the analyzer will round up to the closest valid number.

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.
SENSe<cnum>:SEGMen<snm>:BWIDth | BWIDth:PORT[:RESolution]:CONTrol <bool>

Applicable Models: All

(Read-Write) Turns ON or OFF the individual (Port 1 to Port n) IF Bandwidth control in the segment sweep table. This command can be turned ON when the SENS:SEGM:BWID:CONT is OFF.

Parameters
- <cnum>: Any existing channel number. If unspecified, value is set to 1
- <bool>: Specified the individual IFBW control, either ON or OFF.
  - ON or 1 - Turns ON the individual port IFBW control.
  - OFF or 0 - Turns OFF the individual port IFBW control.

Examples
- SENS:SEGm:BWID:PORT:CONT ON
- sense2:segment2:bandwidth:PORT:resolution:control off

Query Syntax
SENSe<cnum>:SEGMen<snm>:BWIDth | BWIDth:PORT[:RESolution]:CONTrol?

Return Type
Boolean

Default
OFF or 0

SENSe<cnum>:SEGMen<snm>:BWIDth[:RESolution] <num>
Applicable Models: All

(Read-Write) Sets the IF Bandwidth for the specified segment. First set SENS:SEGM:BWIDth:CONTrol ON. All subsequent segments that are added assume the new IF Bandwidth value.

Parameters
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<snum>`: Segment number to modify. Choose any existing segment number.
- `<num>`: IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the VNA model. (Click to see the lists.) If an invalid number is specified, the analyzer will round up to the closest valid number.

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples
- `SENS:SEGM:BWID:RESolution max`

Query Syntax
- `SENSe<cnum>:SEGMent<snum>:BWIDth[:RESolution]?`

Return Type
- Numeric

Default
- Varies with VNA model.

SENSe<cnum>:SEGMent:BWIDth[:RESolution]:CONTrol <ON | OFF>

Applicable Models: All

(Read-Write) Specifies whether the IF Bandwidth resolution can be set independently for each segment. This command can be turned ON when the SENS:SEGM:BWID:PORT:CONT is OFF.

Parameters
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<ON | OFF>`: ON (or 1) - turns Bandwidth control ON. Bandwidth can be independently set for each segment.
- OFF (or 0) - turns Bandwidth control OFF. Use the channel IF bandwidth setting SENS:BWID.

Examples
- `SENSe:SEGM:BWID:CONT ON`
- `sense2:segment:bwidth:control off`

Query Syntax
- `SENSe<cnum>:SEGMent:BWIDth[:RESolution]:CONTrol?`
SENSe<cnum>:SEGMent:COUNt?

Applicable Models: All

(Read-only) Queries the number of segments that exist in the specified channel.

Parameters

- **<cnum>** Any existing channel number. If unspecified, value is set to 1

Examples

SENSe:SEGM:COUNT?
sense2:segment:count?

Return Type Numeric
Default 1 segment

SENSe<cnum>:SEGment<snum>:DELete

Applicable Models: All

(Write-only) Deletes the specified sweep segment. When ALL segments are deleted, SENS:SWE:TYPE is automatically set to Linear because there are no segments to sweep.

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<snum>** Number of the segment to delete. If unspecified, value is set to 1

Examples

SENSe:SEGM:DEL
sense2:segment2:delete

Query Syntax Not applicable
Default Not Applicable

SENSe<cnum>:SEGment:DELete:ALL
Applicable Models: All

(Write-only) Deletes all sweep segments. When this command is executed, SENS:SWE:TYPE is automatically set to Linear because there are no segments to sweep.

Parameters

\(<\text{cnum}>\) Any existing channel number. If unspecified, value is set to 1

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:SEGM:DEL:ALL</td>
</tr>
<tr>
<td>sense2:segment:delete:all</td>
</tr>
</tbody>
</table>

Query Syntax

Not applicable

Default

Not Applicable

SENSe<\text{cnum}>:SEGment<\text{snum}>:FREQuency:CENTer <\text{num}>

Applicable Models: All

(Read-Write) Sets the Center Frequency for the specified segment. The Frequency Span of the segment remains the same. The Start and Stop Frequencies change accordingly.

**Note**: All previous segment's Start and Stop Frequencies that are larger than the new Start Frequency are changed to the new Start Frequency. All following segment's start and stop frequencies that are smaller than the new Stop Frequency are changed to the new Stop Frequency.

Parameters

\(<\text{cnum}>\) Any existing channel number. If unspecified, value is set to 1
\(<\text{snum}>\) Segment number to modify. Choose any existing segment number.
\(<\text{num}>\) Center Frequency in Hz. Choose any number between the **minimum** and **maximum** frequency of the analyzer.

**Note**: This command will accept **MIN** or **MAX** instead of a numeric parameter. See SCPI Syntax for more information.

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:SEGM:FREQ:CENT 1MHZ</td>
</tr>
<tr>
<td>sense2:segment2:frequency:center 1e9</td>
</tr>
</tbody>
</table>

Query Syntax

SENSe<\text{cnum}>:SEGment<\text{snum}>:FREQuency:CENTer?

Return Type

Numeric

Default

Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.
SENSe<cn>:SEGMen<snum>:FREQuency:SPAN <num>

Applicable Models: All

(Read-Write) Sets the Frequency Span for the specified segment. The center frequency of the segment remains the same. The start and stop frequencies change accordingly.

Note: All previous segment's Start and Stop Frequencies that are larger than the new Start Frequency are changed to the new Start Frequency. All following segment's start and stop frequencies that are smaller than the new Stop Frequency are changed to the new Stop Frequency.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1
<snum> Segment number to modify. Choose any existing segment number.
<num> Frequency Span in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

SENS:SEG:FREQ:SPAN 1MHZ
sense2:segment2:frequency:span max

Query Syntax

SENSe<cn>:SEGMen<snum>:FREQuency:SPAN?

Return Type

Numeric

Default

If first segment, frequency span of the analyzer. Otherwise 0.

SENSe<cn>:SEGMen<snum>:FREQuency:START <num>

Applicable Models: All

(Read-Write) Sets the Start Frequency for the specified sweep segment.

Notes

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

To return the start and stop frequency of the entire sweep (all segments), Use SENS:FREQ:STARt and SENS:FREQ:STOP

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1
<snum> Segment number to modify. Choose any existing segment number.
### SENSE<cnm>:SEGM<snm>:FREQ:STAR <num>

**Applicable Models:** All

*(Read-Write)* Sets the Start Frequency for the specified sweep segment.

**Notes**

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

To return the start and stop frequency of the entire sweep (all segments), Use SENS:FREQ:START? and SENS:FREQ:STOP?

**Parameters**

- `<cnm>`: Any existing channel number. If unspecified, value is set to 1
- `<snm>`: Segment number to modify. Choose any existing segment number.
- `<num>`: Start Frequency in Hz. Choose any number between the `minimum` and `maximum` frequency of the analyzer.

**Examples**

<table>
<thead>
<tr>
<th>SENSE&lt;cnm&gt;:SEGM&lt;snm&gt;:FREQ:STAR 1MHZ</th>
</tr>
</thead>
<tbody>
<tr>
<td>sense2:segment2:frequency:start minimum</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe<cnm>:SEGM<snm>:FREQ:STAR?

**Return Type**

Numeric

---

### SENSE<cnm>:SEGM<snm>:FREQ:STOP <num>

**Applicable Models:** All

*(Read-Write)* Sets the Stop Frequency for the specified sweep segment.

**Notes**

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

To return the start and stop frequency of the entire sweep (all segments), Use SENS:FREQ:START? and SENS:FREQ:STOP?

**Parameters**

- `<cnm>`: Any existing channel number. If unspecified, value is set to 1
- `<snm>`: Segment number to modify. Choose any existing segment number.
- `<num>`: Stop Frequency in Hz. Choose any number between the `minimum` and `maximum` frequency of the analyzer.

**Examples**

<table>
<thead>
<tr>
<th>SENSE&lt;cnm&gt;:SEGM&lt;snm&gt;:FREQ:STOP 1MHZ</th>
</tr>
</thead>
<tbody>
<tr>
<td>sense2:segment2:frequency:stop maximum</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe<cnm>:SEGM<snm>:FREQ:STOP?

**Return Type**

Numeric
Default  If first segment, stop frequency of the analyzer. Otherwise, start frequency of the segment.

SENSe<cnum>:SEGMent:LIST <char>,<numSegs>,<data>

Applicable Models: All

(Read-Write) Reads or writes the entire list of values in the segment sweep table.

Note: For binary data transfer, specify 64-bit instead of 32-bit using FORMat[:DATA]. This is because higher frequencies used on VNA exceed the maximum value that can be represented by a 32-bit floating point number.

When sending/receiving this data as binary (FORMat[:DATA] REAL,64), use FORMat:BORDer to specify the correct 'endianness' (byte ordering) corresponding to your programming environment / computer platform.

Parameters

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<char>** Choose from:
  - **SSTOP** - Frequency values are Start and Stop for each segment.
  - **CSPAN** - Frequency values are Center and Span for each segment.
- **<numSegs>** Total number of sweep segments being input. This allows the VNA to determine how many values per-each-segment are in the input <data> block.
- **<data>** A list of segments specified using either a comma-separated string of data, or an array of double (real,64) depending on the state of FORM:DATA. Each segment is specified with a minimum of 4 and maximum of 7 values consecutively. The set of values that specify each segment should be in the following order:

  1. Segment state (Boolean 1 for ON and 0 for OFF)
  2. Number of Points in the segment
  3. Start Freq (when <char> is SSTOP), or Center Freq (when <char> is CSPAN)
  4. Stop Freq (when <char> is SSTOP), or Freq Span (when <char> is CSPAN)
  5. IFBW (optional for the Write)
  6. Dwell Time (optional for the Write)
  7. Power (optional for the Write) - see below.
The first four data elements must always be supplied. After those values, data must be supplied for successive optional elements. For example, to set dwell time values, you must also supply IFBW values, because IFBW (#5) precedes dwell time (#6) in the array order.

The IF Bandwidth, Sweep Time and Source Power Control settings do NOT affect the order in which elements are interpreted.

The number of elements to supply for Power depends on the following two settings:

1. Source Power Option - ON allows segments to have independent power levels.
2. Couple Ports = Off allows different power levels for each test port.

<table>
<thead>
<tr>
<th>CouplePorts</th>
<th>SourcePowerOption</th>
<th>Number of Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>Each port has its own channel-wide power setting, which is set using SOURce:POWer[:LEVel]. Provide exactly 7 elements per segment. The last element (power) is ignored.</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>Provide 6 elements + total number of ports. The first 7 elements are still interpreted the same. The remaining elements (in-order) are interpreted as the power levels to set on that segment for Ports 2 through N, where N is the total number of ports currently enabled for the VNA or for a VNA with multiport external test set.</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>Provide exactly 7 elements per segment. The last element (power) is ignored.</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>Provide exactly 7 elements per segment. The last element (power) is honored.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:SEGM:LIST SSTOP,1,1,201,10E6,26.5E9,1E3,0,-10 1 segment,
```
state ON, 201 points, 10 MHz to 26.5 GHz, 1kHz IFBW, 0 dwell time, -10 dBm (port powers coupled)

sense2:segment:list? cspan

See Upload and Download a Segment List example program

Query Syntax
SENS<cnm>:SEGM<snm>:LIST? [char].

If unspecified, char is set to SSTOP.

The number of data elements per segment returned will be 6 + total number of source ports, regardless of the IF Bandwidth, Sweep Time and Source Power Control settings. For the N5264B, which has no source ports, the query will return just 6 values per segment. For all other VNA models, the last elements in each segment correspond to the power level for each port.

Return Type
Returns block data in the format specified by FORMat[:DATA].

Default
Not Applicable

SENS<cnm>:SEGM<snm>:NFBW <num>

Applicable Models: All with S93070xA/B or S95070A/B

(Read-Write) Sets or returns the noise figure bandwidth.

Parameters
- <cnm> Any existing channel number. If unspecified, value is set to 1
- <snm> Segment number to modify. Choose any existing segment number.
- <num> Noise figure bandwidth. Choose any number between the minimum and maximum IF bandwidth of the analyzer.

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples
SENS:SEG:M:NFBW MIN
sense2:segment2:nfbw maximum

Query Syntax
SENS<cnm>:SEGM<snm>:NFBW?

Return Type
Numeric

Default
If first segment, stop frequency of the analyzer. Otherwise, start frequency of the segment.

SENS<cnm>:SEGM<snm>:NFBW:CONTrol <bool>
Applicable Models: All with S93070xA/B or S95070A/B

(Read-Write) Turns ON or OFF the noise figure bandwidth setting specified using SENSE:SEGMENT:NFBW.

Parameters
<cnm> Any existing channel number. If unspecified, value is set to 1.
<snum> Segment number to modify. Choose any existing segment number.
<bool> ON or 1 - Turns ON the noise figure bandwidth control.
OFF or 0 - Turns OFF the noise figure bandwidth control

Examples
SENSe:SEGm:NFBW:CONt ON
sense:segment:nfbw:control 1

Query Syntax
SENSe<cnm>:SEGm:POW:ATT:REC:CONT?

Return Type
Boolean.

Default
OFF or 0

SENSe<cnm>:SEGm:POW:ATT:REC:REC:CONT <bool>

Applicable Models: N522xB, N523xB, N524xB, M9485A

(Read-Write) Turns ON or OFF the individual receiver attenuator control in the segment sweep table.

Parameters
<cnm> Any existing channel number. If unspecified, value is set to 1.
<bool> ON or 1 - Turns ON the individual receiver attenuator control.
OFF or 0 - Turns OFF the individual receiver attenuator control

Examples
SENSe:SEGm:POW:ATT:REC:REC:CONt ON
sense:segment:power:attenuation:receiver:control 1

Query Syntax
SENSe<cnm>:SEGm:POW:ATT:REC:REC:CONT?

Return Type
Boolean. If querying for the standard (M9376A) port, the return value is 0

Default
OFF or 0

SENSe<cnm>:SEGm<snum>:POW<port>:ATT:REC:REF <num>
Applicable Models: N522xB, N523xB, N524xB, M9485A

(Read-Write) Sets the attenuation level for the specified reference attenuator for each segment.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<snum>`: Segment number to modify. Choose any existing segment number. If unspecified, value is set to 1.
- `<port>`: Port number of the VNA. If unspecified, value is set to 1.
- `<num>`: Attenuation value in dB. 0dB or 35dB.
  
  If a number other than these is entered, the analyzer will select the next lower valid value. For example, if 19dB is entered, then 0dB attenuation will be selected.

**Examples**

```
SENS:SEGM:POW2:ATT:REC:REF 0
```

**Query Syntax**

```
SENS<cnum>:SEGM<snum>:POW<port>:ATT<nnum>:REC:REF
```

**Return Type**

Numeric. If querying for the standard port, the return value is 0

**Default**

35

---

SENSe<cnum>:SEGM<snum>:POW<port>:ATT<nnum>:REC:TEST <num>

Applicable Models: N522xB, N523xB, N524xB, M9485A

(Read-Write) Sets the attenuation level for the specified test attenuator for each segment.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<snum>`: Segment number to modify. Choose any existing segment number. If unspecified, value is set to 1.
- `<port>`: Port number of the VNA. If unspecified, value is set to 1.
- `<num>`: Attenuation value in dB. 0dB or 35dB.

  If a number other than these is entered, the analyzer will select the next lower valid value. For example, if 19dB is entered, then 0dB attenuation will be selected.
SENSe<cnum>:SEGMent<snum>:POWer<port>[:LEVEL] <num>

**Applicable Models:** All

*(Read-Write)* Sets the Port Power level for the specified sweep segment. First set SENSe:SEGM:POW:CONTrol ON.

When port power is Coupled, setting port power for one port will apply port power for all source ports.

All subsequent segments that are added assume the new Power Level value.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<snum>`: Segment number to modify. Choose any existing segment number.
- `<port>`: Port number of the source. If unspecified, value is set to 1.
- `<num>`: Power level.

**Note:** The range of settable power values depends on the VNA model and if source attenuators are installed. To determine the range of values, send SOUR:POW? MAX and SOUR:POW? MIN. (SOUR:POW:ATT:AUTO must be set to ON).

Actual achievable leveled power depends on frequency.

<table>
<thead>
<tr>
<th>Examples</th>
<th>SENSe:SEGM:POW 0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sense2:segment2:power1:level -10</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe<cnum>:SEGMent<snum>:POWer<port>[:LEVEL]?

**Return Type**

Numeric

**Default**

0
SENSe<cnm>:SEGMent:POWer[:LEVel]:CONTrol <ON | OFF>

Applicable Models: All

(Read-Write) Specifies whether Power Level can be set independently for each segment.

Parameters

<cnm> Any existing channel number. If unspecified, value is set to 1
<ON | OFF> ON (or 1) - turns Power Level control ON. Power level can be set for each segment.
OFF (or 0) - turns Power Level control OFF. Use the channel power level setting.

Examples

SENSE:SEGM:POW:CONT ON
sense2:segment:power:level:control off

Query Syntax

SENSe<cnm>:SEGMent:POWer[:LEVel]:CONTrol?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF

SENSe<cnm>:SEGMent<snum>:SA:DTHReshold <num>

Applicable Models: All

(Read-Write) Sets or returns the SA data threshold for the segment.

Parameters

<cnm> Any existing channel number. If unspecified, value is set to 1
<snum> Segment number to modify. Choose any existing segment number.
<num> Data threshold (in dBm).

Examples

SENSE:SEGM:SA:DTHR -60
sense2:segment2:sa:dthreshold -60

Query Syntax

SENSe<cnm>:SEGMent<snum>:SA:DTHReshold?

Return Type

Numeric

Default

-60 dBm

SENSe<cnm>:SEGMent<snum>:SA:DTHReshold:CONTrol <ON | OFF>
Applicable Models: All

(Read-Write) Specifies whether SA Data Threshold can be set independently for each segment.

Parameters
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<snum>`: Segment number to modify. Choose any existing segment number.
- `<ON | OFF>`: ON (or 1) - turns SA Data Threshold control ON.
  OFF (or 0) - turns SA Data Threshold control OFF.

Examples
- `SENS:SEGM:SA:DTHR:CONT ON`
- `sense2:segment:sa:dthreshold:control off`

Query Syntax
- `SENSe<cnum>:SEGMent<snum>:SA:DTHReshold:CONTrol?`

Return Type
- Boolean (1 = ON, 0 = OFF)

Default
- OFF

SENSe<cnum>:SEGMent<snum>:SA:MTReference <num>

Applicable Models: All

(Read-Write) Sets or returns the SA multitone reference for the segment.

Parameters
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<snum>`: Segment number to modify. Choose any existing segment number.
- `<num>`: Multitone reference (in dBm).

Examples
- `SENS:SEGM:SA:MTR 0`
- `sense2:segment2:sa:mtreference 0`

Query Syntax
- `SENSe<cnum>:SEGMent<snum>:SA:MTReference?`

Return Type
- Numeric

Default
- 0

SENSe<cnum>:SEGMent<snum>:SA:MTReference:CONTrol <ON | OFF>
Applicable Models: All

(Read-Write) Specifies whether SA Reference Tone can be set independently for each segment.

Parameters
<cnun> Any existing channel number. If unspecified, value is set to 1
<snum> Segment number to modify. Choose any existing segment number.
<ON | OFF> ON (or 1) - turns SA Reference Tone control ON.
OFF (or 0) - turns SA Reference Tone control OFF.

Examples
SENS:SEGM:SA:MTR:CONT ON
sense2:segment:sa:mtreference:control off

Query Syntax
SENS<cnun>:SEGM<snun>:SA:MTRReference:CONTrol?

Return Type
Boolean (1 = ON, 0 = OFF)

Default
OFF

SENSe<cnun>:SEGM<snun>:SA:MTRReference:MAX?

Applicable Models: All

(Read-only) Queries the maximum value of the SA Reference Tone, which is the maximum frequency.

Parameters
<cnun> Any existing channel number. If unspecified, value is set to 1
<snum> Segment number to modify. Choose any existing segment number.

Examples
SENS:SEGM:SA:MTR:MAX?
sense2:segment:sa:mtreference:max?

Return Type
Numeric

Default
Not Applicable

SENSe<cnun>:SEGM<snun>:SA:MTRReference:MIN?
Applicable Models: All

(Read-only) Queries the minimum value of the SA Reference Tone.

**Parameters**
- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<snum>`  Segment number to modify. Choose any existing segment number.

**Examples**
- `SENS:SEGM:SA:MTR:MIN?`
- `sense2:segment:sa:mtrreference:min?`

**Return Type**  Numeric

**Default**  0

SENSe<cnum>:SEGMent<snum>:SA:VAVerage <num>

Applicable Models: All

(Read-Write) Sets or returns the SA vector averaging points for the segment.

**Parameters**
- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<snum>`  Segment number to modify. Choose any existing segment number.
- `<num>`  Vector average points.

**Examples**
- `SENS:SEGM:SA:VAV 10`
- `sense2:segment2:sa:vaverage 10`

**Query Syntax**  `SENSe<cnum>:SEGMent<snum>:SA:VAVerage?`

**Return Type**  Numeric

**Default**  1

SENSe<cnum>:SEGMent<snum>:SA:VAVerage:CONTrol <ON | OFF>`
Applicable Models: All

(Read-Write) Specifies whether SA Vector Averaging can be set independently for each segment.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<snum>`: Segment number to modify. Choose any existing segment number.
- `<ON | OFF>`
  - ON (or 1) - turns SA Vector Averaging control ON.
  - OFF (or 0) - turns SA Vector Averaging control OFF.

**Examples**

```plaintext
SENS:SEGM:SA:VAV:CONT ON
sense2:segment:sa:vaverage:control off
```

**Query Syntax**

`SENSe<cnum>:SEGMent<snum>:SA:VAVerage:CONTrol?`

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

---

`SENSe<cnum>:SEGMent<snum>:SA:VIDeobw <num>`

Applicable Models: All

(Read-Write) Sets or returns the SA video bandwidth for the segment.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<snum>`: Segment number to modify. Choose any existing segment number.
- `<num>`: Video bandwidth (in Hz).

**Examples**

```plaintext
SENS:SEGM:SA:VID 1E6
sense2:segment2:sa:videobw 1e6
```

**Query Syntax**

`SENSe<cnum>:SEGMent<snum>:SA:VIDeobw?`

**Return Type**

Numeric

**Default**

1E6 Hz

---

`SENSe<cnum>:SEGMent<snum>:SA:VIDeobw:CONTrol <ON | OFF>`
Applicable Models: All

(Read-Write) Specifies whether SA Video Bandwidth can be set independently for each segment.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Segment number to modify. Choose any existing segment number.
- `<ON | OFF>` ON (or 1) - turns SA Video Bandwidth control ON.
  OFF (or 0) - turns SA Video Bandwidth control OFF.

Examples

```
SENS:SEGM:SA:VID:CONT ON
sense2:segment:sa:videobw:control off
```

Query Syntax

```
SENSe<cnum>:SEGMent<snum>:SA:VIDeobw:CONTrol?
```

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF

SENSe<cnum>:SEGMent<snum>[:STATe] <ON | OFF>

Applicable Models: All

(Read-Write) Turns the specified sweep segment ON or OFF. At least ONE segment must be ON or Sweep Mode is automatically set to Linear.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Segment number to be turned ON or OFF
- `<ON | OFF>` ON (or 1) - turns segment ON.
  OFF (or 0) - turns segment OFF.

Examples

```
SENS:SEGm ON
sense2:segment2:state off
```

Query Syntax

```
SENSe<cnum>:SEGMent<snum>[:STATe]?
```

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF

SENSe<cnum>:SEGMent<snum>:SHLO

Applicable Models: N522xB, N523xB, N524xB, E5080A
(Read-Write) Sets or returns the Shift LO state of each segment in the segment sweep table for the selected channel.

**Notes:** The SENS:SEGM:SHLO:CONT command must first be set to ON before using this command.

### Parameters
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Any existing segment number.

### Examples
- `SENS:SEGM2:SHLO`
- `sense2:segment2:shlo`

### Query Syntax
- `SENSe<cnum>:SEGMent<snum>:SHLO?`

### Return Type
- Numeric

### Default
- Not Applicable

---

`SENSe<cnum>:SEGMent:<snum>:SOURce<sport>:RECeiver<rport>:GAIN[:VALue] <string>`

**Applicable Models:** M980xA, P50xxA

(Read-Write) Sets the gain settings to a specified port on the specified sweep segment. SENS:SEGM:SOUR:REC:GAIN:CONT should be turned on when you us this. Use SENS:SOUR:REC:GAIN:CAT? to return a list of available gain states for the specified port.

### Parameters
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Segment number to modify. Choose any existing segment number. If unspecified, value is set to 1.
- `<sport>` Source port number of the VNA. If unspecified, value is set to 1
- `<rport>` Receiver port number of the VNA. If unspecified, value is set to 1
- `<string>` Receiver gain state. Not case sensitive.

For M980xA, P50xxA choose from:

- Auto
- Low
- High

### Examples
SENSe<ch>:SEGMen<seg>:SOUR<port>:RECeiver<rport>:GAIN[:VALue]

**Query Syntax**
SENSe<ch>:SEGMen<seg>:SOUR<port>:RECeiver<rport>:GAIN[:VALue]

**Return Type**
String

**Default**
Auto

SENSe<ch>:SEGMen<seg>:SOUR<port>:RECeiver<rport>:GAIN<state>

**Applicable Models:** M980xA, P50xxA

*(Read-Write)* Sets the gain settings to all ports on the specified sweep segment. SENSe:SEGMen:SOUR:RECeiver:GAIN:CONT should be turned on when you use this. Use SENSe:SOUR:RECeiver:GAIN:CAT? to return a list of available gain states for the specified port.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<seg>` Segment number to modify. Choose any existing segment number. If unspecified, value is set to 1.
- `<port>` This parameter is Ignored
- `<rport>` This parameter is Ignored
- `<state>` Receiver gain state. Not case sensitive.

For M980xA, P50xxA choose from:

- Auto
- Low
- High

**Examples**

SENSe:SEGMen2:SOUR1:RECeiver2:GAIN "Low" ' Low for S21 measurement in segment 2

SENSe:SEGMen2:SOUR1:RECeiver2:GAIN:"High" ' High for S22 measurement in Segment 1

SENSe:SEGMen2:SOUR1:RECeiver2:GAIN:CONT ON
SENSe:SEGMen2:SOUR1:RECeiver2:GAIN:ALL "Low" ' Low for all measurements in Segment 2
SENSe<cnum>:SEGMent<snum>:SOURce<port>:RECeiver<rport>:GAIN:CONTrol <bool>

**Applicable Models:** M980xA, P50xxA

*(Read-Write)* Sets and read the status of the segment receiver gain setting function. This must be turned ON when SENS:SEGM:SOUR:REC:GAIN or SENS:SEGM:SOUR:REC:GAIN:ALL is used.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<snum>`: This parameter is Ignored.
- `<sport>`: This parameter is Ignored.
- `<rport>`: This parameter is Ignored.
- `<bool>`: ON or 1 - Turn the segment receiver gain setting function ON
  
  OFF or 0 - Turn the segment receiver gain setting function OFF

**Examples**

```
SENSe<cnm>:SEGMent<snum>:SOURce<port>:RECeiver<rport>:GAIN:CONT on
SENSe<cnm>:SEGMent<snum>:SOURce<port>:RECeiver<rport>:GAIN:CONT OFF
```

**Query Syntax**

SENSe<cnum>:SEGMent<snum>:SOURce<port>:RECeiver<rport>:GAIN:CONTrol ?

**Return Type**

Boolean

**Default**

OFF

SENSe<cnum>:SEGMent<snum>:SHLO:CONTrol <bool>

**Applicable Models:** N522xB, N523xB, N524xB, E5080A
(Read-Write) Turns ON or OFF the individual Shift LO state control in the segment sweep table.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Any existing segment number.
- `<bool>` **ON or 1** - Turns ON the individual Shift LO state control.
  **OFF or 0** - Turns OFF the individual Shift LO state control.

**Examples**

```plaintext
SENS:SEG:M:SHLO:CONT ON  
sense2:segment2:shlo:control off
```

**Query Syntax**

`SENS<cnum>:SEGment<snum>:SHLO:CONTrol?`

**Return Type**

Boolean

**Default**

OFF or 0

---

`SENS<cnum>:SEGment<snum>:SWEep:DELay <num>`

**Applicable Models:** All

(Read-Write) Sets or returns the sweep delay time of the specified sweep segment.

**Notes:** The SENS:SEG:M:SWE:DEL:CONT command must first be set to ON before using this command.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Any existing segment number.
- `<num>` Range of sweep delay time is between 0 to 1 and the resolution is 0.001.

**Notes:** If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

- `<unit>` s (second)

**Examples**

```plaintext
SENS:SEG:M:SWE:DEL  
sense2:segment2:sweep:delay
```

**Query Syntax**

`SENS<cnum>:SEGment<snum>:SWEep:DELay?`

**Return Type**

Numeric / Double precision floating point
**SENSe<cnum>:SEGMen<snum>:SWEep:DELay:CONTrol <bool>**

Applicable Models: All

(Read-Write) Turns ON or OFF the sweep delay time of the specified sweep segment.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Any existing segment number.
- `<bool>` ON or 1 - Turns ON sweep delay time.
  OFF or 0 - Turns OFF sweep delay time.

**Examples**
```
SENS:SEGM:SWE:DEL:CONT ON
sense2:segment2:sweep:delay:control off
```

**Query Syntax**
SENSe<cnum>:SEGMen<snum>:SWEep:DELay:CONTrol?

**Return Type**
Boolean

**Default**
OFF or 0

---

**SENSe<cnum>:SEGMen<snum>:SWEep:DWELl <num>**

Applicable Models: All

(Read-Write) Sets or returns the sweep dwell time of the specified sweep segment.

**Notes:** The SENS:SEGm:SWE:DWELl:CONT command must first be set to ON before using this command.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Any existing segment number.
- `<num>` Range of sweep dwell time

**Notes:** If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

- `<unit>` s (second)

**Examples**
```
SENS:SEGM:SWE:DWEL
sense2:segment2:sweep:dwell
```
SENSe <cnum>:SEGMen<snum>:SWEep:DWELl:CONTrol <bool>

Applicable Models: All

(Read-Write) Turns ON or OFF the sweep dwell time of the specified sweep segment.

Parameters
- <cnum> Any existing channel number. If unspecified, value is set to 1
- <snum> Any existing segment number.
- <bool> ON or 1 - Turns ON sweep dwell time.
  OFF or 0 - Turns OFF sweep dwell time.

Examples
- SENS:SEGM:SWE:DWEL:CONT ON
- sense2:segment2:sweep:dwell:control off

Query Syntax
SENSe <cnum>:SEGMen<snum>:SWEep:DWELl:CONTrol?

Return Type
Boolean

Default
OFF or 0

SENSe <cnum>:SEGMen<snum>:SWEep:GENeration <char>
Applicable Models: All

(Read-Write) Sets or returns the sweep mode of the specified sweep segment.

Notes: The SENS:SEGM:SWE:GEN:CONT command must first be set to ON before using this command.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<snum>` Any existing segment number.
- `<char>` Select sweep mode from either of the following:
  - "ANALog": Sets the sweep mode to the auto mode.
  - "STEPped": Sets the sweep mode to the stepped mode.

Examples

SENSe<cnum>:SEGM<snuint>:SWEep:GENeration ANAL
sense2:segment2:sweep:generation stepped

Query Syntax
SENSe<cnum>:SEGM<snuint>:SWEep:GENeration?

Return Type
Character

Default
"ANAL"

SENSe<cnum>:SEGM<snuint>:SWEep:GENeration:CONTrol <bool>

Applicable Models: All

(Read-Write) Turns ON or OFF the sweep mode of the specified sweep segment.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<snum>` Any existing segment number.
- `<bool>` ON or 1 - Turn ON sweep mode.
  OFF or 0 - Turn OFF sweep mode.

Examples

SENSe<cnum>:SEGM<snuint>:SWEep:GENeration:CONTrol ON
sense2:segment2:sweep:generation:control off

Query Syntax
SENSe<cnum>:SEGM<snuint>:SWEep:GENeration:CONTrol?

Return Type
Boolean

Default
OFF or 0
SENSe<cnum>:SEGMen<tn>:SWEep:POINts <num>

Applicable Models: All

(Read-Write) Sets the number of data points for the specified sweep segment.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <snum> Any existing segment number. If unspecified, value is set to 1
- <num> Number of points in the segment. The total number of points in all segments cannot exceed 20001. A segment can have as few as 1 point.

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

```
SENS:SEG:M:SWE:POIN 51
sense2:segment2:sweep:points maximum
```

Query Syntax

SENSe<cnum>:SEGMen<tn>:SWEep:POINts?

Return Type Numeric

Default 21

---

SENSe<cnum>:SEGMen<tn>:SWEep:POINts:TOTal? <totalPoints>

Applicable Models: All

(Read-only) Queries the total point count from the active segments or from all segments.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <snum> Any existing segment number.
- <totalPoints> Choose from:

  - **ACTive** - Returns the total point count of the active segments.
  - **ALL** - Returns the total point count of all segments.

Examples

```
sense2:segment:scan:points:total? all
```

Return Type Numeric

Default 21
SENSe<cnum>:SEGMent<snum>:SWEep:TIME <num>

Applicable Models: All

(Read-Write) Sets the time the analyzer takes to sweep the specified sweep segment.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Any existing segment number.
- `<num>` Sweep time in seconds. Choose a number between 0 and 100

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

```
SENS:SEGM:SWE:TIME 1ms
sense2:segment2:sweep:time .001
```

Query Syntax

```
SENSe<cnum>:SEGMent<snum>:SWEep:TIME?
```

Return Type

Numeric

Default

Not Applicable

SENSe<cnum>:SEGMent<snum>:SWEep:TIME:CONTrol <ON | OFF>

Applicable Models: All

(Read-Write) Specifies whether Sweep Time can be set independently for each sweep segment.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<ON | OFF>` ON (or 1) - turns Sweep Time control ON. Sweep Time can be set for each segment.
  
  OFF (or 0) - turns Sweep Time control OFF. Uses the channel Sweep Time setting.

Examples

```
SENS:SEGM:SWE:TIME:CONT ON
sense2:segment:sweep:time:control off
```

Query Syntax

```
SENSe<cnum>:SEGMent<snum>:SWEep:TIME:CONTrol?
```

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF

SENSe<cnum>:SEGMent<snum>:SWEep:TIME:TOTal? <totalTime>
**Applicable Models:** All

*(Read-only)* Queries the total sweep time of the active segments or of all segments.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Any existing segment number.
- `<totalTime>` Choose from:
  - **ACTive** - Returns the total sweep time of the active segments.
  - **ALL** - Returns the total sweep time of all segments.

**Examples**

```plaintext
sense2:segment:sweep:time:total? all
```

**Return Type** Numeric

**Default** 0

---

**SENSe<cnum>:SEGMent:X:SPACing <char>**

**Applicable Models:** All

*(Read-Write)* Sets X-axis spacing ON or OFF

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>`
  - **LINear** - turns X-axis point spacing OFF
  - **OBASe** - turns X-axis point spacing ON

**Examples**

```plaintext
SENS:SEGM:X:SPACing LIN
sense2:segment:spacing obase
```

**Query Syntax** `SENSe<cnum>:SEGMent:X:SPACing?`

**Return Type** Character

**Default** LINear
SENSe:SOUrce Command

SENSe:SOURce
  | PLLBandwidth
    | CATalog
    | [:VALue]
  | RECeiver
    | GAIN
    | ALL[:VALUE]
    | CATalog
    | LIST

- SCPI Command Tree

SENSe<cnum>:SOURce:PLLBandwidth:CATalog?

Applicable Models: M980xA, P50xxA, E5080B

(Read-Write) Lists PLL bandwidth type for SENSe<cnum>:SOURce:PLLBandwidth[:VALue].

Parameters
- <cnum> Any existing channel number. If unspecified, value is set to 1

Examples
- SENSe:SOUR1:PLL:CAT?
- sense:source2:pllbandwidth:catalog?

Query Syntax
SENSe<cnum>:SOURce:PLLBandwidth:CATalog?

Return Type
- <string>, available setting list with comma separated chars

Default
- Not applicable

SENSe<cnum>:SOURce:PLLBandwidth[:VALue] <type>
Applicable Models: M980xA, P50xxA, E5080B

(Read-Write) Sets and read the settings for PLL bandwidth for RF source and LO.

Parameters
- <cnum> Any existing channel number. If unspecified, value is set to 1
- <type> choose from:
  - AUTO: Use an appropriate PLL bandwidth automatically.
  - NARRow: Use the narrow PLL bandwidth. The phase noise of the RF and local sources decreases while the cycle time increases.

Examples
```
SENSe:SOUR1:PLL B NARR
sense:source2:pllbandwidth narrow
```

Query Syntax
```
SENSe<cnum>:PLLBandwidth[:VALue]?
```

Return Type
 Enumeration
 Default
 AUTO

SENSe<cnum>:SOURce<sport>:RECeiver<rport>:GAIN[:VALue] <string>

Applicable Models: M980xA, P50xxA, E5080B

(Read-Write) Sets the gain settings to a specified port. Use SEN:SOUR:REC:GAIN:CAT? to return a list of available gain states for the specified port.

Parameters
- <cnum> Any existing channel number. If unspecified, value is set to 1
- <sport> Source port number of the VNA. If unspecified, value is set to 1
- <rport> Receiver port number of the VNA. If unspecified, value is set to 1
- <string> Receiver gain state. Not case sensitive.

choose from:
- Auto
- Low
- High
SENSe<cnum>:SOURce<sport>:RECeiver<rport>:GAIN:ALL[:VALue] <string>

Applicable Models: M980xA, P50xxA, E5080B

*(Read-Write)* Sets the gain settings to all ports. Use SENSe:SOUR:REC:GAIN:CAT? to return a list of available gain states for the VNA.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<sport>`: Ignored
- `<rport>`: Ignored
- `<string>`: Receiver gain state. Not case sensitive.

choose from:

- Auto
- Low
- High

**Examples**

SENSe:SOUR1:REC2:GAIN "Low" ' Low for all measurements
SENSe:SOUR2:REC2:GAIN:ALL:value "High" ' High for all measurements

SENSe<cnum>:SOURce<sport>:RECeiver<rport>:GAIN:CATalog

SENSe<cnum>:SOURce<sport>:RECeiver<rport>:GAIN:CATalog
Applicable Models: M980xA, P50xxA, E5080B

(Read only) Reads a list of valid state for the receiver gain on the specified port.

Parameters
  <cnum>  Any existing channel number. If unspecified, value is set to 1
  <sport> Source port number of the VNA. If unspecified, value is set to 1
  <rport> Receiver port number of the VNA. If unspecified, value is set to 1

Examples
SENSe:SOUR1:REC2:GAIN:CAT?
  sense:source2:receiver2:gain:catalog?

Query Syntax  SENSE<cnum>:SOURce<sport>:RECeiver<rport>:GAIN:CATalog?
Return Type  String (Comma-separated list of strings.) ("Auto,Low,High" is returned.)
Default  Auto

SENSe<cnum>:SOURce<sport>:RECeiver<rport>:GAIN:LIST?

Applicable Models: M980xA, P50xxA, E5080B

(Read only) Gets the data array for actual gain setting of each measurement points. High: 1, Low: 0. The number of returned data is the same as the NOP.

Parameters
  <cnum>  Any existing channel number. If unspecified, value is set to 1
  <sport> Source port number of the VNA. If unspecified, value is set to 1
  <rport> Receiver port number of the VNA. If unspecified, value is set to 1

Examples
SENSe:SOUR1:REC2:GAIN:LIST?
  sense:source2:receiver2:gain:list?

Query Syntax  SENSE<cnum>:SOURce<sport>:RECeiver<rport>:GAIN:LIST?
Return Type  Data block
Default  Not Applicable
**Sense:Sweep Commands**

Specifies the sweep functions of the analyzer.

```plaintext
SENSe:SWEep:
  BLOCKed
  DWELI
    | AUTO
    | SDELay
  GENeration
    | POINtsweep
  GROups
    | COUNt
  LFEXtension:STATe
  MODE
  POINts
  PULSe More commands
  SLOCal
    | MAXimum
    | STATe
  SPEed
  SRCPort
  STEP
  TIME
    | AUTO
    | START?
    | [:STOP]
  TRIGger
    | DELAY
```
SENSe<cnum>:SWEep:BLOCked?

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, M9485A, P937xA

*(Read-only)* Reads whether the specified channel is currently 'blocked' from sweeping. Learn more about the Mechanical Devices dialog.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Examples**
- `SENSe:SWEep:BLOC?
  sense2:sweep:blocked?`

**Return Type**
- Boolean
  - 0 - No, the channel is NOT blocked.
  - 1 - Yes, the channel is blocked.

**Default** N/A

SENSe<cnum>:SWEep:DWELl <num>
Applicable Models: All

(Read-Write) Sets the dwell time between each sweep point.

- Dwell time is ONLY available with SENSE:SWEep:GENeration set to STEPped; It is Not available in ANALOG.
- Sending dwell = 0 is the same as setting SENS:SWE:DWEL:AUTO ON. Sending a dwell time > 0 sets SENS:SWE:DWEL:AUTO OFF.

Parameters
- <cnum> Any existing channel number. If unspecified, value is set to 1
- <num> Dwell time in seconds.

This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples
- SENS:SWE:DWEL .1
- sense2:sweep:dwell min

Query Syntax
- SENSE<cnum>:SWEep:DWELl?

Return Type
- Numeric
- Default 0 - (Note: dwell time set to 0 is the same as dwell:auto ON)

SENSe<cnum>:SWEep:DWELl:AUTO <ON | OFF>

Applicable Models: All

(Read-Write) Specifies whether or not to automatically calculate and set the minimum possible dwell time. Setting Auto ON has the same effect as setting dwell time to 0.

Parameters
- <cnum> Any existing channel number. If unspecified, value is set to 1
- <ON | OFF> ON (or 1) - turns dwell ON.
  - OFF (or 0) - turns dwell OFF.

Examples
- SENS:SWE:DWEL:AUTO ON
- sense2:sweep:dwell:auto off

Query Syntax
- SENSE<cnum>:SWEep:DWELl:AUTO?

Return Type
- Boolean (1 = ON, 0 = OFF)
- Default ON
SENSe<cnum>:SWEep:DWEL:Dwell:SDELay <num>

Applicable Models: All

(Read-Write) Specifies the time to wait just before acquisition begins for each sweep. This delay is in addition to Dwell Time and the following two External Trigger delays if enabled.

- **Trig:Delay** (global scope)
- **Sens:Swe:Trig:Delay** (channel scope)

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Sweep delay in seconds.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```
SENSe:SWEep:DWEL:Dwell:SDELay 1
sense2:sweep:dwell:sdelay .5
```

**Query Syntax**

SENSe<cnum>:SWEep:DWEL:Dwell:SDELay?

**Return Type** Numeric

**Default** 0

---

SENSe<cnum>:SWEep:GENeration <char>

Applicable Models: All

(Read-Write) Sets sweep as Stepped or Analog.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Choose from:
  - **STEPped** - source frequency is CONSTANT during measurement of each displayed point. More accurate than ANALog. Dwell time can be set in this mode.
  - **ANALog** - source frequency is continuously RAMPING during measurement of each displayed point. Faster than STEPped. Sweep time (not dwell time) can be set in this mode.
SENSe<cnum>:SWEep:GENeration:POINtsweep <bool>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Turns ON and OFF point sweep mode. When enabled, the VNA measures both the forward and reverse parameters at each frequency point before stepping to the next frequency. Learn more.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Choose from:
  - **ON** or (1) - Enable point sweep mode.
  - **OFF** or (0) - Disable point sweep mode.

**Examples**

SENSe:SWE:GEN:POIN 1

sense2:sweep:generation:pointsweep off

**Query Syntax**

SENSe<cnum>:SWEep:GENeration:POINtsweep?

**Return Type**

Boolean

**Default**

OFF

SENSe<cnum>:SWEep:GROups:COUNt <num>
Applicable Models: All

(Read-Write) Sets the trigger count (groups) for the specified channel. Set trigger mode to group after setting this count.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Count (groups) number. Choose any number between: 1 and 2e6 (1 is the same as single trigger)

Examples

```
SENS:SWE:GRO:COUN 10
sense2:sweep:groups:count 50
```

Query Syntax

```
SENSe<cnum>:SWEep:GROups:COUNt?
```

Return Type

Numeric

Default

1

---

SENSe<cnum>:SWEep:LFEXtension:STATe <bool>

Applicable Models: N5222B, N5227B, N5242B, N5247B, N5290A, N5291A

(Read-Write) Turns ON or OFF low frequency extension for extending the range down to 1 kHz start frequency.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<bool>`: Choose from:

  - **ON or 1** - Turns ON low frequency extension.
  - **OFF or 0** - Turns OFF low frequency extension.

Examples

```
SENS:SWE:LFEX:STAT ON
sense2:sweep:lfextension:state off
```

Query Syntax

```
SENSe<cnum>:SWEep:LFEXtension:STATe?
```

Return Type

Boolean

Default

OFF or 0

---

SENSe<cnum>:SWEep:MODE <char>
Applicable Models: All

(Read-Write) Sets the number of trigger signals the specified channel will ACCEPT.

See Triggering the VNA Using SCPI.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<char>`: Trigger mode. Choose from:

  **HOLD** - channel will not trigger
  
  **CONTinuous** - channel triggers indefinitely
  
  **GROups** - channel accepts the number of triggers specified with the last SENS:SWE:GRO:COUN `<num>`. This is one of the VNA overlapped commands. Learn more.
  
  **SINGle** - channel accepts ONE trigger, then goes to HOLD.

Note: To perform simple, single-triggering, use SINGle which requires that TRIG:SOURce remain in the default (internal) setting.

Examples

```
SENS:SWE:MODE CONT
sense2:sweep:mode hold
```

Query Syntax

SENSe<cnum>:SWEep:MODE?

Return Type

Character

Default

CONTinuous

SENSe<cnum>:SWEep:POINts <num>
Applicable Models: All

(Read-Write) Sets the number of data points for the measurement.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Choose any number between 1 and the VNA maximum number of points.

This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```
SENS:SWE:POIN 51
sense2:sweep:points max
```

**Query Syntax**

SENSe<cnun>:SWEep:POINts?

**Return Type**

Numeric

**Default**

201

SENSe<cnun>:SWEep:SLOCal:MAXimum <num>

Applicable Models: N522xB, N523xB, N524xB, E5080, M9485A, M980xA, P50xA

(Read-Write) Sets the Shift LO maximum frequency for the selected channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Range of shift LO maximum frequency is 1.5E8 to Maximum frequency.

**Examples**

```
SENS:SWE:SCLOC:MAX 1.5E8
sense2:sweep:slocal:maximum 1.5E8
```

**Query Syntax**

SENSe<cnun>:SWEep:SLOCal:MAXimum?

**Return Type**

Numeric / Double precision floating point

**Default**

Maximum frequency

SENSe<cnun>:SWEep:SLOCal:STATe <bool>
Applicable Models: N522xB, N523xB, N524xB, E5080, M9485A, M980xA, P50xA

(Read-Write) Turns ON or OFF the Shift LO mode for the selected channel.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` Select shift LO mode from either of the following:
  - **ON or 1** - Turns ON the Shift LO mode.
  - **OFF or 0** - Turns OFF the Shift LO mode.

Examples

```plaintext
SENS:SWE:SLOC:STAT ON
sense2:sweep:slocal:state off
```

Query Syntax

```
SENSe<cnum>:SWEep:SLOCal:STATe?
```

Return Type

Boolean

Default

OFF or 0

---

SENSe<cnum>:SWEep:SRCPort <1 | 2> **Superseded**

Applicable Models: All but M9485A

This command is superseded. The `Calc:Par:Def:Ext` and `Calc:Par:Mod:Ext` can now optionally include the source port.

(Read-Write) Sets the source port when making non S-parameter measurements. Has no effect on S-parameter measurements.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<1 | 2>` 1 - Source power comes out Port 1
  - 2 - Source power comes out Port 2

Examples

```plaintext
SENS:SWE:SRCP 1
sense2:sweep:srcport 2
```

Query Syntax

```
SENSe<cnum>:SWEep:SRCPort?
```

Return Type

Character

Default

1

---

SENSe<cnum>:SWEep:SPEed <char>
(Read-Write) Sets and returns the state of Fast Sweep mode. Learn more about Fast Sweep.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Fast Sweep mode. Choose from:
  - FAST - turns Fast Sweep Mode ON
  - NORMal - turns Fast Sweep Mode OFF (Normal Mode).

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe&lt;cnum&gt;:SWEep:SPE NORM</td>
<td>Turns Fast Sweep Mode OFF (Normal Mode)</td>
</tr>
<tr>
<td>sense2:sweep:speed fast</td>
<td>Turns Fast Sweep Mode ON</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe<cnum>:SWEep:SPEed?

**Return Type** Character

**Default** NORMal

SENSe<cnum>:SWEep:STEP <num>

(Read-Write) Sets the frequency step size across the selected frequency range. This effectively sets the number of data points. Available ONLY when `Sweep Type` = Linear.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Frequency step size in Hz. Select any value up to the frequency range of the analyzer.

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe&lt;cnum&gt;:SWEep:STEP 1e6</td>
<td>Frequency step size in Hz</td>
</tr>
<tr>
<td>sense2:sweep:step 1000000</td>
<td>Frequency step size in Hz</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe<cnum>:SWEep:STEP?

**Return Type** Numeric

**Default** NA

SENSe<cnum>:SWEep:SRCPort:EXTended
This command is a helper command for `CALC:MEAS:DEF`. The user would set the srcport with this command and then request a non S-parameter measurement from `CALC:MEAS:DEF`, such as "A".

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Examples**

```
SENS:SWE:SRCP:EXT
sense2:sweep:srcport:extended
```

**Query Syntax**

`SENSe<cnum>:SWEep:SRCPort:EXTended?`

**Return Type**

- **Default**: Not Applicable

---

`SENSe<cnum>:SWEep:TIME:AUTO <ON | OFF>`

**Applicable Models**: All

(Read-Write) Turns the automatic sweep time function ON or OFF.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<ON | OFF>`
  - **ON** (or 1) - turns the automatic sweep time ON.
  - **OFF** (or 0) - turns the automatic sweep time OFF.

**Examples**

```
SENS:SWE:TIME:AUTO
sense2:sweep:time:auto off
```

**Query Syntax**

`SENSe<cnum>:SWEep:TIME:AUTO?`

**Return Type**

- **Default**: Boolean (1 = ON, 0 = OFF)

---

`SENSe<cnum>:SWEep:TIME:STARt?`
(Read-only) Returns the time the first point of a Time Sweep is measured.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Examples**

```plaintext
SENS:SWE:TIME:STAR?
sense2:sweep:time:start?
```

**Return Type** Numeric

**Default** Not Applicable

---

**SENSe<cnum>:SWEep:TIME[:STOP] <num>**

(Read-Write) Sets the time the analyzer takes to complete one sweep. If sweep time accuracy is critical, use ONLY the values that are attained using the up and down arrows next to the sweep time entry box. See Sweep Time.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Sweep time in seconds.

To select the fastest sweep speed, either send MIN as an argument to this command, or send SENS:SWE:TIME:AUTO 1.

This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

The MAX value will change based on point count, IFBW, and dwell time.

**Examples**

```plaintext
SENS:SWE:TIME 1ms
sense2:sweep:time .001
```

**Query Syntax**

SENSe<cnum>:SWEep:TIME[:STOP]?

**Return Type** Numeric

**Default** NA

---

**SENSe<cnum>:SWEep:TRIgger:DELay <num>**
**Applicable Models:** All

(Read-Write) Sets and reads the trigger delay for all measurements in the specified CHANNEL. This delay is only applied while TRIG:SOURce EXTernal and TRIG:SCOPe CURRent. After an external trigger is applied, the start of the sweep is delayed for the specified delay value plus any inherent latency.

To apply a trigger delay for all channels (Global), use TRIG:DEL

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Trigger delay value in seconds. Range is from 0 to 3 seconds

**Examples**

```plaintext
SENS:SWE:TRIG:DElay .003
sense2:sweep:trigger:delay 1
```

**Query Syntax**

SENSe<cnum>:SWEep:TRIGger:DElay?

**Return Type**

Numeric

**Default**

0

---

**SENSe<cnum>:SWEep:TRIGger:MODE <char>**

**Applicable Models:** All

(Read-Write) Sets and reads the trigger mode for the specified channel. This determines what EACH signal will trigger. [Learn more.]

**Note:** Setting Point and Sweep mode forces Trigger:SCOPe = CURRent

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Trigger mode. choose from:

  - **CHANnel** - Each trigger signal causes ALL traces in that channel to be swept.
  - **SWeep** - Each Manual or External trigger signal causes ALL traces that share a source port to be swept.
  - **POINt** - Each Manual or External trigger signal causes one data point to be measured.
  - **TRACe** - Allowed ONLY when SENS:SWE:GEN:POIN is enabled. Each trigger signal causes two identical measurements to be triggered separately - one trigger signal is required for each measurement. Other trigger mode settings cause two identical parameters to be measured simultaneously.
### Sense2:Sweep:Trigger:Mode Point

**Query Syntax**
```
SENSe<cnum>:SWEep:TRIGger:MODE?
```

**Return Type**
Character

**Default**
Channel

---

**SENSe<cnum>:SWEep:TRIGger:POINt <ON | OFF> Superseded**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

This command is replaced with `SENS:SWE:TRIG:MODE POINT`

(Read-Write) Specifies whether the specified channel will measure one point for each trigger or all of the measurements in the channel. Setting any channel to POINt mode will automatically set the `TRIGger:SCOPe = CURRent`.

**Parameters**
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<ON | OFF>`:
  - **ON** (or 1) - Channel measures one data point per trigger.
  - **OFF** (or 0) - All measurements in the channel made per trigger.

**Examples**
```
SENSe:SWEep:TRIG:POIN ON
sense2:sweep:trigger:point off
```

**Query Syntax**
```
SENSe<cnum>:SWEep:TRIGger:POINt?
```

**Return Type**
Boolean (1 = Point, 0 = Measurement)

**Default**
0 - Measurement

---

**SENSe<cnum>:SWEep:TYPE <char>**
Applicable Models: All

(Read-Write) Sets the type of analyzer sweep mode. First set sweep type, then set sweep parameters such as frequency or power settings.

Note: For SweptIMD channels, use SENS:IMD:SWE:TYPE SEGment

See Also: FCA Segment Sweep commands

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1</td>
</tr>
<tr>
<td>&lt;char&gt;</td>
<td>Choose from: LINear</td>
</tr>
</tbody>
</table>

Note: SWEEP TYPE cannot be set to SEGment if there are no segments turned ON. A segment is automatically turned ON when the analyzer is started.

Examples

SENS:SWE:TYPE LIN
sense2:sweep:type segment

Query Syntax
SENSe<cnum>:SWEep:TYPE?

Return Type Character
Default LINear

SENSe<cnum>:SWEep:TYPE:FACW <num>

Applicable Models: N522xB, N524xB, M937xA, P937xA

(Read-Write) Enables Fast CW sweep and sets the number of data points for the channel. Sweep Type must already be set to CW and FIFO must already be enabled.

See Also
FIFO commands
Example program
N5264B Measurement Receiver

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1</td>
</tr>
<tr>
<td>&lt;num&gt;</td>
<td>Number of data points to measure in Fast CW mode. This setting overwrites the standard number of points setting for the channel. The minimum value is 1.</td>
</tr>
</tbody>
</table>
The maximum value is $2^{32} - 1 = 2,147,483,647$. The "-1" indicates infinite point count (i.e., go forever). Any other value will produce invalid results.

If the data acquisition rate exceeds 400,000 points per second, the upper limit on the number of points is $1 \times 10^6$. The following are conditions that can cause the higher data rate:

- IFBW's $\geq 1$ MHz and internally triggered.
- fastCW sweeps that are externally triggered at a rate faster than 400,000 points per second.

Set to 0 to disable Fast CW.

### Examples

```
sens:swe:typ:facw 1e6
sense2:sweep:typ facw 1e3
```

### Query Syntax

SENSe<cnum>:Sweep:TYPE:FACW?

### Return Type

Numeric

### Default

0 - Disabled
Sense Switch

When you use the PXIe VNA, you can control the PXIe switch module through the VNA firmware. The following commands are available when the launcher includes the PXIe switch module.

```
SENSe:SWITch
 :M9161
  | :COUNT?
  | :MODule
   | :CHASsis
   | :CONTrol[:STATe]
   | :RESet:IMMediate
   | :SLOT
   | :SWITch:PATH
   | :SWITch:PATH:CATalog
 :M9155
  | :COUNT?
  | :MODule
   | :CHASsis
   | :CONTrol[:STATe]
   | RESet:IMMediate
   | :SLOT
   | :SWITch:PATH
   | :SWITch:PATH:CATalog
 :M9156
  | :COUNT?
  | :MODule
   | :CHASsis
   | :CONTrol[:STATe]
   | RESet:IMMediate
```
Click on a keyword to view the command details.

SENSe<cnun>:SWITch:M9161:COUNt?
**Applicable Models:** All PXIe VNAs

*(Read-only)* Returns the total number of M9161D switch modules that are connected to the VNA firmware.

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.</td>
</tr>
</tbody>
</table>

### Examples

```
SENS:SWIT:M9161:COUN?
sense:switch:m9161:count
```

### Return Type

Numeric

### Default

Not applicable

---

**SENSe<cnum>:SWITch:M9161:MODule<mod>:CHASsis?**

**Applicable Models:** All PXIe VNAs

*(Read Only)* Returns the chassis number where the specified M9161D module is located.

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.</td>
</tr>
<tr>
<td>&lt;mod&gt;</td>
<td>Module number of M9161D. The number starts from 1 for the left most module of M9161D.</td>
</tr>
</tbody>
</table>

### Examples

```
SENS:SWIT:M9161:MOD1:CHAS?
sense:SWITch:m9161:module2:chassis?
```

### Return Type

Numeric

### Default

Not applicable

---

**SENSe<cnum>:SWITch:M9161:MODule<mod>:CONTrol[:STATe] <bool>**
**Applicable Models:** All PXIe VNAs

*(Read-Write)* Sets and reads the status of M9161D control.

**Parameters**
- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<mod>`: Module number of M9161D. The number starts from 1 for the leftmost module of M9161D.
- `<bool>`: Module control state. Choose from:
  - **O** or **OFF** - Skips to control the M9161D at the specified channel.
  - **1** or **ON** - Enables to control the M9161D at the specified channel.

**Examples**
- `SENS:SWIT:M9161:MOD1:CONT ON`
- `sense2:SWITch:M9161:module2:control 0`

**Query Syntax**
- `SENS<cnum>:SWITch:M9161:MODule<mod>:CONTrol[:STATe]?`

**Return Type**
- Boolean

**Default**
- 1 or ON

---

**SENSe<cnum>:SWITch:M9161:MODule<mod>:RES:IMMediate**

**Applicable Models:** All PXIe VNAs

*(Write Only)* Resets the switches in the specified module to "All Open" state immediately.

**Parameters**
- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9161D. The number starts from 1 for the leftmost module of M9161D.

**Examples**
- `SENS:SWIT:M9161:MOD1:RES:IM`
- `sense:SWITch:M9161:module2:reset:immediate`

**Query Syntax**
- Not applicable

**Return Type**
- Not applicable

**Default**
- Not applicable
SENSe<cnum>:SWITch:M9161:MODule<mod>:SLOT?

Applicable Models: All PXIe VNAs

(Read Only) Reads the slot number where the specified M9161D is located.

Parameters
- <cnum>: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- <mod>: Module number of M9161D. The number starts from 1 for the leftmost module of M9161D.

Examples
SENSe:SWITch:M9161:MOD1:SLOT?
SENSe:SWITch:M9161:MOD2:SLOT?

Return Type: Numeric
Default: Not applicable

SENSe<cnum>:SWITch:M9161:MODule<mod>:SWITch:PATH <char>

Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the path for the M9161D switch.

Parameters
- <cnum>: Any existing channel number; if unspecified, value is set to 1.
- <mod>: Module number of M9161D. The number starts from 1 for the leftmost module of M9161D.
- <char>: Path. Choose from:
  - STATE1 to STATE4 - State 1 to 4
  - NSFSource - NF source switch (NF measurement only)
  - NFLO - NF LO switch (Option 720 only)
  - NFReceiver - NF receiver switch (NF measurement only)

Examples
SENSe:SWITch:M9161:MOD1:SWIT:PATH STAT1
SENSe:SWITch:M9161:MOD2:SWIT:PATH STATE4

Query Syntax
SENSe<cnum>:SWITch:M9161:MODule<mod>:SWITch:PATH?

Return Type: <char>
SENSe<cnum>:SWITch:M9161:MODule<mod>:SWITch:PATH:CATalog?

Applicable Models: All PXIe VNAs

(Read-Write) Lists the path of the M9161D for SENS:SWIT:M9161:MOD:SWIT:PATH.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.
<mod> Module number of M9161D. The number starts from 1 for the left most module of M9161D.

Examples

SENS:SWIT:M9161:MOD1:SWIT:PATH:CAT?
sense2:SWITch:M9161:module2:switch:path:catalog?

Query Syntax SENSSe<cnum>:SWITch:M9161:MODule<mod>:SWITch:PATH:CAT?

Return Type Comma-separated string

Default Not applicable

SENSe<cnum>:SWITch:M9155:COUNt?

Applicable Models: All PXIe VNAs

(Read-only) Returns the total number of M9155C/M9155CH40 switch modules that are connected to the VNA firmware.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

Examples

SENS:SWIT:M9155:COUN?
sense:switch:m9155:count

Return Type Numeric

Default

SENSe<cnum>:SWITch:M9155:MODule<mod>:CHASsis?
Applicable Models: All PXIe VNAs

(Read Only) Returns the chassis number where the specified M9155C/M9155CH40 module is located.

Parameters

<cnun> Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

<mod> Module number of M9155C/M9155CH40. The number starts from 1 for the leftmost module of M9155C/M9155CH40.

Examples

- SENSE:SWIT:M9155:MOD1:CHAS?
- sense:SWITch:m9155:module2:chassis?

Return Type Numeric

Default Not applicable

SENSe<cnun>:SWITch:M9155:MODule<mod>:CONTrol[:STATe] <bool>

Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the status of M9155C/M9155CH40 control.

Parameters

<cnun> Any existing channel number; if unspecified, value is set to 1.

<mod> Module number of M9155C/M9155CH40. The number starts from 1 for the leftmost module of M9155C/M9155CH40.

<bool> Module control state. Choose from:

0 or OFF - Skips to control the M9155C/M9155CH40 at the specified channel.

1 or ON - Enables to control the M9155C/M9155CH40 at the specified channel.

Examples

- SENS:SWIT:M9155:MOD1:CONT ON
- sense2:SWITch:m9155:module2:control 0

Query Syntax SENSe<cnun>:SWITch:M9155:MODule<mod>:CONTrol[:STATe]?

Return Type Boolean

Default 1 or ON
SENSe<cnum>:SWITch:M9155:MODule<mod>:RESet:IMMediate

**Applicable Models:** All PXIe VNAs

*(Write Only)* Resets the switches in the specified module to "All Open" state immediately.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9155C/M9155CH40 The number starts from 1 for the leftmost module of M9155C/M9155CH40.

**Examples**

```
SENSe:SWITch:M9155:MOD1:RESet:IMMediate
sense:SWITch:M9155:module2:RESet:IMMediate
```

**Query Syntax** Not applicable

**Return Type** Not applicable

**Default** Not applicable

SENSe<cnum>:SWITch:M9155:MODule<mod>:SLOT?

**Applicable Models:** All PXIe VNAs

*(Read Only)* Reads the slot number where the specified M9155C/M9155CH40 is located.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9155C/M9155CH40 The number starts from 1 for the leftmost module of M9155C/M9155CH40.

**Examples**

```
SENSe:SWITch:M9155:MOD1:SLOT?
sense:SWITch:M9155:module2:SLOT?
```

**Return Type** Numeric

**Default** Not applicable

SENSe<cnum>:SWITch:M9155:MODule<mod>:SWITch:PATH <char>
Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the path for the M9155C/M9155CH40 switch.

**Parameters**
- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<mod>` Module number of M9155C/M9155CH40. The number starts from 1 for the left most module of M9155C/M9155CH40.
- `<char>` Path. Choose from:
  - **STATe1** or **STATe2** - State 1 or 2
  - **NFSource** - NF source switch (NF measurement only)
  - **NFLO** - NF LO switch (Option 720 only)
  - **NFReceiver** - NF receiver switch (NF measurement only)

**Examples**
- `SENS:SWIT:M9155:MOD1:SWIT:PATH STAT1`
- `sense2:SWITCH:M9155:module2:switch:path state2`

**Query Syntax**
- `SENSe<cnum>:SWITch:M9155:MODule<mod>:SWITch:PATH?`

**Return Type** `<char>`

**Default** `STATe1`

---


**Applicable Models:** All PXIe VNAs

(Read-Write) Lists the path of the M9155C/M9155CH40 for `SENS:SWIT:M9155:MOD:SWIT:PATH`.

**Parameters**
- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<mod>` Module number of M9155C/M9155CH40. The number starts from 1 for the left most module of M9155C/M9155CH40.

**Examples**
- `sense2:SWITCH:M9155:module2:switch:path:catalog?`

**Query Syntax**
### SENSE<cnum>:SWITch:M9156:COUNt?

**Applicable Models:** All PXIe VNAs

*(Read-only)* Returns the total number of M9156C/M9156CH40 switch modules that are connected to the VNA firmware.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
</table>

**Return Type** Numeric

**Default** Not applicable

### SENSE<cnum>:SWITch:M9156:MODule<mod>:CHASsis?

**Applicable Models:** All PXIe VNAs

*(Read Only)* Returns the chassis number where the specified M9156C/M9156CH40 module is located.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

- `<mod>`: Module number of M9156C/M9156CH40. The number starts from 1 for the left most module of M9156C/M9156CH40.

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
</table>

**Return Type** Numeric

**Default** Not applicable
**SENSe<cnum>:SWITch:M9156:MODule<mod>:CONTroll [<STATe] <bool>**

**Applicable Models:** All PXIe VNAs

*(Read-Write)* Sets and reads the status of M9156C/M9156CH40 control.

**Parameters**
- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<mod>`: Module number of M9156C/M9156CH4. The number starts from 1 for the leftmost module of M9156C/M9156CH40.
- `<bool>`: Module control state. Choose from:
  - OFF or O - Skips to control the M9165C/M9156CH40 at the specified channel.
  - ON or 1 - Enables to control the M9156C/M9156CH40 at the specified channel.

**Examples**
- SENS:SWIT:M9156:MOD1:CONT ON
- sense2:SWITch:M9156:module2:control 0

**Query Syntax** SENS<cnum>:SWITch:M9156:MODule<mod>:CONTroll[:STATe]?

**Return Type** Boolean

**Default** 1 or ON

---

**SENSe<cnum>:SWITch:M9156:MODule<mod>:RESet:IMMediate**

**Applicable Models:** All PXIe VNAs

*(Write Only)* Resets the switches in the specified module to "All Open" state immediately.

**Parameters**
- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9156C/M9156CH40. The number starts from 1 for the leftmost module of M9156C/M9156CH40.

**Examples**
- SENS:SWIT:M9156:MOD1:RES:IMM
- sense:SWITch:M9156:module2:reset:immediate

**Query Syntax** Not applicable

**Return Type** Not applicable
SENSe<cnum>:SWITch:M9156:MODule<mod>:SLOT?

**Applicable Models:** All PXIe VNAs

(Read Only) Reads the slot number where the specified M9156C/M9155CH40 is located.

**Parameters**
- **<cnum>** Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- **<mod>** Module number of M9156C/M9156CH40. The number starts from 1 for the leftmost module of M9156C/M9156CH40.

**Examples**
- `SENSe:SWITch:M9156:MOD1:SLOT?`
- `SENSe:SWITch:M9156:module2:slot?`

**Return Type** Numeric

**Default** Not applicable

SENSe<cnum>:SWITch:M9156:MODule<mod>:SWITch:PATH <char>

**Applicable Models:** All PXIe VNAs

(Read-Write) Sets and reads the path for the M9156C/M9156CH40 switch.

**Parameters**
- **<cnum>** Any existing channel number; if unspecified, value is set to 1.
- **<mod>** Module number of M9156C/M9156CH40. The number starts from 1 for the left most module of M9156C/M9156CH40.
- **<char>** Path. Choose from:
  - **STATE1** or **STATE2** - State 1 or 2
  - **NFSource** - NF source switch (NF measurement only)
  - **NFLO** - NF LO switch (Option 720 only)
  - **NFRReceiver** - NF receiver switch (NF measurement only)
Examples

SENS:SWIT:M9156:MOD1:SWIT:PATH STAT1
sense2:SWITch:M9156:module2:switch:path state2

Query Syntax
SENS<cn>U>:SWITch:M9156:MODule<mod>:SWITch:PATH?

Return Type
<char>

Default
STATe1

SENS<cn>:SWITch:M9156:MODule<mod>:SWITch:PATH:CAalog?

Applicable Models: All PXIe VNAs

(Read-Write) Lists the path of the M9156C/M9156CH40 for SENS:SWIT:M9156:MOD:SWIT:PATH.

Parameters

<cn> Any existing channel number; if unspecified, value is set to 1.
<mod> Module number of M9156C/M9156CH40. The number starts from 1 for the left most module of M9156C/M9156CH40.

Examples
SENS:SWIT:M9156:MOD1:SWIT:PATH:CAT?
sense2:SWITch:M9156:module2:switch:path:catalog?

Query Syntax

Return Type
Comma-separated string

Default Not applicable

SENS<cn>:SWITch:M9157:COUNt?
Applicable Models: All PXIe VNAs

(Read-only) Returns the total number of M9157C/M9157CH40 switch modules that are connected to the VNA firmware.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

Examples

```
SENS:SWIT:M9157:COUN?
sense:switch:m9157:count
```

Return Type Numeric

Default Not applicable

SENSe<cnum>:SWITch:M9157:MODule<mod>:CHASsis?

Applicable Models: All PXIe VNAs

(Read Only) Returns the chassis number where the specified M9157C/M9157CH40 module is located.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

- `<mod>` Module number of M9157C/M9157CH40. The number starts from 1 for the left most module of M9157C/M9157CH40.

Examples

```
SENS:SWIT:M9157:MOD1:CHAS?
sense:SWITch:m9157:module2:chassis?
```

Return Type Numeric

Default Not applicable

SENSe<cnum>:SWITch:M9157:MODule<mod>:CONTrol[:STATe] <bool>
**Applicable Models:** All PXIe VNAs

*(Read-Write)* Sets and reads the status of M9157C/M9157CH40 control.

**Parameters**
- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<mod>` Module number of M9157C/M9157CH4. The number starts from 1 for the leftmost module of M9157C/M9157CH40.
- `<bool>` Module control state. Choose from:
  - **O** or **OFF** - Skips to control the M9167C/M9157CH40 at the specified channel.
  - **1** or **ON** - Enables to control the M9157C/M9157CH40 at the specified channel.

**Examples**

```
SENS:SWIT:M9157:MOD1:CONT ON
sense2:SWITch:M9157:module2:control 0
```

**Query Syntax**

```
SENSe<cnum>:SWITch:M9157:MODule<mod>:CONTrol[:STATe]? 
```

**Return Type**

Boolean

**Default**

1 or ON

---

**SENSe<cnum>:SWITch:M9157:MODule<mod>:RESet:IMMediate**

**Applicable Models:** All PXIe VNAs

*(Write Only)* Resets the switches in the specified module to "All Open" state immediately.

**Parameters**
- `<cnum>` Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>` Module number of M9157C/M9157CH40. The number starts from 1 for the leftmost module of M9157C/M9157CH40.

**Examples**

```
SENS:SWIT:M9157:MOD1:RES:IMM
sense:SWITch:M9157:module2:reset:immediate
```

**Query Syntax**

Not applicable

**Return Type**

Not applicable
SENSe<cnm>:SWITch:M9157:MODule<mod>:SLOT?

Applicable Models: All PXIe VNAs

(Read Only) Reads the slot number where the specified M9157C/M9157CH40 is located.

Parameters

<cnm> Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

<mod> Module number of M9157C/M9157CH40. The number starts from 1 for the leftmost module of M9157C/M9157CH40.

Examples

SENSe:SWITch:M9157:MOD1:SLOT?

sense:SWITch:M9157:mod2:slot?

Return Type Numeric

Default Not applicable

SENSe<cnm>:SWITch:M9157:MODule<mod>:SWITch:PATH <char>

Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the path for the M9157C/M9157CH40 switch.

Parameters

<cnm> Any existing channel number; if unspecified, value is set to 1.

<mod> Module number of M9157C/M9157CH40. The number starts from 1 for the leftmost module of M9157C/M9157CH40.

<char> Path. Choose from:

STATe1 to STATe6 - State 1 to 6

NFSource - NF source switch (NF measurement only)

NFLO - NF LO switch (Option 720 only)

NFReceiver - NF receiver switch (NF measurement only)
<table>
<thead>
<tr>
<th>Examples</th>
<th>SENS:SWIT:M9157:MOD1:SWIT:PATH STAT1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sense2:SWITch:M9157:module2:switch:path state2</td>
</tr>
</tbody>
</table>

**Query Syntax**  
SENSe<cnum>:SWITch:M9157:MODule<mod>:SWITch:PATH?

**Return Type**  
<char>

**Default**  
STATe1

---

**SENSe<cnum>:SWITch:M9157:MODule<mod>:SWITch:PATH:CAT?**

**Applicable Models:**  
All PXIe VNAs

*(Read-Write)* Lists the path of the M9157C/M9157CH40 for SENS:SWIT:M9157:MOD:SWIT:PATH.

**Parameters**

- `<cnum>`  
  Any existing channel number; if unspecified, value is set to 1.

- `<mod>`  
  Module number of M9157C/M9157CH40. The number starts from 1 for the left most module of M9157C/M9157CH40.

**Examples**

- SENS:SWIT:M9157:MOD1:SWIT:PATH:CAT?

- sense2:SWITch:M9157:module2:switch:path:catalog?

**Query Syntax**  
SENSe<cnum>:SWITch:M9157:MODule<mod>:SWITch:PATH:CAT?

**Return Type**  
Comma-separated string

**Default**  
Not applicable

---

**SENSe<cnum>:SWITch:M9164:COUNt?**

4690

4690
Applicable Models: All PXIe VNAs

(Read-only) Returns the total number of M9164x switch modules that are connected to the VNA firmware.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.


Return Type Numeric

Default Not applicable

SENSe<cnum>:SWITch:M9164:MODules<mod>:CHASsis?

Applicable Models: All PXIe VNAs

(Read Only) Returns the chassis number where the specified M9164x module is located.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

<mod> Module number of M9164x. The number starts from 1 for the left most module of M9164x.

Examples SENS:SWIT:M9164:MOD1:CHAS?
sense:SWITch:m9164:module2:chassis?

Return Type Numeric

Default Not applicable

SENSe<cnum>:SWITch:M9164:MODules<mod>:CONTrol[:STATE] <bool>
**Applicable Models:** All PXIe VNAs

*(Read-Write)* Sets and reads the status of M9164x control.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<mod>`: Module number of M9164x. The number starts from 1 for the left most module of M9164x.
- `<bool>`: Module control state. Choose from:
  - O or OFF - Skips to control the M9164x at the specified channel.
  - 1 or ON - Enables to control the M9164x at the specified channel.

**Examples**

```
SENS:SWIT:M9164:MOD1:CONT ON
sense2:SWITch:M9164:module2:control 0
```

**Query Syntax**

`SENSe<cnum>:SWITch:M9164:MODule<mod>:CONTrol[:STATe]?`

**Return Type**

Boolean

**Default**

1 or ON

---

**SENSe<cnum>:SWITch:M9164:MODule<mod>:RESet:IMMediate**

*Applicable Models:* All PXIe VNAs

*(Write Only)* Resets the switches in the specified module to "All Open" state immediately.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9164x. The number starts from 1 for the left most module of M9164x.

**Examples**

```
SENS:SWIT:M9164:MOD1:RES:IMM
sense:SWITch:M9164:module2:reset:immediate
```

**Query Syntax**

Not applicable

**Return Type**

Not applicable

**Default**

Not applicable
**SENSe\(<cnum>\):SWITch:M9164:MODules<mod>:SLOT?**

**Applicable Models:** All PXIe VNAs

*(Read Only)* Reads the slot number where the specified M9164x is located.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9164x. The number starts from 1 for the left most module of M9164x.

**Examples**

- `SENS:SWIT:M9164:MOD1:SLOT?`
- `sense:SWITch:M9164:module2:slot?`

**Return Type** Numeric

**Default** Not applicable

---

**SENSe\(<cnum>\):SWITch:M9164:MODules<mod>:SWITch<1-2>:PATH <char>**

**Applicable Models:** All PXIe VNAs

*(Read-Write)* Sets and reads the path for the M9164x switch.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<mod>`: Module number of M9164x. The number starts from 1 for the left most module of M9164x.
- `<char>`: Path. Choose from:
  - `STATe0` to `STATe16` - State 0 to 16

**Examples**

- `SENS:SWIT:M9164:MOD1:SWIT:PATH STAT1`
- `sense2:SWITch:M9164:module2:switch:path state4`

**Query Syntax** `SENSe\(<cnum>\):SWITch:M9164:MODule<mod>:SWITch:PATH?`

**Return Type** `<enum>`

**Default** `STATe1 or STATe2`

---

**SENSe\(<cnum>\):SWITch:M9164:MODules<mod>:MODE?**
Applicable Models: All PXIe VNAs

*(Read Only)* Returns the model number of M9164x.

**Parameters**

<cnun> Any existing channel number; if unspecified, value is set to 1.

<mod> Module number of M9164x. The number starts from 1 for the left most module of M9164x

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:SWIT:M9164:MOD1:MOD?</td>
</tr>
<tr>
<td>sense:SWITch:M9164:module2:model?</td>
</tr>
</tbody>
</table>

**Return Type** string

**Default** Not applicable

**SENSe<cnun>:SWITch:M9165:COUNt?**

Applicable Models: All PXIe VNAs

*(Read-only)* Returns the total number of M9165x switch modules that are connected to the VNA firmware.

**Parameters**

<cnun> Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:SWIT:M9165:COUN?</td>
</tr>
<tr>
<td>sense:switch:m9165:count</td>
</tr>
</tbody>
</table>

**Return Type** Numeric

**Default** Not applicable

**SENSe<cnun>:SWITch:M9165:MODules<mod>:CHASsis?**
Applicable Models: All PXIe VNAs

(Read Only) Returns the chassis number where the specified M9165x module is located.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of M9165x. The number starts from 1 for the left most module of M9165x.

**Examples**

```
SENSe:SWITch:M9165x:MOD1:CHAS?
sense:SWITch:m9165x:module2:chassis?
```

**Return Type** Numeric

**Default** Not applicable

---

**SENSe<cnm>:SWITCH:M9165x:MODules<mod>:CONTrol[:STATe] <bool>**

Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the status of M9165x control.

**Parameters**

- `<cnm>`: Any existing channel number; if unspecified, value is set to 1.
- `<mod>`: Module number of M9165x. The number starts from 1 for the left most module of M9165x.
- `<bool>`: Module control state. Choose from:
  - **O** or **OFF** - Skips to control the M9165x at the specified channel.
  - **1** or **ON** - Enables to control the M9165x at the specified channel.

**Examples**

```
SENSe:SWITch:M9165x:MOD1:CONT ON
sense2:SWITch:m9165x:module2:control 0
```

**Query Syntax**

```
SENSe<cnm>:SWITCH:M9165x:MODule<mod>:CONTrol[:STATe]?
```

**Return Type** Boolean

**Default** **1** or **ON**

---

**SENSe<cnm>:SWITCH:M9165x:MODule<mod>:RESet:IMMediate**
Applicable Models: All PXIe VNAs

(Write Only) Resets the switches in the specified module to "All Open" state immediately.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>` Module number of M9165x. The number starts from 1 for the left most module of M9165x.

Examples

- `SENS:SWIT:M9165:MOD1:RES:IMM`
- `sense:SWITch:M9165:module2:reset:immediate`

Query Syntax Not applicable

Return Type Not applicable

Default Not applicable

SENSe<cnum>:SWITch:M9165:MODules<mod>:SLOT?

Applicable Models: All PXIe VNAs

(Read Only) Reads the slot number where the specified M9165x is located.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>` Module number of M9165x. The number starts from 1 for the left most module of M9165x.

Examples

- `SENS:SWIT:M9165:MOD1:SLOT?`
- `sense:SWITch:M9165:module2:slot?`

Return Type Numeric

Default Not applicable

SENSe<cnum>:SWITch:M9165:MODules<mod>:SWITch<1-2>:PATH <char>
**Applicable Models:** All PXIe VNAs

**(Read-Write)** Sets and reads the path for the M9165x switch.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<mod>` Module number of M9165x. The number starts from 1 for the left most module of M9165x.
- `<char>` Path. Choose from:

  STATe0 to STATe16 - State 0 to 16

**Examples**

```
SENS:SWIT:M9165:MOD1:SWIT:PATH STAT1
sense2:SWITch:M9165:module2:switch:path state4
```

**Query Syntax**

`SENS<cnum>:SWITch:M9165:MODule<mod>:SWITch:PATH?`

**Return Type** `<enum>`

**Default** STATe1 or STATe2

---

**SENS<cnum>:SWITch:M9165:MODules<mod>:MODel?**

**Applicable Models:** All PXIe VNAs

**(Read Only)** Returns the model number of M9165x.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<mod>` Module number of M9165x. The number starts from 1 for the left most module of M9165x

**Examples**

```
SENS:SWIT:M9165:MOD1:MOD?
sense:SWITch:M9165:module2:model?
```

**Return Type** `string`

**Default** Not applicable

---

**SENS<cnum>:SWITch:P9164:COUNT?**
Applicable Models: All PXIe VNAs

(Read-only) Returns the total number of P9164x switch modules that are connected to the VNA firmware.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

Examples

```
SENS:SWIT:P9164:COUN?
sense:switch:p9164:count
```

Return Type Numeric

Default Not applicable

---

SENSe<i>:SWITch:P9164:MODules<i>:CHASsis?

Applicable Models: All PXIe VNAs

(Read Only) Returns the chassis number where the specified P9164x module is located.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>` Module number of P9164x. The number starts from 1 for the left most module of P9164x.

Examples

```
SENS:SWIT:P9164:MOD1:CHAS?
sense:SWITch:p9164:module2:chassis?
```

Return Type Numeric

Default Not applicable

---

SENSe<i>:SWITch:P9164:MODules<i>:CONTrol[:STATe] <bool>

4698
Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the status of P9164x control.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<mod>` Module number of P9164x. The number starts from 1 for the left most module of P9164x.
- `<bool>` Module control state. Choose from:
  - `OFF` or `OFF` - Skips to control the P9164x at the specified channel.
  - `ON` or `ON` - Enables to control the P9164x at the specified channel.

Examples

```
SENS:SWIT:P9164:MOD1:CONT ON
sense2:SWITch:p9164:module2:control 0
```

Query Syntax

```
SENSe<cnum>:SWITch:P9164:MODule<mod>:CONTrol[:STATe]?
```

Return Type

Boolean

Default

1 or ON

---

SENSe<cnum>:SWITch:P9164:MODules<mod>:SLOT?

Applicable Models: All PXIe VNAs

(Read Only) Reads the slot number where the specified P9164x is located.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>` Module number of P9164x. The number starts from 1 for the left most module of P9164x.

Examples

```
SENS:SWIT:P9164:MOD1:SLOT?
sense:SWITch:p9164:module2:slot?
```

Return Type

Numeric

Default

Not applicable

---

SENSe<cnum>:SWITch:P9164:MODules<mod>:SWITch<1-2>:PATH <char>
Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the path for the P9164x switch.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<mod>` Module number of P9164x. The number starts from 1 for the left most module of P9164x.
- `<char>` Path. Choose from:

  STATe0 to STATe16 - State 0 to 16

**Examples**

```plaintext
SENS:SWIT:P9164:MOD1:SWIT:PATH STAT1
sense2:SWITCH:p9164:module2:switch:path state4
```

**Query Syntax**

```
SENSe<cnum>:SWITch:P9164:MODule<mod>:SWITch:PATH?
```

**Return Type** `<enum>`

- **Default** STATe1 or STATe2

---

**SENSe<cnum>:SWITch:P9164:MODules<mod>:MODEL?**

Applicable Models: All PXIe VNAs

(Read Only) Returns the model number of P9164x.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<mod>` Module number of P9164x. The number starts from 1 for the left most module of P9164x.

**Examples**

```plaintext
SENS:SWIT:P9164:MOD1:MOD?
sense:SWITCH:p9164:module2:model?
```

**Return Type** string

- **Default** Not applicable

---

**SENSe<cnum>:SWITch:P9165:COUNT?**
Applicable Models: All PXIe VNAs

(Read-only) Returns the total number of P9165x switch modules that are connected to the VNA firmware.

Parameters

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

Examples

```
SENS:SWIT:P9165:COUN?
sense:switch:p9165:count
```

Return Type: Numeric

Default: Not applicable

---

SENSe<cn><num>:SWITch:P9165:MODules<mod>:CHASsis?

Applicable Models: All PXIe VNAs

(Read Only) Returns the chassis number where the specified P9165x module is located.

Parameters

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>`: Module number of P9165x. The number starts from 1 for the left most module of P9165x.

Examples

```
SENS:SWIT:P9165:MOD1:CHAS?
sense:SWITch:p9165:module2:chassis?
```

Return Type: Numeric

Default: Not applicable

---

SENSe<cn><num>:SWITch:P9165:MODules<mod>:CONTrol[:STATe] <bool>

4701
**Applicable Models:** All PXIe VNAs

*(Read-Write)* Sets and reads the status of P9165x control.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<mod>` Module number of P9165x. The number starts from 1 for the left most module of P9165x.
- `<bool>` Module control state. Choose from:
  - **O** or **OFF** - Skips to control the P9165x at the specified channel.
  - **1** or **ON** - Enables to control the P9165x at the specified channel.

**Examples**

```
SENS:SWIT:P9165:MOD1:CONT ON
sense2:SWITch:p9165:module2:control 0
```

**Query Syntax**

```
SENSe<cnum>:SWITch:P9165:MODule<mod>:CONTrol[:STATe]?
```

**Return Type** Boolean

**Default** 1 or ON

---

**SENSe<cnum>:SWITch:P9165:MODules<mod>:SLOT?**

**Applicable Models:** All PXIe VNAs

*(Read Only)* Reads the slot number where the specified P9165x is located.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<mod>` Module number of P9165x. The number starts from 1 for the left most module of P9165x.

**Examples**

```
SENS:SWIT:P9165:MOD1:SLOT?
sense:SWITch:p9165:module2:slot?
```

**Return Type** Numeric

**Default** Not applicable

---

**SENSe<cnum>:SWITch:P9165:MODules<mod>:SWITch<1-2>:PATH <char>**
Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the path for the P9165x switch.

Parameters

<i>cnum</i>  Any existing channel number; if unspecified, value is set to 1.
<i>mod</i>  Module number of P9165x. The number starts from 1 for the left most module of P9165x.
<i>char</i>  Path. Choose from:

- STATe0 to STATe16 - State 0 to 16

Examples

- SENS:SWIT:P9165:MOD1:SWIT:PATH STAT1
- sense2:SWITch:p9165:module2:switch:path state4

Query Syntax

SENS<enum>:SWITch:P9165:MODULE<mod>:SWITch:PATH?

Return Type

<i>enum</i>

Default

STATe1 or STATe2

---

SENS<enum>:SWITch:P9165:MODULEs<mod>:MODEL?

Applicable Models: All PXIe VNAs

(Read Only) Returns the model number of P9165x.

Parameters

<i>cnum</i>  Any existing channel number; if unspecified, value is set to 1.
<i>mod</i>  Module number of P9165x. The number starts from 1 for the left most module of P9165x.

Examples

- SENS:SWIT:P9165:MOD1:MODEL?
- sense:SWITch:p9165:module2:model?

Return Type  string

Default  Not applicable
SENSe:TDR Commands

These commands control bandwidth, DUT information, avoid spurious function, and sweep.

<table>
<thead>
<tr>
<th>SENSe:TDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>BWIDth</td>
</tr>
<tr>
<td>[:RESolution]</td>
</tr>
<tr>
<td>DLENght</td>
</tr>
<tr>
<td>AUTO</td>
</tr>
<tr>
<td>IMMEDIATE</td>
</tr>
<tr>
<td>DATA</td>
</tr>
<tr>
<td>SPURious</td>
</tr>
<tr>
<td>AVOID</td>
</tr>
<tr>
<td>IMMEDIATE</td>
</tr>
<tr>
<td>STATe?</td>
</tr>
<tr>
<td>INPut</td>
</tr>
<tr>
<td>DRATE</td>
</tr>
<tr>
<td>STATe?</td>
</tr>
<tr>
<td>SWEeep</td>
</tr>
<tr>
<td>AVERAGE</td>
</tr>
<tr>
<td>MODE</td>
</tr>
<tr>
<td>SINGLE</td>
</tr>
</tbody>
</table>

Click on a red keyword to view the command details.

See Also

- Synchronizing the Analyzer and Controller
- SCPI Command Tree
SENSe<cnum>:TDR:BWIDth[:RESolution] <value>

Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the IF bandwidth value.

Parameters
- <cnum> Channel number of the measurement. If unspecified, <cnum> is set to 1.
- <value> IF bandwidth value in Hz.

Examples

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:TDR:BWID:RES</td>
</tr>
<tr>
<td>sense:tdr:bwidth:resolution</td>
</tr>
</tbody>
</table>

Query Syntax  SENSSe<cnum>:TDR:BWIDth[:RESolution]?

Return Type  Double

Default  100 kHz

SENSe<cnum>:TDR:DLENgth:AUTO:IMMediate

Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) This command executes auto DUT length setting.

Parameters
- <cnum> Channel number of the measurement. If unspecified, <cnum> is set to 1.

Examples

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:TDR:DLEN:AUTO:IMM</td>
</tr>
<tr>
<td>sense:tdr:dlength:auto:immediate</td>
</tr>
</tbody>
</table>

Default  Not Applicable

SENSe<cnum>:TDR:DLENgth:DATA <value>

Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the DUT length value.

Parameters
- <cnum> Channel number of the measurement. If unspecified, <cnum> is set to 1.
- <value> DUT length value in seconds. The range is 6.26n to 416n

Examples

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:TDR:DLEN:DATA</td>
</tr>
<tr>
<td>sense:tdr:dlength:data</td>
</tr>
</tbody>
</table>

Query Syntax  SENSSe<cnum>:TDR:DLENgth:DATA?

Return Type  Double

Default  6.26n

SENSe<cnum>:TDR:SPURious:AVOid:IMMediate
Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) This command executes avoid spurious.

Parameters

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

Examples

```
SENS:TDR:SPUR:AVO:IMM
sense:tdr:spurious:avoid:immediate
```

Default Not Applicable

---

SENSe<cnum>:TDR:SPURious:AVOid:STATe?

Applicable Models: All with TDR Options (S9x011A/B)

(Read-only) This command queries the avoid spurious state. This command is used only with Hot TDR mode.

- This command is ON when :SENS:TDR:SPURious:AVOid:IMMediate command succeeds.
- This command is OFF when :SENS:TDR:SPURious:AVOid:IMMediate command fails to find a spur.

Parameters

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

Examples

```
SENS<TDR:SPURious:AVOid:STAT
sense:tdr:spurious:avoid:stat
```

Return Type

Boolean

1 - ON

0 - OFF

---

SENSe<cnum>:TDR:SPURious:INPut:DRATe <value>
Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the value of input bit rate for avoid spurious.

Parameters

- **<cnum>**  
  Channel number of the measurement. If unspecified, <cnum> is set to 1.

- **<value>**  
  Value of input bit rate in bits per second (bps). The range is 1.21E6 to 60.8E9.

Examples

```
SENS:TDR:SPUR:INP:DRAT 1.5E9
sense:tdr:spurious:input:drate 1.5e9
```

Query Syntax

```
SENSe<cnum>:TDR:SPURious:INPut:DRATe?
```

Return Type

Double

Default

1E9

SENSe<cnum>:TDR:SPURious:STATe?

Applicable Models: All with TDR Options (S9x011A/B)

(Read-only) This command queries the Hot TDR mode status.

- To turn ON Hot TDR mode, use  
  `SENSe:TDR:SPURious:AVOid:IMMediate`

- To turn OFF Hot TDR mode, use  
  `SYSTem:PRESet`

Parameters

- **<cnum>**  
  Channel number of the measurement. If unspecified, <cnum> is set to 1.

Examples

```
SENS:TDR:SPUR:STAT?
sense:tdr:spurious:state?
```

Return Type

Boolean

```
1 - ON
0 - OFF
```

SENSe<cnum>:TDR:SWEep:AVERage <bool>
Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the TDR averaging trigger state on/off. When averaging trigger is on, one trigger makes one averaging measurement. For example, if the averaging factor is set at 16, one trigger makes a measurement 16 times.

Parameters

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<bool>` ON or 1 - Turns averaging trigger on.
- OFF or 0 - Turns averaging trigger off.

Examples

- `SENS:TDR:SWE:AVER ON`
- `sense:tdr:sweep:average on`

Query Syntax

- SENSE<cnum>:TDR:SWEep:AVERage?

Return Type

- Boolean

Default

- OFF

SENSe<cnum>:TDR:SWEep:MODE <enum>

Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets trigger mode.

Parameters

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<enum>` Trigger mode. Choose from:
  - HOLD - Trigger is on hold until the conditions are met, then the trigger event starts.
  - SINGle - Trigger event is run once.
  - RUN - Trigger event runs continuously.

Examples

- `SENS:TDR:SWE:MODE RUN`
- `sense:tdr:sweep:mode run`

Query Syntax

- SENSE<cnum>:TDR:SWEep:MODE?

Return Type

- String

Default

- RUN

SENSe<cnum>:TDR:SWEep:SINGle
Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) This command executes single trigger.

Parameters

<enum> Channel number of the measurement. If unspecified, <enum> is set to 1.

Examples

SENS:TDR:SWE:SING
sense:tdr:sweep:single

Default Not Applicable
SENSe<cnum>:X[:VALues]? - Superseded

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Replaced with CALC:X?

(Read-only) Returns the stimulus values for the specified channel. If the channel is sweeping the source backwards, the values will be in descending order.

**Note:** To avoid frequency rounding errors, specify FORM:DATA <Real,64> or <ASCii, 0>

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.

**Examples**

<table>
<thead>
<tr>
<th></th>
<th>SENS:X?</th>
<th>sense2:x:values?</th>
</tr>
</thead>
</table>

**Return Type**

- Depends on FORM:DATA command

**Default**

- Not applicable
Service Commands

Controls and queries settings related with Service.

SERVice:
  :DUMMy:DUT:FILE
  :DUMMy:DUT:NOISe:STATe
  :LOGGing:CLEAR

SERVice:DUMMy:DUT:FILE <string>

Applicable Models: ALL

(Read-Write) Set/get trace sNp file for dummy DUT. Preset & Save/Recall does not work for this command

<table>
<thead>
<tr>
<th>Parameters</th>
<th>&lt;string&gt; sNp file to be set to dummy DUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>SERV:DUMM:DUT:FILE?</td>
</tr>
<tr>
<td></td>
<td>service:dummy:dut:file?</td>
</tr>
<tr>
<td>Return Type</td>
<td>string</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

SERVice:DUMMy:DUT:NOISe:STATe <bool>

Applicable Models: ALL

(Read-Write) Set/get trace noise state for dummy DUT. Preset & Save/Recall does not work for this command

<table>
<thead>
<tr>
<th>Parameters</th>
<th>&lt;bool&gt; Noise state.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>SERV:DUMM:DUT:NOIS:STAT?</td>
</tr>
<tr>
<td></td>
<td>service:dummy:dut:noise:state?</td>
</tr>
<tr>
<td>Return Type</td>
<td>bool</td>
</tr>
<tr>
<td>Default</td>
<td>ON</td>
</tr>
</tbody>
</table>

SERVice:LOGGing:CLEAR

Applicable Models: E5080, M9485A, M980xA, P50xA
(Write-only) Delete the service error log files stored in the modules and PC. See Error Log.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
</tr>
</thead>
</table>
| **Examples** | SERV:LOGG:CLE  
| | service:logging:clear |
| **Return Type** | N/A |
| **Default** | Not Applicable |
Source Commands

Controls the power delivered to the DUT and turn pulse on and off with an external source.

```
SOURce:
  CATalog?
  DC - More commands
  MODulation - More commands
  Multi-Dimensional Sweep - More commands

M9810
  COUNT
  MODule<mod>:ATTenuation[:VALue]
  MODule<mod>:ATTenuation:AUTO

PHASe - More commands
PORT:NUM?

POWer
  ALC:MODE
    CATalog?
    RECeiver - More commands
  ATTenuation
    AUTO
    RECeiver
      REFerence
      TEST
  CENTer
  CORRection - More commands
  COUPlle
  DETector
  [LEVel]
    [[IMMediate][AMPLitude]
    SLOPe
    STATe
  MODE
  PORT
    STARt
    STOP
  SPAN
  STARt
  STOP

PULSe
  MODulator
```
SOURce<cnum>:CATalog?

**Applicable Models:** All

*(Read-only)* Returns a list of valid port names that can be controlled. Some ports only have string names, NOT numbers. All commands that require a port argument have provisions for specifying either a port number OR a string name.

See also: Remotely Specifying a Source Port.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Examples**

- `SOUR:CAT?`  
  `source:catalog`  
  `'Some PNA-X models return  
  "Port 1,Port 2,Port 3,Port 4,Port 1 Src2,Source3"`

**Return Type**  
Comma-separated list of strings.

**Default**  
Not applicable
Applicable Models: M980xA, M981xAS

(Read-only) Returns the total number of M9810A vector modulator modules that are connected to the VNA firmware. Only one M9810A module can be installed, Hence, the return value could be 0 or 1.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1

Examples

```
SOUR:M9810:COUNT?
source:m9810:coun?
```

Return Type: Numeric

Default: Not Applicable

SOURce<cnum>:M9810:MODule<mod>:ATTenuation[:VALue] <num>

Applicable Models: M980xA, M981xAS

(Read-Write) Sets the M9810A attenuation level for the selected channel. This command disables the automatic attenuation control (SOURce:M9810:MODule:ATTenuation:AUTO). The parameter `<mod>` should be 1 because only one M9810A can be installed.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Attenuation level (dB). The range is from 0 to 60 dB with the step size 10 dB.

Examples

```
SOUR:M9810:MODULE1:ATTENUATION 10
source:m9810:mod1:att?
```

Return Type: Numeric

Default: 0

SOURce<cnum>:M9810:MODule<mod>:ATTenuation:AUTO <bool>
(Read-Write) Enable/disable the automatic M9810A attenuation control. When the attenuation value (using SOURce:M9810:MODule:ATTenuation) is set, this setting is turned off automatically. The parameter <mod> should be 1 because only one M9810A can be installed.

**Parameters**

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<bool>** ON (or 1) - Enable the automatic attenuator control.  
  OFF (or 0) - Disable the automatic attenuator control.

**Examples**

- SOUR:M9810:MODULE1:ATTENUATION:AUTO 1
- source:m9810:mod1:att:auto?

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** ON

---

SOURce<cnum>:PORT:NUM? <string>

**Applicable Models:** N522xB, N523xB, N524xB

(Read-only) Returns a port number for a named source.

All source ports have string names: "Port 1", "Port 2", etc. All external sources have customized names.

To convert a string name to a port number use this query.

**Parameters**

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<string>** String names.

**Examples**

- SOUR:PORT:NUM? "MVG"
- source:port:num "port 1"

**Return Type** String

**Default** Not Applicable

---

SOURce<cnum>:POWer<port>:ALC[:MODE] <char>, [src]
Applicable Models: N522xB, N523xB, N524xB, M9485A

(Read-Write) Sets and returns the ALC mode for the specified channel and port. Use SOUR:POW:ALC:MODE:CAT? to return a list of valid ALC modes for the VNA.

Learn more about ALC mode.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<char>` ALC Mode.

For the PNA-X choose from:

- **Internal** Standard ALC loop
- **OPENloop** No ALC loop

To set Leveling Mode to Receiver Leveling, use the Receiver Leveling commands.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW:ALC INT
source2:power2:alc:mode openloop
source:power:alc:mode openloop,"Port 1 Src2"
```

Query Syntax

SOURce<cnum>:POWer<port>:ALC:MODE? [src]

Return Type

Character

Default

INTernal
**Applicable Models:** N522xB, N523xB, N524xB, M9485A

(Read-only) Returns a list of valid ALC modes for the specified channel and port number. Use the returned values to set `SOUR:POW:ALC:MODE`.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

**Examples**

```
SOUR:POW:ALC:CAT?
source2:power2:alc:mode:catalog?
source:power:alc:mode:catalog? "INTernal,OPENloop,RxLeveling"
```

**Return Type** Comma-separated list of strings.

**Default** Not applicable

---

**SOURce<cnum>:POWer<port>:ATTenuation <num>, [src]**

**Applicable Models:** All

(Read-Write) Sets the attenuation level for the selected channel. Sending this command turns automatic attenuation control (`SOUR:POW:ATT:AUTO`) to OFF. If the ports are coupled, changing the attenuation on one port will also change the attenuation on all other ports. To turn port coupling OFF use `SOURce:POWer:COUPle OFF`.

**Note:** Attenuation cannot be set with **Sweep Type** set to **Power**

See `Sens:Power:ATT` to change receiver attenuation.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
<port> Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<num> Attenuation value. The range of settable values depends on the VNA model. To determine the valid settings, do one of the following:

- See VNA models and options to see the range and step size for each model / option.
- Perform a query using MAX, then MIN, as an argument. Example: SOURce:POWer:ATT? Max However, this will not tell you the attenuation step size.

If an invalid attenuation setting is entered, the VNA will select the next lower valid value. For example, if 19 is entered, then for an E8361A, 10 dB attenuation will be selected.

**Note**: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

SOUR:POW:ATT 10

source2:power2:attenuation maximum

source:power:att 20, "Port 1 Src2"

**Query Syntax** SOURce<cnum>:POWer<port>:ATTenuation? [min/max] [src] [min/max,src]

**Return Type** Numeric

**Default** 0

SOURce<cnum>:POWer<port>:ATTenuation:AUTO <bool>, [src]
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Turns automatic attenuation control ON or OFF. Setting an attenuation value (using \texttt{SOURce:POWe\textsubscript{r}:ATTenuation <num>}) sets AUTO OFF.

**Parameters**

- \texttt{<cnum>} Any existing channel number. If unspecified, value is set to 1.
- \texttt{<port>} Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- \texttt{<bool>} ON (or 1) - turns coupling ON. The analyzer automatically selects the appropriate attenuation level to meet the specified power level.
- \texttt{OFF} (or 0) - turns coupling OFF. Attenuation level must be set using \texttt{SOURce:POWe\textsubscript{r}:ATTenuation <num>}.
- [src] \texttt{String}. (NOT case sensitive). Source port. Optional. Use \texttt{SOUR:CAT?} to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the \texttt{<port>} argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
SOUR:POW2:ATT:Auto On
source2:power:attenuation:auto off
sour:pow:att:auto 1, "Port 1 Src2"
```

**Query Syntax** \texttt{SOURce<cnum>:POWer<port>:ATTenuation:RECeiver:REFerence <num>}

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** ON
Applicable Models: M9377A

(Read-Write) Sets the attenuation level for the specified reference attenuator/port.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Port number of the VNA. If unspecified, value is set to 1.
- `<num>` Attenuation value in dB. 0dB (Low) or 35dB (High)

If a number other than these is entered, the analyzer will select the next lower valid value. For example, if 19dB is entered, then 0dB attenuation will be selected.

Examples

```
SOUR:POW2:ATT:REC:REF 0
source:power:attenuation:receiver:reference 35
```

Query

Syntax

```
SOURce<cnum>:POWer<port>:ATTenuation:RECeiver:REFerence?
```

Return

Type Numeric. If querying for the standard port, the return value is 0

Default 35

---

SOURce<cnum>:POWer<port>:ATTenuation:RECeiver:TEST <num>

Applicable Models: N522xB, N523xB, N524xB, M9485A, M980xA and P50xxA with option S9x090A

(Read-Write) Sets the attenuation level for the specified test attenuator/port.

Note: On M980xA and P50xxA, this command works only for Spectrum Analyzer channels. For S-parameter channels, use the Receiver Gain dialog or `SENSe:SOURce:RECeiver:GAIN:*` commands.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<num>` Attenuation value in dB. 0dB (Low) or 35dB (High) / 0dB (Low) or 20dB (High) for M980xA and P50xxA.

If a number other than these is entered, the analyzer will select the next lower valid value. For example, if 19dB is entered, then 0dB attenuation will be selected.
Examples

SOUR:POW2:ATT:REC:TEST 0
source:power:attenuation:receiver:test 35

Query Syntax

SOURce<cnum>:POWer<port>:ATTenuation:RECeiver:TEST?

Return Type
Numeric. If querying for the standard port, the return value is 0

Default 35 for others / 0 for M980xA and P50xxA

SOURce<cnum>:POWer<port>:CENTer <num>

Applicable Models: All

(Read-Write) Sets the power sweep center power. Must also set:
SENS:SWE:TYPE POWer and SOURce:POWer:SPAN <num>.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <num> Center power. Actual achievable leveled power depends on frequency.
- <port> If provided, this argument is ignored by the VNA.

Examples

SOUR:POW:CENT -15
source2:power:center -7

Query Syntax

SOURce<cnum>:POWer:CENTer?

Return Type
Numeric

Default 0 dBm

SOURce<cnum>:POWer<port>:COUPle <ON | OFF>
Applicable Models: All

(Read-Write) Turns Port Power Coupling ON or OFF.

Parameters

- `<cnum>`
  - Any existing channel number. If unspecified, value is set to 1
- `<ON | OFF>`
  - **ON** (or 1) - turns coupling ON. The same power level is used for both source ports.
  - **OFF** (or 0) - turns coupling OFF. Power level can be set individually for each source port.

Examples

```
SOUR:POW:COUP ON
source2:power:couple off
```

Query Syntax

`SOURce<cnum>:POWer:COUPle?`

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON

---

SOURce<cnum>:POWer:DETructor <char> OBSOLETE

(Read-Write) The VNA models with external leveling are now OBSOLETE.

Sets the source leveling loop as Internal or External.

Parameters

- `<cnum>`
  - Any existing channel number. If unspecified, value is set to 1
- `<char>`
  - **INTernal** - Internal leveling is applied to the source
  - **EXTernal** - External leveling is applied to the source through a rear-panel connector. ONLY provided on 3 GHz, 6 GHz, and 9 GHz VNA models.

Examples

```
SOUR:POW:DET INT
source2:power:detector external
```

Query Syntax

`SOURce<cnum>:POWer:DETrctor?`

Return Type

Character

Default

INTernal

---

SOURce<cnum>:POWer<port>[:LEVel][:IMMediate][:AMPLitude] <num>, [src]
Applicable Models: All

(Read-Write) Sets the RF power output level.

Note: When the sweep type is "segment sweep", and "independent port power per segment" is set, then this command is not used.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.
- `<num>` Source power in dBm.

Note: The range of settable power values depends on the VNA model and if source attenuators are installed. To determine the range of values, perform a query using MAX, then MIN, as an argument. (SOUR:POW:ATT:AUTO must be set to ON) Example: SOURce:POWer? Max

Actual achievable leveled power depends on frequency.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

- SOUR:POW1 5
- source2:power:level:immediate:amplitude maximum
- sour:pow 5, "Port 1 Src2"

Query Syntax

SOURce<cnm>:POWer[:LEVel][:IMMediate][:AMPLitude]? [src]

Return Type

Numeric

Default

0 dBm

SOURce<cnm>:POWer[:LEVel]:SLOPe <num>
Applicable Models: All

(Read-Write) Sets the RF power slope value.

Also enable the slope state using `SOUR:POW:SLOP:STAT ON`.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Slope value in db/GHz. Choose any value between -2 and 2 (0 is no slope).

**Examples**

- `SOUR:POW:SLOP .5234434`
- `source2:power:level slope -1.345`

**Query Syntax**

`SOURce<cnum>:POWer[:LEVel]:SLOPe?`

**Return Type**

Numeric

**Default**

0

---

SOURce<cnum>:POWer<port>:MODE <state>, [src]

Applicable Models: All

(Read-Write) Turns Power Slope ON or OFF. Set the slope using `SOUR:POW:SLOP`.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<ON|OFF>`: ON (or 1) - turns slope ON.
- `OFF` (or 0) - turns slope OFF.

**Examples**

- `SOUR:POW:SLOP:STAT ON`
- `source2:power:slope:state off`

**Query Syntax**

`SOURce<cnum>:POWer[:LEVel]:SLOPe:STATe?`

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

---

4725
Applicable Models: All

(Read-Write) Sets the state of VNA source for the specified port.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.
- `<state>` Source state. Choose from:
  - **AUTO** Source power is turned ON when required for a measurement.
  - **ON** Source power is always ON regardless of the measurement.
  - **OFF** Source power is always OFF regardless of the measurement.
  - **NOCTL** Do not send OFF commands to the external sources. If an external source is in the OFF state, this option is used to stop sending OFF commands to the external source to increase sweep speed.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

**Examples**

- `SOUR:POW:MODE ON`
- `source2:power4:mode OFF`
- `sour:pow:mode on, "Port 1 Src2"

**Query Syntax**

`SOURce<cnum>:POWer<port>:MODE? [src]`

**Return Type**

Character

**Default**

Auto

`SOURce<cnum>:POWer<port>:PORT:STARt <num>, [src]`
Applicable Models: All

(Read-Write) Sets and reads the power sweep start power value for a specific port. This allows uncoupled forward and reverse power sweep ranges. Must also set SENS:SWE:TYPE POWer and SOUR:POW:COUPle OFF.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, <port> is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<num>` Start power in dBm.

**Note:** The range of settable power values depends on the VNA model and if source attenuators are installed. To determine the range of values, perform a query using MAX, then MIN, as an argument. (SOUR:POW:ATT:AUTO must be set to ON) Example: SOURce:POWer:STARt? MIN

Actual achievable leveled power depends on frequency.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
SOUR:POW1:PORT:STAR -15
```
```
source2:power:port:start 5, "bal port 1"
```

**Query Syntax** SOURce<cnum>:POWer<port>:PORT:STARt? [src]

**Return Type** Numeric

**Default** -10 dBm

SOURce<cnum>:POWer<port>:PORT:STOP <num>, [src]
Applicable Models: All

(Read-Write) Sets and reads the power sweep stop power value for a specific port. This allows uncoupled forward and reverse power sweep ranges. Must also set SENS:SWE:TYPE POWer and SOUR:POW:COUPle OFF.

**Note:** When the sweep type is "segment sweep", this command is not used.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<num>`: Stop power in dBm.

**Note:** The range of settable power values depends on the VNA model and if source attenuators are installed. To determine the range of values, perform a query using MAX, then MIN, as an argument. (SOUR:POW:ATT:AUTO must be set to ON) Example: SOURce:POWer:STARt? MIN

Actual achievable leveled power depends on frequency.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

SOUR:POW1:PORT:STOP -15
source2:power:port:stop 5, "bal port 1"

**Query Syntax**

SOURce<cnum>:POWer<port>:PORT:STOP? [src]

**Return Type**

- **Numeric**

  **Default**

  0 dBm

SOURce<cnum>:POWer<port>:SPAN <num>
Applicable Models: All

(Read-Write) Sets the power sweep span power. Must also set:

SENS:SWE:TYPE POWer and SOURce:POWer:CENTer <num>.

Parameters

- <cnum>: Any existing channel number. If unspecified, value is set to 1.
- <num>: Span power. Actual achievable leveled power depends on frequency.
- <port>: If provided, this argument is ignored by the VNA.

Examples

SOUR:POW:SPAN -15
source2:power:span -7

Query Syntax

SOURce<cnum>:POWer:SPAN?

Return Type

Numeric

Default

0 dBm

SOURce<cnum>:POWer<port>:STARt <num>

Applicable Models: All

(Read-Write) Sets the power sweep start power for ALL ports being used by the specified channel. Must also set:

SENS:SWE:TYPE POWer and SOURce:POWer:STOP <num>.

Note: This command is not recommended, however this command is supported for backwards compatibility. This command will limit the possible power setting to the available range on Port 1, even if other ports have a wider power range. It is recommended to use the SOURce:POWer:PORT:STARt command instead.

To set start power for a specific port, use SOUR:POW:PORT:STARt.

Note: When the sweep type is "segment sweep", this command is not used.

Parameters

- <cnum>: Any existing channel number. If unspecified, value is set to 1
- <num>: Start power.

Note: The range of settable power values depends on the VNA model and if source attenuators are installed. To determine the range of values, perform a
query using MAX, then MIN, as an argument. (SOUR:POW:ATT:AUTO must be set to ON) Example: SOURce:POWer:STARt? MIN

Actual achievable leveled power depends on frequency.

<port>  If provided, this argument is ignored by the VNA.

Examples  SOUR:POW:STAR -15
          source2:power:start -7

Query Syntax  SOURce<cnum>:POWer:STARt?

Return Type  Numeric

Default  0 dBm

SOURce<cnum>:POWer<port>:STOP <num>

Applicable Models: All

(Read-Write) Sets the power sweep stop power for ALL ports being used by the specified channel. Must also set: SENS:SWE:TYPE POWer and SOURce:POWer:START <num>.

To set start power for a specific port, use SOUR:POW:PORT:STOP.

Note: When the sweep type is "segment sweep", this command is not used.

Parameters

<cn>  Any existing channel number. If unspecified, value is set to 1
<num>  Stop power.

Note: The range of settable power values depends on the VNA model and if source attenuators are installed. To determine the range of values, perform a query using MAX, then MIN, as an argument. (SOUR:POW:ATT:AUTO must be set to ON) Example: SOURce:POWer:STOP? MAX

Actual achievable leveled power depends on frequency.

<port>  If provided, this argument is ignored by the VNA.

Examples  SOUR:POW:STOP -15
          source2:power:stop -7

Query Syntax  SOURce<cnum>:POWer:STOP?

Return Type  Numeric

Default  0 dBm
SOURce<cnum>:PULSe<port>:MODulator[:STATe] <ON | OFF>[src]

Applicable Models: All

(Read-Write) Turns pulse on and off with an external modulation source.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<ON|OFF> ON (or 1) - turns pulse ON.
OFF (or 0) - turns pulse OFF.

<port> Source port number of the VNA. If unspecified, <port> is set to 1.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

SOUR:PULS:MOD:STAT ON, "MyMxg"
source2:pulse1:modulator:state off

Query Syntax SOURce<cnum>:PULSe<port>:MODulator[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

SOURce<cnum>:PULSe:MODulator:EXISts? [src]
Applicable Models: All

(Read-only) Checks if pulse source exists.

Parameters

Examples

看了一眼: SOUR:PULS:MOD:EXIS?
source:pulse:modulator:exists? "MyMxg"


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Return Type

String

Default

Not applicable
Source Modulation Commands

These commands control an external source to set up and calibrate I/Q modulation.

```
SOURce:
  MODulation
    | ARB
    | CLOCk:SRATe
    | DATA:I
     | DATA:Q
    | AUTO
    | ACPR
      | GBANd
      | [:STATe]
    | IMMEDIATE
    | NPR
      | GBANd
      | [:STATe]
    | SA
      | [:STATe]
    | CORRection
    | COLLection
    | ACP
      | ENABle
    | ITERations
    | LOWer
      | ENABle
      | GBANd
      | ITERations
      | RECEiver
      | SPAN
      | TOLerance
      | RCeiver
      | SPAN
      | TOLerance
```
| UPPer
| ENABle
| GBANd
| ITERations
| RECeiver
| SPAN
| TOLerance
| ACQuire
| STATus?
| APPend
| DISTortion
| ENABle
| ITERations
| RECeiver
| SPAN
| TOLerance
| EQUalization
| ENABle
| ITERations
| RECeiver
| SPAN
| TOLerance
| FAST
| ENABle
| FLATness
| ENABle
| ITERations
| RECeiver
| SPAN
| TOLerance
| FREQuency
| [:FIXed]
| POINts
| STARt
| STOP
| TYPE

4734


| LO
| FTHRu
| ENABle
| ITERations
| RECEiver
| SPAN
| TOLerance
| NOTch
| ENABle
| ITERations
| RECEiver
| SPAN
| TOLerance
| POWER
| ENABle
| [:FIXed]
| ITERations
| POINts
| RECEiver
| SPAN
| STARt
| STOP
| TOLerance
| TYPE
| [:STATe]
| FILE?
| CORREction
| CATalog?
| DELe te
| FREQuency?
| POWer?
| INITialize
| LOAD
| SAVE
| SIGNAL
| CARRier
| FREQuency |
| LIMit |
| DDIGits |
| ENABle |
| TOLerance |
| HREJect |
| MAX |
| TONE |
| SPACing |
| MIN |
| TONE |
| NUMBer |
| WAVeform |
| PERiod |
| NYQReject |
| ENABle |
| TYPE |
| PAVG |
| CALCulated? |
| PHASe |
| FIXed |
| RANDom |
| SEED |
| TYPE |
| SPAN |
| CALCulated? |
| PRIority |
| SRATe |
| AUTO |
| CALCulated? |
| TONE |
| NUMBer |
| CALCulated? |
| PRIority |
| SPACing |
| CALCulated? |
| PRIority |
| TONE |
| ALL |
| [:STATe] |
| COUNT? |
| FREQuency? |
| LOAD |
| PHASe |
| POWer |
| SAVE |
| [:STATe] |
| TYPE |
| LOAD |
| SAVE |
| [STATe] |

Click on a keyword to view the command details.
See Also

- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree
- Remotely Specifying a Source Port

SOURce<cnum>:MODulation<port>:ARB:CLOCk:SRATe <value> [,srcPort]

Applicable Models: N522xB, N523xB, N524xB, All with S9x070A/B

(Read-Write) Sets and reads the sample clock rate of the arbitrary waveform generator.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1. To make settings for ports that are not simple numbers, use the [srcPort] argument.
- `<value>` Clock frequency in units of Hertz (Hz-MHz).

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

SOUR:MOD1:ARB:LOC:SRAT 1E8
source2:modulation:arb:clock:srate 1E8

Query Syntax

SOURce<cnum>:MODulation<port>:ARB:CLOCk:SRATe? [srcPort]

Return Type

Numeric

Default

0

SOURce<cnum>:MODulation<port>:ARB:DATA:I <propName> [,srcPort]
Applicable Models: N522xB, N523xB, N524xB, All with S9x070A/B

(Read-Write) Sets and reads the I data for I/Q modulation.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1. To make settings for ports that are not simple numbers, use the `[srcPort]` argument.
- `<propName>` Array for I data

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Query Syntax
SOURce<cnum>:MODulation<port>:ARB:DATA:I? [srcPort]

Return Type Array
Default 0

---

SOURce<cnum>:MODulation<port>:ARB:DATA:Q <propName> [,.srcPort]

Applicable Models: N522xB, N523xB, N524xB, All with S9x070A/B

(Read-Write) Sets and reads the Q data for I/Q modulation.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1. To make settings for ports that are not simple numbers, use the `[srcPort]` argument.
- `<propName>` Array for Q data

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both
Arguments are specified, [srcPort] takes priority.

**Query Syntax**  
`SOURce<cnum>:MODulation<port>:DARB:DATA:Q? [srcPort]`

**Return Type**  
Array

**Default**  
0

---

**SOURce<cnum>:MODulation<port>:AUTO:ACPR:GBANd <num>[,srcPort]**

**Applicable Models:** All with S93070xB or S9x070A/B

*(Read-Write)* Sets and reads the guard band between the end of the modulation signal and where ACPR is measured (SA channel only).

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Guard band for ACP measurement.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:AUTO:ACPR:GBAN 0 Hz
source2:modulation:auto:acpr:gband 0 Hz
```

**Query Syntax**  
`SOURce<cnum>:MODulation<port>:AUTO:ACPR:GBANd?`

**Return Type**  
Numeric

**Default**  
0 Hz

---

**SOURce<cnum>:MODulation<port>:AUTO:ACPR[:STATe] <bool>[,srcPort]**
Applicable Models: All with S93070xB or S9x070A/B

(Read-Write) Enable or disable Autoset ACPR markers (SA channel only).

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable ACPR markers.
  - **1 - ON** - Enable ACPR markers.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:AUTO:ACPR:STAT 1
source:modulation:auto:acpr:state on
```

**Query Syntax** SOURce<cnum>:MODulation<port>:AUTO:ACPR [:STATe]?

**Return Type** Boolean

**Default** OFF

```
SOURce<cnum>:MODulation<port>:AUTO:IMMediate [,srcPort]
```
Applicable Models: All with S93070xB or S9x070A/B

(Write-only) Adjusts frequencies and markers immediately if modulation settings are changed (SA channel only). This command depends on the status of the SA, NPR, and ACPR autoset booleans.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

SOUR1:MOD1:AUTO:IMM

Query Syntax

Not applicable

Default

Not applicable

SOURce<cnum>:MODulation<port>:AUTO:NPR:GBANd <num>[,srcPort]

Applicable Models: All with S93070xB or S9x070A/B

(Read-Write) Sets and reads the guard band on each side of the notch in an NPR measurement (SA channel only).

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Guard band for NPR measurement.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.
Examples

SOUR:MOD1:AUTO:NPR:GBAN 0 Hz
source2:modulation:auto:npr:gband 0 Hz

Query Syntax

SOURce<cnum>:MODulation<port>:AUTO:NPR[:STATe] <bool> [.srcPort]

Return Type

Numeric

Default

0 Hz

SOURce<cnum>:MODulation<port>:AUTO:NPR[:STATe] <bool> [.srcPort]

Applicable Models: All with S93070xB or S9x070A/B

(Read-Write) Enable or disable Autoset NPR markers (SA channel only).

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1.

<port> Source port number of the VNA. If unspecified, <port> is set to 1.

<bool> Choose from:

0 - OFF - Disable NPR markers.

1 - ON - Enable NPR markers.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

SOUR:MOD1:AUTO:NPR:STAT 1
source:modulation:auto:npr:state on

Query Syntax

SOURce<cnum>:MODulation<port>:AUTO:NPR [:STATe]?

Return Type

Boolean

Default

ON

SOURce<cnum>:MODulation<port>:AUTO:SA[:STATe] <bool>[.srcPort]
Applicable Models: All with S93070xB or S9x070A/B

(Read-Write) Enable or disable automatic updating of SA sweep settings and coherence settings if the
modulation settings are changed (SA channel only).

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable automatic updating.
  - **1 - ON** - Enable automatic updating.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

```
SOUR:MOD1:AUTO:SA:STAT 1
source:modulation:auto:sa:state on
```

Query Syntax

SOURce<cnum>:MODulation<port>:AUTO:SA [:STATe]?

Return Type

Boolean

Default

ON
**Applicable Models:** All with S93070xB or S9x070A/B

*(Read-Write)* Set and read the ACP modulation calibration state.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  
  0 - **OFF** - Disable ACP calibration.
  1 - **ON** - Enable ACP calibration.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:ACP:ENAB 1
source2:modulation:correction:collection:acp:enable on
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:ENABle ?

**Return Type** Boolean

**Default** OFF

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:ITERations <num>[,srcPort]
```
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the maximum number of iterations used by the calibration routine. The calibration routine uses successive approximation.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Maximum number of iterations.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MOD1:CORR:COLL:ACP:ITER 2
source2:modulation:correction:collection:acp:iterations 2
```

Query Syntax

SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:ITERations?

Return Type

Numeric

Default

2

SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:LOWer:ENAMEle <bool>[,srcPort]
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B, S9x090A/B

**Read-Write** Enables or disables the lower ACP (ACPLo) modulation calibration.

![ACPLo Signal ACPUp](image)

**Parameters**

- `<ch>`  Any existing channel number. If unspecified, value is set to 1.
- `<port>`  Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>`  Choose from:
  
  **0 - OFF** - Disable lower ACP calibration.
  
  **1 - ON** - Enable lower ACP calibration.


  While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:ACP:LOW:ENAB 1
source2:modulation:correction:collection:acp:lower:enable on
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:LOWer:ENABle ?

**Return Type**  Boolean

**Default**  OFF

SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:LOWer:GBANd <num>[,srcPort]
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B, S9x090A/B

*(Read-Write)* Sets and reads the ACP lower frequency delta from the edge of the carrier to the beginning of the calibration span.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Guard band for lower ACP.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:ACP:LOW:GBAN 0 Hz
source2:modulation:correction:collection:acp:lower:gband 0 Hz
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:LOWer:GBANd?

**Return Type** Numeric

**Default** 0 Hz
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the maximum number of iterations used by the calibration routine. The calibration routine uses successive approximation.

### Parameters

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<port>** Source port number of the VNA. If unspecified, <port> is set to 1.
- **<num>** Maximum number of iterations.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

### Examples

```
source2:modulation:correction:collection:acp:lower:iterations 2
```

### Query Syntax

`SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:LOWer:ITERations?`

### Return Type

Numeric

### Default

2

`SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:LOWer:RECeiver <rcvr>[,srcPort]`
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B, S9x090A/B

*(Read-Write)* Sets and reads the receiver for the lower ACP modulation calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<rcvr>` **String**. Calibration plane.
  
  For options S9x09xxA/B, S9x090A/B choose from:
  
  A, B, C, D, R1, R2, R3, R4, a1, a2, a3, a4, b1, b2, b3, b4.

  For options S93070xB, choose from:
  
  DUTIn1, DUTOut2, DUTOut3, DUTOut4, DUTOut5.


  While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:ACP:LOW:REC "R1"
source2:modulation:correction:collection:acp:lower:receiver 'R1'
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:LOWer:RECeiver?

**Return Type** String

**Default** a1 (options S9x09xxA/B, S9x090A/B)

DUTIn1 (options S93070xB)
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B, S9x090A/B

*(Read-Write)* Sets and reads the calibration span for a lower ACP modulation calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Calibration span for lower ACP.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:ACP:LOW:SPAN 100 MHz
source2:modulation:correction:collection:acp:lower:span 100 MHz
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:LOWer:SPAN?
```

**Return Type** Numeric

**Default** Signal Span

---

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B, S9x090A/B

*(Read-Write)* Sets and reads the desired ACP calibration tolerance for the lower ACP modulation calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Calibration tolerance in dBc.
While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

```
```

Query Syntax

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:LOWer:TOLerance?
```

Return Type

Numeric

Default

-40

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:RECeiver
<rcvr>[,srcPort]
```

Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the receiver for the ACP modulation calibration.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, <port> is set to 1.
- `<rcvr>`: **String**. Calibration plane.
  - For options S9x09xxA/B, S9x090A/B choose from: A, B, C, D, R1, R2, R3, R4, a1, a2, a3, a4, b1, b2, b3, b4.
  - For options S93070xB, choose from: DUTIn1, DUTOut2, DUTOut3, DUTOut4, DUTOut5.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.
Examples

```plaintext
SOUR:MOD1:CORR:COLL:ACP:REC "R1"
source2:modulation:correction:collection:acp:receiver "R1"
```

Query Syntax

`SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:RECeiver?`

Return Type

String

Default

a1 (options S9x09xxA/B, S9x090A/B)

DUTIn1 (options S93070xB)

---

**SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:SPAN <num>[,srcPort]**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the calibration span for an ACP modulation calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, <port> is set to 1.
- `<num>` Calibration span.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```plaintext
SOUR:MOD1:CORR:COLL:ACP:SPAN 100 MHz
source2:modulation:correction:collection:acp:span 100 MHz
```

**Query Syntax**

`SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:SPAN?`

**Return Type**

Numeric

**Default**

Signal Span

---

**SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:TOLerance <num>[,srcPort]**
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the desired ACP calibration tolerance for the ACP modulation calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Calibration tolerance in dBC.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:ACP:TOL -40dBc
source2: modulation: correction: collection: acp: tolerance -40dBc
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:TOLerance?

**Return Type** Numeric

**Default** -40
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Enables or disables the upper ACP (ACPUp) modulation calibration.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - **0** - **OFF** - Disable upper ACP calibration.
  - **1** - **ON** - Enable upper ACP calibration.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:ACP:UPP:ENAB 1
source2:modulation:correction:collection:acp:upper:enable on
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:UPPer:ENABle ?

**Return Type**

Boolean

**Default**

OFF
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the ACP upper frequency delta from the edge of the carrier to the beginning of the calibration span.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Guard band for upper ACP.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MOD1:CORR:COLL:ACP:UPP:GBAN 0 Hz
source2:modulation:correction:collection:acp:upper:gband 0 Hz
```

Query Syntax

SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:UPPer:GBANd?

Return Type

Numeric

Default

0 Hz

SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:UPPPer:ITERations <num>[,srcPort]
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the maximum number of iterations used by the calibration routine. The calibration routine uses successive approximation.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Maximum number of iterations.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:UPPer:ITERations?

**Return Type**

Numeric

**Default**

2

**Return Type**

Numeric

**Default**

2
**Applicable Models:** All with Option S93070x/B/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the receiver for the upper ACP modulation calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<rcvr>` String. Calibration plane.

For options S9x09xxA/B, S9x090A/B choose from:

A, B, C, D, R1, R2, R3, R4, a1, a2, a3, a4, b1, b2, b3, b4.

For options S93070xB, choose from:

DUTIn1, DUTOOut2, DUTOOut3, DUTOOut4, DUTOOut5.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:ACP:UPP:REC "R1"
source2:modulation:correction:collection:acp:upper:receiver "R1"
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:UPPer:RECeiver?
```

**Return Type** String

**Default** a1 (options S9x09xxA/B, S9x090A/B)

DUTIn1 (options S93070xB)
(Read-Write) Sets and reads the calibration span for an upper ACP modulation calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Calibration span for upper ACP.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:ACP:UPP:SPAN 100 MHz
source2:modulation:correction:collection:acp:upper:span 100 Mhz
```

**Query Syntax**

SOURce`<cnum>`:MODulation`<port>`:CORRection:COLLection:ACP:UPPer:SPAN?

**Return Type** Numeric

**Default** Signal Span

(Read-Write) Sets and reads the desired ACP calibration tolerance for the upper ACP modulation calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Calibration tolerance in dBc.
While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACP:UPPer:TOLerance?
```

**Return Type**

Numeric

**Default**

-40

---

**SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACQuire <enum>[,srcPort]**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Write-only)* Sets the collection acquire.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - **SYNChronous**
  - **ASYNchronous**


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR1:MOD1:CORR:COLL:ACQ SYNC
```

**Query Syntax**

Not applicable

**Default**

SYNChronous
**SOURce<cnum>:MODulation<port>:CORRection:COLLection:ACQuire:STATus? [,srcPort]**

**Applicable Models:** N522xB, N523xB, N524xB, All with S9x070A/B

*(Read-only)* Returns a message indicating if the calibration was successful or not.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `[srcPort]` **String** *(NOT case sensitive)*. Source port. Optional. Use SOUR:CAT? to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD:CORR:COL:ACQ:STAT?
"Calibration succeeded."
```

**Return Type** Comma-separated list of strings.

**Default** Not applicable

---

**SOURce<cnum>:MODulation<port>:CORRection:COLLection:APPend <bool>[,srcPort]**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Enable appending source modulation calibration data to the stored *.mdx file. If disabled, old calibration data in the *.mdx file will be overwritten.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable appending calibration data. Old calibration data in the *.mdx file will be overwritten.
  - **1 - ON** - Enable appending calibration data.
- `[srcPort]` **String** *(NOT case sensitive)*. Source port. Optional. Use SOUR:CAT? to return a list of valid port names.
While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD6:LOAD 'D:\ModulationFile\npr.csv'
...
SOUR:MOD6:CORR:COLL:APP 1
SOUR:MOD6:CORR:COLL:ACQ
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:APPend ?

**Return Type**

Boolean

**Default**

OFF

SOURce<cnum>:MODulation<port>:CORRection:COLLection:DISTortion:ENABle <bool>[,srcPort]

**Applicable Models:** N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Enable or disable the distortion calibration state. The distortion calibration minimizes the vector error of the modulation signal over the Cal Span.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, <port> is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable distortion calibration.
  - **1 - ON** - Enable distortion calibration.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.
### SOUR:MOD1:CORR:COLL:DIST:ENAB 1

**Query Syntax**
```
SOUR:MODulation<port>:CORRection:COLLection:DISTortion:ENABle 1
```

**Return Type**
Boolean

**Default**
OFF

---

### SOURce<cnum>:MODulation<port>:CORRection:COLLection:DISTortion:ITERations

#### Applicable Models:
N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the maximum number of distortion correction iterations used by the calibration routine. The calibration routine uses successive approximation.

**Parameters**
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, <port> is set to 1.
- `<num>`: Maximum number of distortion correction iterations.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**
```
SOUR:MOD1:CORR:COLL:DIST:ITER 3
source2:modulation:correction:collection:distortion:iterations 3
```

**Query Syntax**
```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:DISTortion:ITERations?
```

**Return Type**
Numeric

**Default**
3

---

### SOURce<cnum>:MODulation<port>:CORRection:COLLection:DISTortion:RECeiver

**Examples**
```
SOUR:MOD1:CORR:COLL:DIST:ITER 3
source2:modulation:correction:collection:distortion:iterations 3
```

**Query Syntax**
```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:DISTortion:RECeiver?
```

**Return Type**
Numeric

**Default**
3
Applicable Models: N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the receiver for distortion calibration.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<rcvr>` String. Calibration plane.

For options S9x09xxA/B, S9x090A/B choose from:
A, B, C, D, R1, R2, R3, R4, a1, a2, a3, a4, b1, b2, b3, b4.

For options S93070xB, choose from:
DUTIn1, DUTOut2, DUTOut3, DUTOut4, DUTOut5.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MOD1:CORR:COLL:DIST:REC "R1"
source2:modulation:correction:collection:distortion:receiver "R1"
```

Query Syntax

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:DISTortion:RECeiver?
```

Return Type

String

Default

- a1 (options S9x09xxA/B, S9x090A/B)
- DUTIn1 (options S93070xB)
Applicable Models: N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the calibration span for a distortion calibration.

Parameters

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<port>** Source port number of the VNA. If unspecified, <port> is set to 1.
- **<num>** Calibration span.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

```
SOUR:MOD1:CORR:COLL:DIST:SPAN 1.5GHz
source2:modulation:correction:collection:distortion:span 1.5ghz
```

Query Syntax

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:DISTortion:SPAN?
```

Return Type

Numeric

Default

0

---

SOURce<cnum>:MODulation<port>:CORRection:COLLection:DISTortion:TOLerance <num>[,srcPort]

Applicable Models: N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the desired distortion calibration tolerance.

Parameters

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<port>** Source port number of the VNA. If unspecified, <port> is set to 1.
- **<num>** Desired tolerance is the un-equalized EVM in dBc.

While this argument can be used to make settings for ALL ports, it is designed
to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:DIST:TOL -50 dBc
source2:modulation:correction:collection:distortion:tolerance -50 dBc
```

### Query Syntax

SOURce<n>:MODulation<port>:CORRection:COLLection:DISTortion:TOLerance?

### Return Type

Numeric

### Default

-40

---

**SOURce<n>:MODulation<port>:CORRection:COLLection:EQUalization:ENABle <bool>[,srcPort]**

**Applicable Models:** N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Enable or disable the equalization calibration state.

**Note:** This command is the same as the SOURce:MODulation:CORRection:COLLection:FLATness:ENABle command.

**Parameters**

- **<ch>** Any existing channel number. If unspecified, value is set to 1.
- **<port>** Source port number of the VNA. If unspecified, <port> is set to 1.
- **<bool>** Choose from:
  - **0 - OFF** - Disable equalization calibration.
  - **1 - ON** - Enable equalization calibration.
- **[srcPort]** **String** (NOT case sensitive). Source port. Optional. Use SOUR:CAT? to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:EQU:ENAB 1
source2:modulation:correction:collection:equalization:enable on
```
**Query Syntax**  
SOURce<cnum>:MODulation<port>:CORRection:COLLection:EQUalization:ENABle ?

**Return Type**  
Boolean

**Default**  
OFF

---

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>SOURce&lt;cnum&gt;:MODulation&lt;port&gt;:CORRection:COLLection:EQUalization:ITERations &lt;num&gt;[,srcPort]</th>
</tr>
</thead>
</table>

**Applicable Models:** N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

*(Read-Write)* Sets and reads the maximum number of iterations used by the calibration routine. The calibration routine uses successive approximation.

**Note:** This command is the same as the SOURce:MODulation:CORRection:COLLection:FLATness:ITERations command.

**Parameters**

- `<cnum>`  
  Any existing channel number. If unspecified, value is set to 1
- `<port>`  
  Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`  
  Maximum number of iterations.
- `[srcPort]`  
  **String** (NOT case sensitive). Source port. Optional. Use SOUR:CAT? to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

<table>
<thead>
<tr>
<th>SOURce&lt;port&gt;:CORRection:COLLection:EQUalization:ITERations</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

**Query Syntax**  
SOURce<cnum>:MODulation<port>:CORRection:COLLection:EQUalization:RECeiver <rcvr>[,srcPort]  

**Return Type**  
Numeric

**Default**  
3
**Applicable Models:** N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the receiver for equalization calibration.

**Note:** This command is the same as the SOURce:MODulation:CORRection:COLLection:FLATness:RECeiver command.

**Parameters**

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<port>`  Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<rcvr>`  **String**. Calibration plane.

  For options S9x09xxA/B, S9x090A/B choose from:

  A, B, C, D, R1, R2, R3, R4, a1, a2, a3, a4, b1, b2, b3, b4.

  For options S93070xB, choose from:

  DUTIn1, DUTOu2t, DUTOu3t, DUTOu4t, DUTOu5t.


  While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```plaintext
SOUR:MOD1:CORR:COLL:EQU:REC "R1"
source2:modulation:correction:collection:equalization:receiver "R1"
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:EQUalization:RECeiver?

**Return Type**  String

**Default**  a1 (options S9x09xxA/B, S9x090A/B)

DUTIn1 (options S93070xB)
**Applicable Models:** N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

*(Read-Write)* Sets and reads the calibration span for an equalization calibration.

**Note:** This command is the same as the `SOURce:MODulation:CORRection:COLLection:FLATness:SPAN` command.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Calibration span.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:EQU:SPAN 1.5GHz
source2:modulation:correction:collection:equalization:span 1.5ghz
```

**Query Syntax** `SOURce<cnum>:MODulation<port>:CORRection:COLLection:EQUalization:SPAN?`

**Return Type** Numeric

**Default** Signal Span

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:EQUalization:TOLerance <num>[,srcPort]
```
Applicable Models: N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09x/A/B, S9x090/A/B

(Read-Write) Sets and reads the desired equalization calibration tolerance.

**Note:** This command is the same as the SOURce:MODulation:CORRection:COLLection:FLATness:TOLerance command.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Calibration equalization tolerance in dB-pk.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:EQU:TOL 0.100
source2:modulation:correction:collection:equalization:tolerance 0.100
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:EQUalization:TOLerance?

**Return Type**

Numeric

**Default**

0.1

```plaintext
SOURce<cnum>:MODulation<port>:CORRection:COLLection:FAST:ENABle <bool>[,srcPort]
```
Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Enable or disable a fast calibration with reduced accuracy.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:

  0 - OFF - Disable fast calibration.

  1 - ON - Enable fast calibration.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

SOUR:MOD1:CORR:COLL:FAST:ENAB 1
source2:modulation:correction:collection:fast:enable on

Query Syntax

SOURce<cnum>:MODulation<port>:CORRection:COLLection:FAST:ENABle ?

Return Type

Boolean

Default

OFF

SOURce<cnum>:MODulation<port>:CORRection:COLLection:FLATness:ENABle <bool>[,srcPort]
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Enable or disable the flatness calibration state.

Note: This command is the same as the SOURce:MODulation:CORRection:COLLection:EQUalization:ENABle command.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - **0 - OFF**: Disable flatness calibration.
  - **1 - ON**: Enable flatness calibration.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

- `SOUR:MOD1:CORR:COLL:FLAT:ENAB 1`
- `source2:modulation:correction:collection:flatness:enable on`

Query Syntax

SOURce<cnum>:MODulation<port>:CORRection:COLLection:FLATness:ENABle?

Return Type

Boolean

Default

OFF

SOURce<cnum>:MODulation<port>:CORRection:COLLection:FLATness:ITERations
<num>[,srcPort]
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the maximum number of iterations used by the calibration routine. The calibration routine uses successive approximation.

**Note:** This command is the same as the SOURce:MODulation:CORRection:COLLection:EQUalization:ITERations command.

### Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Maximum number of iterations.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

### Examples

```
SOUR:MOD1:CORR:COLL:FLAT:ITER 3
```

### Query Syntax

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:FLATness:ITERations?
```

### Return Type

Numeric

### Default

3

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:FLATness:RECeiver <rcvr>[,srcPort]
```
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the receiver for flatness modulation calibration.

Note: This command is the same as the SOURce:MODulation:CORRection:COLLection:EQUalization:RECeiver command.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, <port> is set to 1.
- `<rcvr>` String. Calibration plane.
  - For options S9x09xxA/B, S9x090A/B choose from: A, B, C, D, R1, R2, R3, R4, a1, a2, a3, a4, b1, b2, b3, b4.
  - For options S93070xB, choose from: DUTIn1, DUTOut2, DUTOut3, DUTOut4, DUTOut5.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

```
SOUR:MOD1:CORR:COLL:FLAT:REC "R1"
source2:modulation:correction:collection:flatness:receiver "R1"
```

Query Syntax

SOURce<cnum>:MODulation<port>:CORRection:COLLection:FLATness:RECeiver?

Return Type

String

Default

- a1 (options S9x09xxA/B, S9x090A/B)
- DUTIn1 (options S93070xB)
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the calibration span for a flatness modulation calibration.

Note: This command is the same as the SOURce:MODulation:CORRection:COLLection:EQUalization:SPAN command.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <port> Source port number of the VNA. If unspecified, <port> is set to 1.
- <num> Calibration span.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

- SOUR:MOD1:CORR:COLL:FLAT:SPAN 1.5GHz
- source2:modulation:correction:collection:flatness:span 1.5ghz

Query Syntax

SOURce<cnum>:MODulation<port>:CORRection:COLLection:FLATness:SPAN?

Return Type

Numeric

Default

Signal Span
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the desired flatness modulation calibration tolerance.

Note: This command is the same as the SOURce:MODulation:CORRection:COLLection:EQUalization:TOLerance command.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Calibration flatness tolerance in dB-pk.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MOD1:CORR:COLL:FLAT:TOL 0.100
source2:modulation:correction:collection:flatness:tolerance 0.100
```

Query Syntax

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:FLATness:TOLerance?
```

Return Type

Numeric

Default

0.1
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads a fixed frequency to use for the source modulation calibration.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Fixed frequency to perform calibration.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MOD1:CORR:COLL:FREQ:FIX 10E9
source2:modulation:correction:collection:frequency:fixed 10E9
```

Query Syntax

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:FREQuency[:FIXed]?
```

Return Type

Numeric

Default

Not Applicable

---

SOURce<cnum>:MODulation<port>:CORRection:COLLection:FREQuency:POINts <num>[,srcPort]

Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the number of frequency measurement points to use for a swept frequency source modulation calibration.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Number of frequency points.
While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:FREQ:POIN 5
source2:modulation:correction:collection:frequency:points 5
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:FREQuency:POINts?
```

**Return Type**

Numeric

**Default**

Not Applicable

---

**SOURce<cnum>:MODulation<port>:CORRection:COLLection:FREQuency:STARt <num>[,srcPort]**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

**(Read-Write)** Sets and reads the start frequency to use for a swept frequency source modulation calibration.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, <port> is set to 1.
- `<num>`: Start frequency.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:FREQ:STAR 10E9
source2:modulation:correction:collection:frequency:start 10E9
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:FREQuency:STARt?
```

**Return Type**

Numeric

**Default**

Not Applicable
SOURce<cnum>:MODulation<port>:CORRection:COLLection:FREQuency:STOP
<num>[,srcPort]

Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the stop frequency to use for a swept frequency source modulation calibration.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <port> Source port number of the VNA. If unspecified, <port> is set to 1.
- <num> Stop frequency.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples


Query Syntax

SOURce<cnum>:MODulation<port>:CORRection:COLLection:FREQuency:STOP?

Return Type

Numeric

Default

Not Applicable
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the calibration frequency type to fixed or swept.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - **FIXed**
  - **SWEpt**


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

```
SOUR1:MOD1:CORR:COLL:FREQ:TYPE FIX
```

Query Syntax

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:FREQuency:TYPE?
```

Return Type

Enumeration

Default

FIX

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:LO:FTHRu:ENABle <bool>[,srcPort]
```
Applicable Models: N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Enable or disable the LO feedthru calibration state.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<port> Source port number of the VNA. If unspecified, <port> is set to 1.

<bool> Choose from:

0 - OFF - Disable LO feedthru calibration.

1 - ON - Enable LO feedthru calibration.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

SOUR:MOD1:CORR:COLL:LO:FTHR:ENAB 1

source2:modulation:correction:collection:lo:fthru:enable on

Query Syntax

SOURce<cnum>:MODulation<port>:CORRection:COLLection:LO:FTHRu:ENABLe ?

Return Type

Boolean

Default

OFF

SOURce<cnum>:MODulation<port>:CORRection:COLLection:LO:FTHRu: ITERations <num>[,srcPort]
Applicable Models: N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the maximum number of iterations used by the calibration routine. The calibration routine uses successive approximation.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, <port> is set to 1.
- `<num>` Maximum number of iterations.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:LO:FTHRu:ITERations?

**Return Type** Numeric

**Default** 6

**Syntax**

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:LO:FTHRu :RECeiver <rcvr>[,.srcPort]
```
Applicable Models: N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the receiver for LO feedthru modulation calibration.

Parameters
- <cnum> Any existing channel number. If unspecified, value is set to 1
- <port> Source port number of the VNA. If unspecified, <port> is set to 1.
- <rcvr> **String**. Calibration plane.

For options S9x09xxA/B, S9x090A/B choose from:
A, B, C, D, R1, R2, R3, R4, a1, a2, a3, a4, b1, b2, b3, b4.

For options S93070xB, choose from:
DUTIn1, DUTOut2, DUTOut3, DUTOut4, DUTOut5.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples
```
SOUR:MOD1:CORR:COLL:LO:FTHR:REC "R1"
source2:modulation:correction:collection:lo:fthru:receiver "R1"
```

Query Syntax
SOURce<cnum>:MODulation<port>:CORRection:COLLection:LO:FTHRu:RECeiver?

Return Type
String

Default
a1 (options S9x09xxA/B, S9x090A/B)

DUTIn1 (options S93070xB)
Applicable Models: N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the calibration span for a LO feedthru modulation calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Calibration span.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```plaintext
SOUR:MOD1:CORR:COLL:LO:FTHR:SPAN 1.5GHz
source2:modulation:correction:collection:lo:fthru:span 1.5ghz
```

**Query Syntax**

```plaintext
SOURce<cnum>:MODulation<port>:CORRection:COLLection:LO:FTHRu:SPAN?
```

**Return Type** Numeric

**Default** 0 Hz

Applicable Models: N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the desired LO feedthru modulation calibration tolerance.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Calibration LO feedthru tolerance in dBc

While this argument can be used to make settings for ALL ports, it is designed
to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:LO:FTHR:TOL -40.00
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:LO:FTHRu:TOLerance?
```

**Return Type** Numeric

**Default** -40.00 dBc

---

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Set and read the notch modulation calibration state.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - Disable Notch calibration.
  - 1 - ON - Enable Notch calibration.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:NOT:ENAB 1
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:NOTch:ENABle?
```

**Return Type** Boolean

**Default** OFF

---

**SOURce<cnum>:MODulation<port>:CORRection:COLLection:NOTch:ITERations**

```
4786
```

4786
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the maximum number of iterations used by the calibration routine. The calibration routine uses successive approximation.

Parameters

- `<num>`: Any existing channel number. If unspecified, value is set to 1.
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Maximum number of iterations.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MOD1:CORR:COLL:NOT:ITER 2
source2:modulation:correction:collection:notch:iterations 2
```

Query Syntax

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:NOTch:ITERations?
```

Return Type

Numeric

Default

2

SOURce<cnum>:MODulation<port>:CORRection:COLLection:NOTch:RECeiver <rcvr>[,srcPort]
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

**(Read-Write)** Sets and reads the receiver for an notch modulation calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<rcvr>` **String**. Calibration plane.
  
  For options S9x09xxA/B, S9x090A/B choose from:
  
  A, B, C, D, R1, R2, R3, R4, a1, a2, a3, a4, b1, b2, b3, b4.

  For options S93070xB, choose from:
  
  DUTIn1, DUTOut2, DUTOut3, DUTOut4, DUTOut5.

  
  While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:NOT:REC "R1"
source2:modulation:correction:collection:notch:receiver "R1"
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:NOTch:RECeiver?
```

**Return Type** String

**Default** a1 (options S9x09xxA/B, S9x090A/B)

DUTIn1 (options S93070xB)
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the calibration span for a notch modulation calibration.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Calibration span.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MOD1:CORR:COLL:NOT:SPAN 100MHz
source2:modulation:correction:collection:notch:span 100MHz
```

Query Syntax

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:NOTch:SPAN?
```

Return Type: Numeric

Default: 0

SOURce<cnum>:MODulation<port>:CORRection:COLLection:NOTch:TOLerance <num>[,srcPort]

Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the desired notch calibration tolerance for the notch modulation calibration.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Calibration tolerance in dBc.
While this argument can be used to make settings for ALL ports, it is designed
to access ports such as an external source, true mode balanced port, or one of
the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1
Src2". Otherwise, the <port> argument performs the same function. If both
arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:NOT:TOL -40dBc
source2:modulation:correction:collection:notch:tolerance -40dBc
```

**Query Syntax**

SOURce<chnum>:MODulation<port>:CORRection:COLLection:NOTch:TOLerance?

**Return Type**

Numeric

**Default**

-40

---

**SOURce<chnum>:MODulation<port>:CORRection:COLLection:POWer:ENABle**

<bool>[,srcPort]

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option
S9x09xxA/B/S9x090A/B

(Read-Write) Set and read the power modulation calibration state.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, <port> is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable Power calibration.
  - **1 - ON** - Enable Power calibration.

While this argument can be used to make settings for ALL ports, it is designed
to access ports such as an external source, true mode balanced port, or one of
the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1
Src2". Otherwise, the <port> argument performs the same function. If both
arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:POW:ENAB 1
source2:modulation:correction:collection:power:enable on
```

**Query Syntax**

SOURce<chnum>:MODulation<port>:CORRection:COLLection:POWer:ENABle?

**Return Type**

Boolean
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads a fixed power level to use for the source modulation calibration.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Fixed power level to perform calibration.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

- SOUR:MOD1:CORR:COLL:POW:FIX 0
- source2:modulation:correction:collection:power:fixed 0

Query Syntax

- SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer[:FIXed]?

Return Type

- Numeric

Default

- 0
Applicable Models: N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the maximum number of iterations used by the calibration routine. The calibration routine uses successive approximation.

**Parameters**

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<port>`  Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`  Maximum number of iterations.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:POW:ITER 3
source2:modulation:correction:collection:power:iterations 3
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:ITERations?
```  

**Return Type**  Numeric

**Default**  3

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:POINts
<num>[,srcPort]
```
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

**(Read-Write)** Sets and reads the number of power measurement points to use for a swept power source modulation calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Number of power points.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:POW:POIN 5
source2:modulation:correction:collection:power:points 5
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:POINts?

**Return Type**

Numeric

**Default**

Not Applicable

```
SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:RECeiver
<rcvr>[,srcPort]
```
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the receiver for a power modulation calibration.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<rcvr>` String. Calibration plane.
  
  For options S9x09xxA/B, S9x090A/B choose from:
  
  A, B, C, D, R1, R2, R3, R4, a1, a2, a3, a4, b1, b2, b3, b4.
  
  For options S93070xB, choose from:
  
  DUTIn1, DUTOut2, DUTOut3, DUTOut4, DUTOut5.
  
  
  While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MOD1:CORR:COLL:POW:REC "R1"
source2:modulation:correction:collection:power:receiver "R1"
```

Query Syntax

SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:RECeiver?

Return Type

String

Default

a1 (options S9x09xxA/B, S9x090A/B)

DUTIn1 (options S93070xB)
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the calibration span for a power modulation calibration.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Calibration span.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

```
SOUR:MOD1:CORR:COLL:POW:SPAN 1.5GHz
source2:modulation:correction:collection:power:span 1.5ghz
```

Query Syntax

SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:SPAN?

Return Type

Numeric

Default

Signal Span

---

SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:STARt <num>[,srcPort]

Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the start power level to use for a swept power source modulation calibration.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Start power level.
While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:POW:STAR -20 dBm
source2:modulation:correction:collection:power:start -20 dBm
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:STARt?

**Return Type**

Numeric

**Default**

Not Applicable

---

**SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:STOP <num>[,srcPort]**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the stop power level to use for a swept power source modulation calibration.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Stop power level.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:POW:STOP 10 dBm
source2:modulation:correction:collection:power:stop 10 dBm
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:STOP?

**Return Type**

Numeric

**Default**

Not Applicable
SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:TOLerance
<num>[,srcPort]

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the desired power calibration tolerance for the power modulation calibration.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Calibration tolerance in dB.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:CORR:COLL:POW:TOL 0.100 dB
source2:modulation:correction:collection:power:tolerance 0.100 dB
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:TOLerance?

**Return Type** Numeric

**Default** 0.1

SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:TYPE
<enum>[,srcPort]
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the calibration power type to fixed or swept.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<enum>` Choose from:

  **FIXed**

  **SWEpt**


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

SOUR1:MOD1:CORR:COLL:POW:TYPE FIX

**Query Syntax**

SOURce<cnum>:MODulation<port>:CORRection:COLLection:POWer:TYPE?

**Return Type**

Enumeration

**Default**

FIX

SOURce<cnum>:MODulation<port>:CORRection[:STATe] <bool>[,srcPort]
Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Set and read the modulation correction state.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>`: Choose from:
  - 0 - **OFF** - Disable modulation correction.
  - 1 - **ON** - Enable modulation correction.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

```
SOUR:MOD1:CORR:STAT 1
source:modulation:correction:state on
```

Query Syntax

`SOURce<cnum>:MODulation<port>:CORRection [:STATe]`?

Return Type

Boolean

Default

OFF

```
SOURce<cnum>:MODulation<port>:FILE? [,srcPort]
```
Applicable Models: N522xB, N523xB, N524xB

(Read-only) Returns the most recently used filename. For example, if a new file is loaded, then this file is returned.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

SOUR:MOD:FILE?
source:modulation1:file?

Return Type Comma-separated list of strings.

Default Not applicable

SOURce<cnum>:MODulation<port>:FILE:CORRection:CATalog? [,srcPort]

Applicable Models: All with Option S93070x/B/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-only) Returns a list of the calibrations stored in the .mdx file. Each calibration displayed in the list is for one power level. Calibrations may have been performed on multiple power levels during a single calibration. In this case, multiple calibrations will be saved in the .mdx file. Delete any of these calibrations using the SOURce:MODulation:FILE:CORRection:DELe command.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1
Src2. Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```markdown
SOUR:MOD:FILE:CORR:CAT?
"ModCal_1,ModCal_2,ModCal_3,ModCal_4"
```

**Return Type** String

**Default** Not applicable

---

**SOURce<cnum>:MODulation<port>:FILE:CORRection:DELete <calName>**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Write-only) Deletes any of the calibration files stored in the .mdx file.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<calName>` **String**. Name of the source modulation calibration file.

**Examples**

```markdown
SOUR1:MOD1:FILE:CORR:DEL "ModCal_1"
```

**Query Syntax** Not applicable

**Default** Not applicable

---

**SOURce<cnum>:MODulation<port>:FILE:CORRection:FREQuency? <string>**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-only) Returns the frequency of the specified source modulation calibration file.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<String>` **String**. Source modulation calibration file name.

**Examples**

```markdown
SOUR:MOD:FILE:CORR:FREQ? "ModCal_1"
```

**Return Type** String

**Default** Not applicable

---

4801
**SOURce<cnum>:MODulation<port>:FILE:CORRection:POWer? <string>**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-only)* Returns the power level of the specified source modulation calibration file.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<String>` *String*. Source modulation calibration file name.

**Examples**
```
SOUR:MOD:FILE:CORR:POW? "ModCal_1"
```

**Return Type** String

**Default** Not applicable

---

**SOURce<cnum>:MODulation<port>:FILE:INITialize [,srcPort]**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Write-only)* Resets the modulation file to default values.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**
```
SOUR1:MOD1:FILE:INIT
```

**Query Syntax** Not applicable

**Default** Not applicable

---

**SOURce<cnum>:MODulation<port>:FILE:LOAD <fileName>[,srcPort]**
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Write-only) Specifies the file path to recall a previous modulation file. Current setting are overwritten with those in the file.

Parameters

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<port>** Source port number of the VNA. If unspecified, value is set to 1.
- **<fileName>** String. Modulation file name.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

```
SOUR1:MOD1:FILE:LOAD "C:/modulation/MyModFile.mdx"
```

Query Syntax Not applicable

Default Not applicable

SOURce<cnum>:MODulation<port>:FILE:SAVE <fileName>[,srcPort]

Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Write-only) Specifies the file path and file name to save a modulation file. If the file exists, it is overwritten.

Parameters

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<port>** Source port number of the VNA. If unspecified, value is set to 1.
- **<fileName>** String. Modulation file name.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.
the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR1:MOD1:FILE:SAVE "C:/modulation/MyModFile.mdx"
```

**Query Syntax**

Not applicable

**Default**

Not applicable

---

**SOURce<cnum>:MODulation<port>:FILE:SIGNal:CARRier:OFFSet <num>[,srcPort]**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the Carrier offset value.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, <port> is set to 1.
- `<num>` Carrier offset value.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:FILE:SIGN:CARR:OFFS 1e6
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:FILE:SIGNal:CARRier:OFFSet?

**Return Type**

Numeric

**Default**

0

---

**SOURce<cnum>:MODulation<port>:FILE:SIGNal:COMPact:FILE:NUMBer <num>[,srcPort]**
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the number of compact modulation files to create. This function is useful to create several signals, compare them, then save the best signal.

Parameters
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Number of compact modulation files.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples
```
```

Query Syntax
```
SOURce<cnum>:MODulation<port>:FILE:SIGNal:COMPact:FILE:NUMBer?
```

Return Type
Numeric

Default
1

SOURce<cnum>:MODulation<port>:FILE:SIGNal:COMPact:FILE:SELect <num>[,srcPort]

Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the compact modulation file selection. This function is useful to compare several signals, then save the best signal.

Parameters
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Compact modulation file selection.
While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>1</td>
</tr>
</tbody>
</table>

SOURce<cnum>:MODulation<port>:FILE:SIGNal:COMPact:OFIle <fileName>[,srcPort]

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

**(Read-Write)** Specifies the file path of the original signal from which to create a compact signal.

**Note:** The original signal can be created from Signal Studio, Matlab, SystemVue, etc.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<fileName>` File name of original signal.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

<table>
<thead>
<tr>
<th>Examples</th>
<th>SOUR:MOD1:FILE:SIGN:COMP:OFIle &quot;C:/modulation/MyModFile.mdx&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Syntax</td>
<td>SOURce&lt;cnum&gt;:MODulation&lt;port&gt;:FILE:SIGNal:COMPact:OFIle?</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

S OURce<cnum>:MODulation<port>:FILE:SIGNal:COMPact:OFILe:SRATe <num>[,srcPort]
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the sample rate of the compact modulation file.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Compact modulation file sample rate.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MOD1:FILE:SIGN:COMP:OFIL:SRAT 0
```

Query Syntax

`SOURce<cnum>:MODulation<port>:FILE:SIGNal:COMPact:OFILe:SRATe?`

Return Type: Numeric

Default: 0

---


Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-only) Returns the peak-to-average value of the original signal.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both
arguments are specified, [srcPort] takes priority.

Examples SOUR:MOD:FILE:SIGN:COMP:PAVG?

Return Type Numeric

Default Not applicable


Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-only) Returns the peak-to-average value of the signal created from the original signal.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <port> Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples SOUR:MOD:FILE:SIGN:COMP:PAVG:CALC?

Return Type Numeric

Default Not applicable

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the Peak-to-Avg priority for Compact signals.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - **0** - OFF - Disable priority.
  - **1** - ON - Enable priority

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:FILE:SIGN:COMP:PAVG:PRI ON
```

**Query Syntax**


**Return Type**

Boolean

**Default**

OFF
Applicable Models: N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Allows setup of multiple carriers when defining a multicarrier signal. Select 0 for none, or 1 - 9.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<nnum>`: Subcarrier index (1 - 9).
- `<num>`: Subcarrier number.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:FILE:SIGN:COMP:SUBC:NUMB 1
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:FILE:SIGNal:COMPact:SUBCarrier<nnum>:NUMBer?

**Return Type**

Numeric

**Default**

0

```
```
**Applicable Models:** N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

*(Read-Write)* Sets and reads the offset of the selected subcarrier.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<nnum>` Subcarrier index (1 - 9).
- `<num>` Subcarrier offset.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
```

**Query Syntax**


**Return Type** Numeric

**Default** 0

```
SOURce<cnum>:MODulation<port>:FILE:SIGNal:COMPact:SUBCarrier<nnum>:SPAN <num>[,srcPort]
```
Applicable Models: N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

(Read-Write) Sets and reads the span of the selected subcarrier.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<nnum>` Subcarrier index (1 - 9).
- `<num>` Subcarrier span.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

SOUR:MOD1:FILE:SIGN:COMP:SUBC:NUMB 100E6

Query Syntax

SOURce<cnum>:MODulation<port>:FILE:SIGNal:COMPact:SUBCarrier<nnum>:SPAN?

Return Type

Numeric

Default

0
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads where to start the compact signal within the original signal. The compact signal is a slice of the original signal.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Compact signal start time.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:FILE:SIGN:COMP:TIME:STAR 0
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:FILE:SIGNal:COMPact:TIME:STARt?

**Return Type** Numeric

**Default** 0

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-only) Returns the calculated start time of the signal created from the original signal.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.
the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>


Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the start time priority for Compact signals.

Parameters
- <cnum>: Any existing channel number. If unspecified, value is set to 1
- <port>: Source port number of the VNA. If unspecified, <port> is set to 1.
- <bool>: Choose from:
  - **0**: OFF - Disable priority.
  - **1**: ON - Enable priority

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>OFF</td>
</tr>
</tbody>
</table>

SOURce<cnum>:MODulation<port>:FILE:SIGNal:DAC:SCALing <num>[,srcPort]
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the scaling factor used for the waveform (full scale = 100%). This ensures that the DAC filter does not output a signal that is larger than the DAC's maximum output level, which can cause distortion in the system. Setting the scaling factor to 100% will usually cause excessive distortion.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` DAC scale as a percentage of full scale.
- `[srcPort]` **String** *(NOT case sensitive)*. Source port. Optional. Use SOUR:CAT? to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

SOUR:MOD1:FILE:SIGNal:DAC:SCAL 70

**Query Syntax**

SOURce<cnum>:MODulation<port>:FILE:SIGNal:DAC:SCALing?

**Return Type** Numeric

**Default** 70%
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Write-only) Sets the NPR notch location type for the selected NPR Notch modulation type.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - **SYMMetric** - Symmetric locates the notch in the center of the signal span.
  - **ACARrier** - Avoid Carrier locates the notch near the center of the signal span but will be shifted to avoid the LO carrier feedthrough.
  - **CUSTom** - Allows the user to define the offset of the notch.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

```
SOUR1:MOD1:FILE:SIGN:NPR:NOTC1:LOC CUST
```

Query Syntax Not applicable

Default SYMMetric

SOURce<cnum>:MODulation<port>:FILE:SIGNal:NPR:NOTCh[1-3]:NUMBer <num>[,srcPort]
(Read-Write) Sets and reads the number of NPR notches in the modulated signal. A notch can be up to 10% of the Signal Span.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Number of notches (1 - 3)

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:FILE:SIGN:NPR:NOTC1:NUMB 1
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:FILE:SIGNal:NPR:NOTCh:NUMBer?
```

**Return Type**

Numeric

**Default**

1

(Read-Write) Sets and reads the NPR notch offset frequency of the selected notch. This offset is the center frequency of the selected notch relative to the LO carrier frequency. Typically, the notch will have a 0 Hz offset, meaning it is centered on the LO carrier. If you have more than one notch, you can offset some of the notches from the carrier. For example, if you have three notches 1 MHz wide, you might set their offsets to -10 MHz, 0 MHz and +10 MHz so that they are spaced across the wideband carrier.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
**SOURce<cnum>:MODulation<port>:FILE:SIGNal:NPR:NOTCh[1-3]:SPAN**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the span of the selected notch.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` NPR notch offset.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:FILE:SIGN:NPR:NOTC:OFFS 0
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:FILE:SIGNal:NPR:NOTCh:SPAN?
```

**Return Type** Numeric

**Default** 0
**Default** 10 MHz

```
SOURce<cnum>:MODulation<port>:FILE:SIGNal:OPTimize:ENABle <bool>[,srcPort]
```

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Enables or disables signal optimization settings. When enabled, the calculated modulated signal will be optimized according to the constraints defined in this group box. If disabled, then the signal will not be optimized.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable signal optimization for NPR Notch modulation type.
  - **1 - ON** - Enable signal optimization for other modulation types (Compact, Flat Tones).

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:FILE:SIGN:OPT:ENAB ON
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:FILE:SIGNal:OPTimize:ENABle?
```

**Return Type** Boolean

**Default** OFF

```
```
**Applicable Models:** N524xB models with Option S93070xB, and/or N522xB, N523xB, N524xB with Option S9x09xxA/B, S9x090A/B

*(Read-Write)* Enables or disables a brick-wall filter for spectral leakage (for Compact signals only). The brick-wall filter is applied to the band-power span calculated for the signal. The brick-wall filter cuts of signals outside this span. Disable filter to retain signals outside the calculated band power span.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable brick-wall filter.
  - **1 - ON** - Enable brick-wall filter.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:FILE:SIGN:OPT:FILT:ENAB ON
```

**Query Syntax**

```
```

**Return Type** Boolean

**Default** ON
### Applicable Models
All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the number of decimal digits limit for calculated frequencies.

#### Parameters
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Number of decimal digits.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

#### Examples

#### Query Syntax

#### Return Type
Numeric

#### Default
2

---

### Applicable Models
All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Enables or disables the number of decimal digits limit for calculated frequencies.

#### Parameters
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - Disable decimal digit limit setting.
  - 1 - ON - Enable decimal digit limit setting.
return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

### Examples

```
```

### Query Syntax

```
```

### Return Type

Boolean

### Default

OFF

---


*Applicable Models:* All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the frequency tolerance value (in percent).

#### Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Tolerance value.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

### Examples

```
SOUR:MOD1:FILE:SIGN:OPT:FREQ:TOL 1
```

### Query Syntax

```
```

### Return Type

Numeric

### Default

1%

---

**SOURce<cnum>:MODulation<port>:FILE:SIGNal:OPTimize:HREJect**

*Parameters*

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Tolerance value.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

### Examples

```
SOUR:MOD1:FILE:SIGN:OPT:FREQ:TOL 1
```

### Query Syntax

```
SOURce<cnum>:MODulation<port>:FILE:SIGNal:OPTimize:HREJect <num>[,srcPort]
```

### Return Type

Numeric

### Default

1%
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the number of test signal harmonics you want to be protected against. This adds constraints to the list of LOs used to cover the span.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Number of test signal harmonics to reject.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

```
SOUR:MOD1:FILE:SIGN:OPT:HREJ 5
```

Query Syntax

SOURce<cnum>:MODulation<port>:FILE:SIGNal:OPTimize:HREJect?

Return Type
Numeric

Default
5

---


Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the tone spacing less than or equal to the value (Hz).

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Distance between each tone.


While this argument can be used to make settings for ALL ports, it is designed
to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples
SOUR:MOD1:FILE:SIGN:OPT:MAX:TONE:SPAC 100 kHz

Query Syntax

Return Type
Numeric

Default
100 kHz


Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the minimum number of tones.

Parameters
- <cnum> Any existing channel number. If unspecified, value is set to 1
- <port> Source port number of the VNA. If unspecified, <port> is set to 1.
- <num> Number of tones.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

Query Syntax

Return Type
Numeric

Default
1001

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the minimum waveform period. This command minimizes the period of the waveform greater than or equal to the value (seconds).

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Waveform period value.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
```

**Query Syntax**

```
```

**Return Type**

Numeric

**Default**

10 usec

---


**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Enables or disables the rejection of Nyquist frequencies. This ensures that Nyquist images of the signal tones in the IF bandwidth are not falling back on top of real signal frequencies.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
<bool> Choose from:

0 - OFF - Disable Nyquist frequency rejection.

1 - ON - Enable Nyquist frequency rejection.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples SOUR:MOD1:FILE:SIGN:OPT:NYQR:ENAB ON


Return Type Boolean

Default OFF

SOURce<cnum>:MODulation<port>:FILE:SIGNal:OPTimize:TYPE <enum>[,srcPort]

Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Write-only) Sets the optimize signal type.

Parameters

<enum> Any existing channel number. If unspecified, value is set to 1

<port> Source port number of the VNA. If unspecified, value is set to 1.

<enum> Choose from:

MIWPeriod - (Min Waveform Period) Minimizes the period of the waveform greater than or equal to the value (seconds).

MITNumber - (Min Number of Tones) Minimizes the number of tones greater than or equal to the value. This will ignore the Number of Tones selection.

MATSpacing - (Max Tone Spacing) Maximizes the tone spacing greater than or equal to the value (Hz). This will ignore the Tone Spacing selection.

FTOLerance - (Frequency Tolerance) Set frequency tolerance value (in percent).

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR1:MOD1:FILE:SIGN:OPT:TYPE FTOL
```

**Query Syntax**

Not applicable

**Default**

FTOLerance

---


**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-only)* Returns the calculated peak-to-average value.

**Parameters**

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<port>`  Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD:FILE:SIGN:PAVG:CALC?
```

**Return Type**

Numeric

**Default**

Not applicable

---

SOURce<cnum>:MODulation<port>:FILE:SIGNal:PHASe:FIXed <num>[,srcPort]
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the phase when Fixed phase is the Phase Type.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Phase setting.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MOD1:FILE:SIGN:PHAS:FIX 0
```

Query Syntax

SOURce<cnum>:MODulation<port>:FILE:SIGNal:PHAse:FIXed?

Return Type

Numeric

Default

0

---

SOURce<cnum>:MODulation<port>:FILE:SIGNal:PHAse:RANDom:SEED <num>[,srcPort]

Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the phase seed when Random phase is the Phase Type.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Phase setting.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MODE2:FILE:SIGN:PHAS:RAND 10
```

Query Syntax

SOURce<cnum>:MODulation<port>:FILE:SIGNal:PHAse:RANDom?

Return Type

Numeric

Default

0

---
the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:FILE:SIGN:PHAS:RAND:SEED 1
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:FILE:SIGNal:PHASe:RANDom:SEED?

**Return Type**

Numeric

**Default**

1

---

**SOURce<cnum>:MODulation<port>:FILE:SIGNal:PHAsE:TYPE <enum>[,srcPort]**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Write-only) Sets the phase type.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<enum>` Choose from:
  - RANDom
  - FIXed
  - PARabolic

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR1:MOD1:FILE:SIGN:PHAS:TYPE RAND
```

**Query Syntax**

Not applicable

**Default**

RANDom

---

**SOURce<cnum>:MODulation<port>:FILE:SIGNal:SPAN <num>[,srcPort]**
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the signal span.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Signal span setting.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

SOUR:MOD1:FILE:SIGN:SPAN 100e6

**Query Syntax**

SOURce<cnum>:MODulation<port>:FILE:SIGNal:SPAN?

**Return Type** Numeric

**Default** 100 MHz

---


**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-only)* Returns the calculated signal span.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.
arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD:FILE:SIGN:SPAN:CALC?
```

**Return Type** Numeric

**Default** Not applicable

---

**SOURce<cnum>:MODulation<port>:FILE:SIGNal:SPAN:PRIority <bool>[,srcPort]**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the signal span priority.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable priority.
  - **1 - ON** - Enable priority

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:FILE:SIGN:SPAN:PRI ON
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:FILE:SIGNal:SPAN:PRIority?
```

**Return Type** Boolean

**Default** ON

---

**SOURce<cnum>:MODulation<port>:FILE:SIGNal:SRATe <num>[,srcPort]**
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the source sample rate.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>` Source sample rate setting.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MOD1:FILE:SIGN:SRAT 200e6
```

Query Syntax

```
SOURce<cnum>:MODulation<port>:FILE:SIGNal:SRATe?
```

Return Type

Numeric

Default

200 MHz

SOURce<cnum>:MODulation<port>:FILE:SIGNal:SRATe:AUTO <bool> [,srcPort]

Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Set and read the auto sample rate.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - Disable auto sample rate.
  - 1 - ON - Enable auto sample rate.
While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**  
SOUR:MOD:FILE:SIGN:SRAT:CALC?

**Query Syntax**  
SOURce<cnum>:MODulation<port>:FILE:SIGNal:SRATe:CALCulated? [srcPort]

**Return Type**  
Numeric

**Default**  
Not applicable

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**  
SOUR:MOD:FILE:SIGN:SRAT:CALC?

**Return Type**  
Numeric

**Default**  
Not applicable
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the number of tones. This setting is related to the span and tone spacing: \( \text{Number of Tones} = \frac{\text{Signal Span}}{\text{Tone Spacing}} + 1 \).

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Number of tones setting.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:FILE:SIGN:TONE:NUMB 1001
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:FILE:SIGNal:TONE:NUMBer?
```

**Return Type**

Numeric

**Default**

1001

---


Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-only) Returns the calculated number of tones.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<port>`: Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1
Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**


**Return Type** Numeric

**Default** Not applicable


**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-only) Returns the calculated spacing between the tones.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**


**Return Type** Numeric

**Default** Not applicable

**SOURce<cnum>:MODulation<port>:FILE:SiGNal:TONE:NUMBer:PRIority**

<bool>[,srcPort]
Applicable Models: All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the tone number priority.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable priority for Compact modulation type.
  - **1 - ON** - Enable priority for NPR Notch and Flat Tone modulation types.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

Examples

```
SOUR:MOD1:FILE:SIGN:TONE:NUMB:PRI ON
```

Query Syntax

```
SOURce<cnum>:MODulation<port>:FILE:SIGNal:TONE:NUMBer:PRIority?
```

Return Type

Boolean

Default

OFF

```
SOURce<cnum>:MODulation<port>:FILE:SIGNal:TONE:SPACing <num>[,srcPort]
```
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the tone spacing.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<num>`: Tone spacing setting.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:FILE:SIGN:TONE:SPACing 100e3
```

**Query Syntax**

`SOURce<cnum>:MODulation<port>:FILE:SIGNal:TONE:SPACing?`

**Return Type**

Numeric

**Default**

100 kHz

---

**SOURce<cnum>:MODulation<port>:FILE:SIGNal:TONE:SPACing:PRIority**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the tone spacing priority.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<port>`: Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>`: Choose from:
  - **0 - OFF**: Disable priority for NPR Notch and Flat Tone modulation types.
  - **1 - ON**: Enable priority for Compact modulation type.
return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOUR:MOD1:FILE:SIGN: TONE:SPAC:PRI ON</td>
</tr>
</tbody>
</table>

**Query Syntax**

SOURce<cnum>:MODulation<port>:FILE:SIGNal:TONE:SPACing:PRIority?

**Return Type**

Boolean

**Default**

OFF

---

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOUR:MOD1:FILE:SIGN: TONE:ALL:STAT 1</td>
</tr>
</tbody>
</table>

**Query Syntax**

SOURce<cnum>:MODulation<port>:FILE:TONE:ALL [:STATE] <bool> [.srcPort]

**Return Type**

Boolean

---

(Read-Write) Set all tone states to on or off.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable all tones.
  - **1 - ON** - Enable all tones.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOUR:MOD1:FILE: TONE:ALL:STAT 1</td>
</tr>
</tbody>
</table>

**Query Syntax**

SOURce<cnum>:MODulation<port>:FILE:TONE:ALL [:STATE] <bool> [.srcPort]

**Return Type**

Boolean
**SOURce<cnum>:MODulation<port>:FILE:TONE:COUNt? [,srcPort]**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-only)* Returns the number of tones.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**
```
SOUR:MOD:FILE:TONE:COUN?
```

**Return Type** Numeric

**Default** Not applicable

---

**SOURce<cnum>:MODulation<port>:FILE:TONE:FREQuency? <toneNum>[,srcPort]**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-only)* Returns the tone frequency in Hz relative to the carrier.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<toneNum>` Tone number.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.
the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

SOUR:MOD:FILE:TONE:FREQ? 118

Return Type Numeric

Default 0

```
SOURce<cnum>:MODulation<port>:FILE:TONE:LOAD <fileName> [,srcPort]
```

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Write-only)* Loads the specified multitone file.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<fileName>` **String**. Name of the file to be loaded (csv).

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

Examples

SOUR1:MOD1:FILE:TONE:LOAD "C:/modulation/MyToneFile.csv"

Query Syntax Not applicable

Default Not applicable

```
SOURce<cnum>:MODulation<port>:FILE:TONE:PHASe <toneNum>,<phaseVal>[,srcPort]
```
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

(Read-Write) Sets and reads the phase in degrees of the specified tone number.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<toneNum>` Tone number.
- `<phaseVal>` Phase value in degrees.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:FILE:TONE:PHAS 10,45
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:FILE:TONE:PHASe? <toneNum>

**Return Type**

Numeric

**Default**

- 0 - toneNum
- 0 - phaseVal

**SOURce<cnum>:MODulation<port>:FILE:TONE:POWer**

```
SOURce<cnum>:MODulation<port>:FILE:TONE:POWer <toneNum>,<powerVal>[,srcPort]
```

4841
**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Read-Write)* Sets and reads the power in dBm of the specified tone number.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<toneNum>` Tone number.
- `<powerVal>` Power value in dBm.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR:MOD1:FILE:TONE:POW 10,-30
```

**Query Syntax**

SOURce<cnum>:MODulation<port>:FILE:TONE:POWer? <toneNum>

**Return Type**

Numeric

**Default**

0 - toneNum
0 - powerVal

```
SOURce<cnum>:MODulation<port>:FILE:TONE:SAVE <fileName>[,srcPort]
```
(Write-only) Saves the specified multitone file.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<fileName>` *String*. Name of the file to be saved (csv).

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR1:MOD1:FILE:TONE:SAVE "C:/modulation/MyToneFile.csv"
```

**Query Syntax**

Not applicable

**Default**

Not applicable

---

(SOURce)<cnum>:MODulation<port>:FILE:TONE[:STATe] <toneNum>,<bool> [,srcPort]

(Read-Write) Set specified tone state to on or off.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - 0 - OFF - Disable tone.
  - 1 - ON - Enable tone.
- `<toneNum>` Tone number.
While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

```
SOUR:MOD1:FILE:TONE:STAT 10,1
source:modulation:file:tone:state 10, on
```

**Query Syntax**

```
SOURce<cnum>:MODulation<port>:FILE:TONE[ :STATe]? <toneNum>
```

**Return Type**

Boolean

**Default**

0 - toneNum

1 - bool

---

**SOURce<cnum>:MODulation<port>:FILE:TYPE <enum> [,srcPort]**

**Applicable Models:** All with Option S93070xB/9x070A/B, and/or All with Option S9x09xxA/B/S9x090A/B

*(Write-only)* Sets the modulation type.

**Parameters**

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<port>`  Source port number of the VNA. If unspecified, value is set to 1.
- `<enum>`  Choose from:
  - **COMPact** *(Compact)* Shortened version of any type of modulation. Compact signals cut a slice of the IQ data from an original waveform.
  - **FLATtones** *(Flat Tones)* This signal is a set of constant amplitude tones over a defined signal span.
  - **NPRNotch** *(NPR Notch)* This signal is a set of constant amplitude tones over a defined signal span where a subset of those tones are set to zero over a notch span.

While this argument can be used to make settings for ALL ports, it is designed
to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

SOUR1:MOD1:FILE:TYPE COMP

**Query Syntax**

Not applicable

**Default**

NPRNotch

---

### SOURce<cnum>\:MODulation<port>:LOAD <fileName> [,.srcPort]

**Applicable Models:** N522xB, N523xB, N524xB, All with S9x070A/B

*(Write-only)* Loads the specified modulation file.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<fileName>` **String**. Name of the file to be loaded.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [srcPort] takes priority.

**Examples**

SOUR1:MOD1:LOAD "C:/modulation/MyModFile.mdx"

**Query Syntax**

Not applicable

**Default**

Not applicable

---

### SOURce<cnum>\:MODulation<port>:SAVE <file>[,.srcPort]

---

4845
Applicable Models: N522xB, N523xB, N524xB, All with S9x070A/B

(Write-only) Saves the specified modulation file.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Source port number of the VNA. If unspecified, value is set to 1.
- `<file>` *String*. Name of the file to be saved.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[srcPort]` takes priority.

**Examples**

```
SOUR1:MOD1:SAVE "C:/modulation/MyModFile.mdx"
```

**Query Syntax**

Not applicable

**Default**

Not applicable

---

SOURce<cnum>:MODulation<port>[:STATe] <bool> [,srcPort]

**Applicable Models:** N522xB, N523xB, N524xB, All with S9x070A/B

(Read-Write) Set and read the modulation state.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Source port number of the VNA. If unspecified, `<port>` is set to 1.
- `<bool>` Choose from:
  - **0 - OFF** - Disable I/Q modulation.
  - **1 - ON** - Enable I/Q modulation.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1
Src2\". Otherwise, the \textless port\textgreater{} argument performs the same function. If both arguments are specified, [srcPort] takes priority.

<table>
<thead>
<tr>
<th>Examples</th>
<th>SOUR:MOD1:STAT 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>source:modulation:state on</td>
</tr>
</tbody>
</table>

| Query Syntax   | SOURce\textless cnum\textgreater{}:MODulation\textless port\textgreater{} [:STATe]? [srcPort] |
| Default        | OFF               |

| Return Type    | Boolean           |
Source: Power: Correction Commands

Used to perform source power calibration on internal and external sources.

**Note:** Only ONE Source Power Cal can be performed at a time.

<table>
<thead>
<tr>
<th>COLLect</th>
<th>ABORt</th>
<th>ACQuire</th>
<th>AVERage</th>
</tr>
</thead>
<tbody>
<tr>
<td>[COUNt]</td>
<td>[COUNt]</td>
<td>[NTOLerance]</td>
<td></td>
</tr>
<tr>
<td>[COUNT]</td>
<td></td>
<td>DISPlay</td>
<td></td>
</tr>
<tr>
<td>[STATe]</td>
<td></td>
<td>FCHeck</td>
<td></td>
</tr>
<tr>
<td>[STATe]</td>
<td></td>
<td>ITERation</td>
<td></td>
</tr>
<tr>
<td>[COUNT]</td>
<td></td>
<td>METHod</td>
<td></td>
</tr>
<tr>
<td>[NTOLerance]</td>
<td></td>
<td>SAVE</td>
<td></td>
</tr>
<tr>
<td>SENSor</td>
<td></td>
<td>WARN</td>
<td></td>
</tr>
<tr>
<td>[FRANge]</td>
<td></td>
<td>DATA</td>
<td></td>
</tr>
<tr>
<td>RCFactor</td>
<td></td>
<td>PRIor</td>
<td></td>
</tr>
<tr>
<td>SELect</td>
<td></td>
<td>DATA</td>
<td></td>
</tr>
<tr>
<td>TABLe</td>
<td></td>
<td>LOSS</td>
<td></td>
</tr>
<tr>
<td>[STATe]</td>
<td></td>
<td>POINts?</td>
<td></td>
</tr>
<tr>
<td>[STATe]</td>
<td></td>
<td>SELect</td>
<td></td>
</tr>
</tbody>
</table>
Click on a keyword to view the command details.

Blue commands are superseded.

See Also

- Example program using these commands.
- Template for creating your own Power Meter Driver
- Learn about Source Power Cal
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**Note:** The SOURce:POWer:CORRection:COLLect:ACQuire command, used to step the VNA and read a power meter, cannot be sent over the GPIB unless the power meter is connected to a different GPIB interface. See the alternative methods described in the command details.

---

**SOURce<ch>:POWer<port>:CORRection:COLLect:ABORt**

**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

(Write-only) Aborts a source power calibration sweep that is in progress.

To use this ABORt command, you MUST use the ASYNchronous argument with SOUR:POW:CORR:COLL:ACQ

After aborting, this message appears in the error log: **+243,"Requested operation was canceled".**

**Parameters**

- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1
- `<port>` Port number to correct for source power. If unspecified, value is set to 1.

**Examples**

```
SOUR:POW:CORR:COLL:ABOR
source1:power2:correction:collect:abort
```

**Query Syntax** Not Applicable

**Default** Not Applicable
SOURce<ch>:POWer<port>:CORRection:COLLect[:ACQuire] <char>,<id>[,[src][,[sync]]

**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Write-only)* Initiates a source power cal acquisition sweep using the power sensor attached to the specified channel (A or B) on the power meter, using a USB power sensor, or using the specified VNA receiver.

For source power cal, the power meter can NOT be controlled by the VNA using the GPIB Talker/Listener interface. Instead use one of the following methods:

- If present, use the GPIB dedicated controller port.
- Connect the power meter to the VNA using a USB / GPIB interface (Keysight 82357A).
- SCPI programming of the VNA using a LAN Client interface (see example).
- Send SCPI commands through the COM interface using the SCPI String Parser object.
- Directly control the Power Meter and VNA to step frequency; then acquire and store the Power reading. (see example).
- Configure the Power Meter/Sensor as a PMAR Device. Learn how. See SCPI commands.

**Parameters**

- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1
- `<port>` Port number to correct for source power. If unspecified, value is set to 1.
- `<char>` Acquisition Choose from:
  - **PMETer** - Power Meter is used for all readings.
  - **PMReceiver** - Power meter for the first iteration; then use the reference receiver for remaining readings if necessary (same as "fast iteration" box checked on dialog box)
  - **RECeiver** - Use VNA measurement receiver for all readings.
- `<id>` String (Not case sensitive). The power sensor or VNA receiver to use for measuring power.

For **PMETer** or **PMRECeiver**, choose from:

- "ASENSOR" or "BSENSOR". For U series USB sensors, always specify "ASENSOR"

For **RECeiver**, choose from:
• Any VNA receiver to acquire readings using physical or logical receiver notation.
• Any configured PMAR device name. Learn more about PMAR Devices. See PMAR commands.


While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

[sync] If this argument is specified, must also specify [src].

Choose from:

• **SYNchronous** - Blocks SCPI commands during standard measurement (default behavior).
• **ASYNchronous** - Does NOT block SCPI commands during standard measurement.

Learn more about this argument

**Examples**
```
SYST:COMM:PSEN gpib, "13"
SOURce:POWer:CORRection:COLLect:FCheck:STATe ON
sour:pow:corr:coll:ASEN:FRANge 4e9, 26.5e9
sour:pow:corr:coll:BSEN:FRANge 0, 4.2e9
SOUR:POW3:CORR:COLL:ACQ PMR,"BSENSOR";*OPC?
SOUR:POW3:CORR:COLL:ACQ PMR,"ASENSOR";*OPC?
SOUR:POW3:CORR:COLL:SAVE RREC
```

**Query Syntax**
Not Applicable

**Default**
Not Applicable

**SOURce:POWer<port>:CORRRection:COLLect:AVERage[:COUNt] <num>**

**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

(Read-Write) This command, along with SOUR:POW:CORR:COLL:AVER:NTOLerance, allows for settling of the power sensor READINGS.
Note: This command is global and does not depend on a specific channel number.

Sets the maximum number of acquisitions that will be used to acquire one settled reading from the power meter.

These settings affect every use of the power meter (PMAR and source power cal).

This setting and corresponding SOUR:POW:CORR:COLL:AVER:NTOLerance command only effect the settled reading of the currently selected legacy power meter.


Users may want to adjust this number if they know the signal is noisy as these settings set a threshold that determines when the power meter reading is done.

Each reading is averaged with the previous readings. When this average meets the Average:NTOLerance value or this number of readings has been made, the average is returned as the valid reading.

Learn more.

Parameters

<port> If provided, this argument is ignored by the VNA.

<num> Maximum number of readings to make to allow for settling. Choose any number between 3 and 1000.

Examples

// configure the power meter settling (up to 2 acquisitions to produce one settled meter reading)
SOUR:POW:CORR:COLL:AVER 2
SOUR:POW:CORR:COLL:AVER:NTOL .05

// configure the number of (settled) readings to acquire at each frequency point.

// 3 settled readings are averaged to produce one bucket of data per frequency
SOUR:POW:CORR:COLL:ITER 3
SOUR:POW:CORR:COLL:ITER:NTOL .005

Query Syntax SOURce:POWer:CORRection:COLLect:AVERage[:COUNt]?

Return Type Numeric

Default 3
**SOURce:POWer<port>:CORRection:COLLect:AVERage:NTOLerance <num>**

**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-Write)* This command, along with SOUR:POW:CORR:COLL:AVER:COUNT, allows for settling of the power sensor READINGS.

**Note:** This command is global and does not depend on a specific channel number.

This setting and corresponding SOUR:POW:CORR:COLL:AVER:COUNT command only effect the settled reading of the currently selected legacy power meter.

**Note:** To set the COUNT/NTOLerance of a specific PMAR, use the SYST:CONF:EDEV:PMAR:READ:COUNT and SYST:CONF:EDEV:PMAR:READ:NTOLerance commands.

Each power reading is averaged with the previous readings. When the average meets this nominal tolerance value or the max number of readings has been made, the average is returned as the valid reading.

Learn more.

**Parameters**

- `<port>` If provided, this argument is **ignored** by the VNA.
- `<num>` Power measurement settling tolerance value in dB. Choose any number between 0 and 5.

**Examples**

// configure the power meter settling (up to 2 acquisitions to produce one settled meter reading)

SOUR:POW:CORR:COLL:AVER 2
SOUR:POW:CORR:COLL:AVER:NTOL 0.05

// configure the number of (settled) readings to acquire at each frequency point.

// 3 settled readings are averaged to produce one bucket of data per frequency

SOUR:POW:CORR:COLL:ITER 3
SOUR:POW:CORR:COLL:ITER:NTOL 0.005

**Query Syntax** SOURce:POWer:CORRection:COLLect:AVERage:NTOLerance?

**Return Type** Numeric
SOURce<ch>:POWer<port>:CORRection:COLLect:DISPlay[:STATe] <ON | OFF>

**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-Write)* Enables and disables the display of power readings on the VNA screen. Send this command BEFORE you begin a source power cal acquisition. After the source power cal data is acquired, this setting is reset to ON.

**Parameters**
- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1
- `<port>` If provided, this argument is ignored by the VNA.
- `<ON|OFF>` **ON (1)** Source power calibration dialog box is shown on the VNA screen. Power readings are plotted against the Tolerance value as limit lines.
  - **OFF (0)** - Source power calibration dialog box is NOT shown on the VNA screen.

**Examples**
```
SOUR:POW:CORR:COLL:DISP ON
source1:power2:correction:collect:display:state off
```

**Query Syntax** SOURce:POWer:CORRection:COLLect:DISPlay[:STATe]?

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** ON (1)

---

SOURce<ch>:POWer<port>:CORRection:COLLect:FCHeck[:STATe] <ON | OFF>

**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-Write)* Enables and disables frequency checking of source power cal acquisition sweeps. ONLY use when you have more than one power sensor.

**Parameters**
- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1
- `<port>` If provided, this argument is ignored by the VNA.
- `<ON|OFF>` **ON (1)** turns source power cal frequency checking ON. A requested acquisition will only succeed for those frequency points which fall within a frequency range specified for the power sensor being used. An acquisition will pause in mid-sweep if the frequency is about to exceed the maximum frequency limit specified for that sensor. When the sweep is paused in this state...

**Examples**
```
SOUR:POW:CORR:COLL:FCHE ON
source1:power2:correction:collect:frequency:check:state off
```

**Query Syntax** SOURce:POWer:CORRection:COLLect:FCHeck[:STATe]?

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** OFF (0)
manner, a sensor connected to the other channel input of the power meter can be connected to the measurement port in place of the previous sensor, and used to complete the sweep. However, the maximum frequency specified for the second sensor would need to be sufficient for the sweep to complete. Frequency limits are specified using the SOUR:POW:CORR:COLL:SEN command.

**OFF (0)** - turns source power cal frequency checking OFF. An acquisition will use just one power sensor for the entire sweep, regardless of frequency.

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Syntax</td>
<td>SOURce:POWer:CORRection:COLLect:FCHeck[:STATe]?</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean (1 = ON, 0 = OFF)</td>
</tr>
<tr>
<td>Default</td>
<td>OFF (0)</td>
</tr>
</tbody>
</table>

**SOURce<ch>:POWer<port>:CORRection:COLLect:ITERation[:COUNt] <num>**

**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-Write)* This command, along with SOUR:POW:CORR:COLL:ITER:NTOL control the number of settled readings taken to produce a single power point during source power cal.

The source power cal reads the power (performed by SOUR:POW:CORR:COLL:AVER:COUNt and SOUR:POW:CORR:COLL:AVER:NTOLerance ) and makes internal adjustments to set the power to a desired level. These settings determine how many attempts (COUNt) the analyzer will make in an attempt to get close enough (NTOLerance ) to the target power level.

Learn more.

**Parameters**

- **<ch>** If provided, this argument is **ignored** by the VNA.
- **<port>** If provided, this argument is **ignored** by the VNA.
- **<num>** Maximum number of readings. Choose any number between 1 and 1000.

<table>
<thead>
<tr>
<th>Examples</th>
<th>// configure the power meter settling (up to 2 acquisitions to produce one settled meter reading) SOUR:POW:CORR:COLL:AVER 2 SOUR:POW:CORR:COLL:AVER:NTOL .05 // configure the number of (settled) readings to acquire at each frequency point.</th>
</tr>
</thead>
</table>

4855
// 3 settled readings are averaged to produce one bucket of data per frequency
SOUR:POW:CORR:COLL:ITER 3
SOUR:POW:CORR:COLL:ITER:NTOL .005

**Query Syntax**  SOURce:POWer:CORRection:COLLect:ITERation[:COUNt]?

**Return Type**  Numeric

**Default**  1

SOURce<ch>:POWer<port>:CORRection:COLLect:ITERation:NTOLerance <num>

**Applicable Models:**  N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-Write)* This command, along with SOUR:POW:CORR:COLL:ITER:COUNT describes the number of adjustments to make to the source power.

Sets the maximum desired deviation from the sum of the test port power and the offset value. Power READINGS (performed by SOUR:POW:CORR:COLL:AVER:COUNT and SOUR:POW:CORR:COLL:AVER:NTOLerance ) will continue to be made, and source power adjusted, until a measurement is within this tolerance value or the max number of measurements has been met. The last value is the valid measurement for that data point.

Learn more.

**Parameters**
- `<ch>` If provided, this argument is *ignored* by the VNA.
- `<port>` If provided, this argument is *ignored* by the VNA.
- `<num>` Tolerance value in dBm. Choose any number between 0 and 5

**Examples**

```
// configure the power meter settling (up to 2 acquisitions to produce one settled meter reading)
SOUR:POW:CORR:COLL:AVER 2
SOUR:POW:CORR:COLL:AVER:NTOL .05

// configure the number of (settled) readings to acquire at each frequency point.

// 3 settled readings are averaged to produce one bucket of data per frequency
SOUR:POW:CORR:COLL:ITER 3
SOUR:POW:CORR:COLL:ITER:NTOL .005
```
<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>SOURce:POWer:CORRection:COLLect:ITERation:NTOLerance?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>.05</td>
</tr>
</tbody>
</table>

**SOURce<ch>:POWer<port>:CORRection:COLLect:METHod <char> Superseded**

**Applicable Models:** N522xB, N523xB, N524xB

This command is replaced with SOUR:POW:CORR:COLLect[:ACQuire] which now specifies the method and the device. The only parameter required by that command was either ASENsor or BSENsor which are still supported but not documented.

*(Read-Write)* Selects the calibration method to be used for the source power cal acquisition.

**Parameters**

- `<ch>`: Channel number of the source power cal. If unspecified, value is set to 1
- `<port>`: Port number to correct for source power. If unspecified, value is set to 1.
- `<char>`: Choose from:
  - **NONE**: No Cal method
  - **PMETer**: Power Meter is used for all readings. (same as "fast iteration" box not checked on dialog box)
  - **PMReceiver**: Power meter for the first iteration; then use the reference receiver for remaining readings if necessary (same as "fast iteration" box checked on dialog box)

**Examples**

```plaintext
SOUR:POW:CORR:COLL:METH PMET
SOURce1:power2:correction:collect:method pmreceiver
```

**Query Syntax**

SOURce<ch>:POWer<port>:CORRection:COLLect:METHod?

**Return Type**

Character

**Default**

NONE

**SOURce<ch>:POWer<port>:CORRection:COLLect:SAVE [<RREC>]**
**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Write-only)* Applies the array of correction values after a source power calibration sweep has completed. The source power correction will then be active on the specified source port for channel `<ch>`. This command does NOT save the correction values. To save correction values, save an instrument / calibration state (*.cst file) after performing a source power cal.

**Parameters**

- `<ch>`: If provided, this argument is **ignored** by the VNA.
- `<port>`: If provided, this argument is **ignored** by the VNA.
- `<RREC>`: Optional argument.

**RRECiever** In addition to a source Power Cal, perform a calibration of the reference receiver used in the measurement. ONLY the Reference Receiver calibration is then saved to a Cal Set or Cal Register as specified by the current setting of SENS:CORR:PREF:CSET:SAVE.

This argument only applies to standard S-parameter channels.

**Examples**

```
SOUR:POW:CORR:COLL:SAVE
source:power:correction:collect:save rreceiver
```

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**SOURce<ch>:POWer<port>:CORRection:COLLect:<pmChan>SENsor[:FRANge] <num1>,<num2>**

*Applicable Models:* N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-Write)* Specifies the frequency range over which the power sensors connected to the specified channels (A and B) of the power meter can be used (minimum frequency, maximum frequency). If the power meter has only a single channel, that channel is considered channel A.

**Parameters**

- `<ch>`: Channel number of the source power cal. If unspecified, value is set to 1
- `<port>`: If provided, this argument is **ignored** by the VNA. (It is required for query).
- `<pmChan>`: Power Meter channel. Choose from:
  - **A** - Channel A
  - **B** - Channel B
Minimum frequency for the sensor. If a frequency unit is not specified, Hz is assumed.

Maximum frequency for the sensor. If a frequency unit is not specified, Hz is assumed.

**Examples**

```
SOUR:POW:COR:COLL:ASEN 100E3, 3E9
source1:power:correction:collect:bsensor:frange 10 MHz, 18 GHz
```

**Query Syntax**

```
SOURce<ch>:POWer<port>:CORRection:COLLect:ASENsor[:FRANge]?
SOURce<ch>:POWer<port>:CORRection:COLLect:BSENsor[:FRANge]?
```

**Return Type**  
Numeric

**Default**  
0,0

---

**SOURce<ch>:POWer<port>:CORRection:COLLect:<pmChan>SENsor:RCFactor <num>**

**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-Write)*

Specifies the reference cal factor for the power sensor connected to channel A or B of the power meter. If the power meter has only a single channel, that channel is considered channel A.

**Note:** If the sensor connected to the specified channel of the power meter contains cal factors in EPROM (such as the Keysight E-series power sensors), those will be the cal factors used during the calibration sweep. The reference cal factor value associated with this command, and any cal factors entered into the VNA for that sensor channel, will not be used.

**Parameters**

- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1
- `<port>` If provided, this argument is **ignore**d by the VNA.
- `<pmChan>` Power Meter channel. Choose from:
  
  **A** - Channel A
  
  **B** - Channel B

- `<num>` Reference cal factor in percent. Choose any number between 1 and 150.

**Examples**

```
SOUR:POW:COR:COLL:ASEN:RCF 98.7
source1:power2:correction:collect:bsensor:rcfactor 105
```

**Query Syntax**

```
SOURce:POWer:CORRection:COLLect:ASENsor:RCFactor?
SOURce:POWer:CORRection:COLLect:BSENsor:RCFactor?
```

**Return Type**  
Numeric
SOURce<ch>:POW<port>:CORRection:COLLect:<pmChan>SENsor:SELect

Applicable Models: N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

(Read-Write) Sets and returns the power sensor channel (A or B) to be used. This performs the same function as the Use this sensor only checkbox in the Power Sensor Settings dialog.

Notes:

- This command is NOT necessary when performing a Guided Power Cal using Multiple Sensors.
- This command can be used with Application channels.

Parameters

- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1
- `<port>` If provided, this argument is ignored by the VNA.
- `<pmChan>` Power Meter channel. Choose from:
  - A - Channel A
  - B - Channel B

Examples

- SOUR:POW:CORR:COLL:<pmChan>SEN:SEL 'Write

-sourcem1:power2:correction:collect:bsensor:select? 1e9 'Read

Query Syntax

SOURce:POWer:CORRection:COLLect:ASENsor:SELect? <Frequency>

SOURce:POWer:CORRection:COLLect:BSENsor:SELect? <Frequency>

Returns a boolean 1 or 0 (ON or OFF) indicating whether the sensor is to be used at the specified frequency.

If frequency checking is OFF, then the <Frequency> parameter is ignored. The query returns if the sensor is selected for ALL frequencies.

Return Type

Numeric

Default
Not Applicable

SOURce<ch>:POW<port>:CORRection:COLLect:TABLE:DATA <data>
**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-Write)* Read or write data into the selected table. Use `SOUR:POW:CORR:COLL:TABL:SELeet` to select a table.

- When the power sensor table is selected, the data is interpreted as cal factors in *percent*.
- When the loss table is selected, POSITIVE values in dB are interpreted as LOSS. To compensate for gain, use negative values.
- Each table can contain up to 9999 segments. Values can be loaded using the Characterize Adapter macro.
- Learn more about Power Loss Compensation.

**Parameters**

- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1
- `<port>` If provided, this argument is ignored by the VNA.
- `<data>` Data to write into the selected table.

**Examples**

```
SOURce:POWer:CORRection:COLLect:TABLE:DATA 0.12, 0.34, 0.56
```

**Query Syntax**

```
SOURce<ch>:POWer:CORRection:COLLect:TABLE:DATA?
```

If the selected table is currently empty, no data is returned.

**Return Type** Numeric - one number per table segment.

**Default** Not Applicable

---

**SOURce<ch>:POWer<port>:CORRection:COLLect:TABLE:FREQuency <data>**

**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-Write)* Read or write frequency values for the selected table (cal factor table for a power sensor, or the loss compensation table). Use `SOUR:POW:CORR:COLL:TABL:SELeet` to select a table.

**Parameters**

- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1
- `<port>` If provided, this argument is ignored by the VNA.
- `<data>` Frequency data to write into the selected table.

**Examples**

```
SOURce:POWer:CORRection:COLLect:TABLE:FREQuency 10E6, 1.5E9, 9E9
```

**Query Syntax**

```
SOURce<ch>:POWer:CORRection:COLLect:TABLE:FREQuency?
```

If the selected table is currently empty, no data is returned.
**Return Type**  Numeric - one number per table segment

**Default**  Not Applicable

**SOURce<ch>:POWer<port>:CORRection:COLLect:TABLE:LOSS[:STATe] <ON | OFF>**

**Applicable Models:**  N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-Write)* Indicates whether or not to adjust the power readings using the values in the loss table during a source power cal sweep. Learn more about Power Loss Compensation.

**Parameters**

- `<ch>`  Channel number of the source power cal. If unspecified, value is set to 1
- `<port>`  If provided, this argument is ignored by the VNA.
- `<ON|OFF>`  ON (or 1) - turns use of the loss table ON.
  
  OFF (or 0) - turns use of the loss table OFF.

**Examples**

```
SOUR:POW:CORR:COLL:LOSS ON
source1:power2:correction:collect:table:loss:state off
```

**Query Syntax**  SOURce:POWer:CORRection:COLLect:TABLE:LOSS[:STATe]?

**Return Type**  Boolean (1 = ON, 0 = OFF)

**Default**  OFF (0)

---

**SOURce<ch>:POWer<port>:CORRection:COLLect:TABLE:POINts?**

**Applicable Models:**  N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-only)* Returns the number of segments that are currently in the selected table.

**Parameters**

- `<ch>`  Channel number of the source power cal. If unspecified, value is set to 1
- `<port>`  If provided, this argument is ignored by the VNA.

**Examples**

```
SOUR:POW:CORR:COLL:TABLE:POIN?
source1:power2:correction:collect:table:points?
```

**Return Type**  Numeric

**Default**  0

---

**SOURce<ch>:POWer<port>:CORRection:COLLect:TABLE[:SELect] <char>**
Applicable Models: N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

(Read-Write) Selects which table you want to write to or read from. Read or write using SOURce:POWer:CORRection:COLLect:TABLE:FREQuency and SOURce:POWer:CORRection:COLLect:TABLE:DATA

Parameters

<ch> Channel number of the source power cal. If unspecified, value is set to 1

<port> If provided, this argument is ignored by the VNA.

<char> Choose from:

NONE - No table selected

ASENsor - Cal Factor table for Power Sensor A

BSENSor - Cal Factor table for Power Sensor B

LOSS - Loss compensation table

Examples

SOUR:POW:CORR:COLL:TABLE ASEN
sourc1:power2:correction:collect:table:select bsensor

Query Syntax SOURce:POWer:CORRection:COLLect:TABLE[:SE lect]?

Return Type Character

Default NONE

SOURce<ch>:POWer<port>:CORRection:COLLect:WARN <bool>

Applicable Models: N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

(Read-Write) Enables/disables the use of error messages during a source calibration if the calibration fails to achieve the desired power level at the power sensor.

This property affects error reporting during the acquisition of a source power calibration.

When the power calibration sweep occurs, the tolerance set by “SOUR:POW:CORR:COLL:ITER:NTOL “ is indicated by a set of limit lines.

When those limits fail and SOUR:POW:CORR:COLL:WARN is set to OFF, the failure is not reported. This is the default.

When those limits fail and SOUR:POW:CORR:COLL:WARN is set to ON, the failure is reported to the SCPI error queue.
**Parameters**

- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1.
- `<port>` Port number to correct for source power. If unspecified, value is set to 1.
- `<bool>`
  - **ON (or 1)** - enables SCPI error on source power calibration failure.
  - **OFF (or 0)** - disables SCPI error on source power calibration failure.

**Examples**

```
SOUR:POW:CORR:COLL:WARN ON
source1:power2:correction:collect:warn off
```

**Query Syntax**

```
SOURce:POWer:CORRection:COLLect:WARN?
```

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF (0)

---

**SOURce<ch>:POWer<port>:CORRection:DATA <data>[,src]**

**Applicable Models:** All

(Read-Write) Writes and reads source power calibration data.

The effect from this command on the channel is immediate. Do NOT send
SOUR:POW:CORR:COLL:SAVE after this command as it may invalidate the uploaded data.

When querying source power calibration data, if no source power cal data exists for the specified
channel and source port, then no data is returned.

If a change in the instrument state causes interpolation and/or extrapolation of the source power cal,
the correction data associated with this command correspond to the new instrument state (interpolated
and/or extrapolated data).

If the channel is sweeping the source backwards, then the first data point is the highest frequency
value; the last data point is the lowest. Use the SENS:X:VALues? command to return the X-axis
values in the displayed order.

**Parameters**

- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1.
- `<port>` Port number to correct for source power. If unspecified, value is set to 1.
- `<data>` Correction Data
While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
SOURce1:POWer2:CORRection:DATA 0.12, -0.34, 0.56
```

**Query Syntax**

```
SOURce<ch>:POWer<port>:CORRection:DATA? [src]
```

**Return Type**

Depends on FORMat:DATA command

**Default**

Not Applicable

---

**SOURce<ch>:POWer<port>:CORRection:DATA:PRIor <data>[,src]**

**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-Write)* Writes and reads power correction values from the previous iteration of the source power cal. Data for which the first power meter reading were within the tolerance limit, the prior correction value is 0.

In all other respects, this command is the same as SOUR:POW:CORR:DATA .

This command can be used to determine the final power reading at each point of the power cal, for a cal that did not pass tolerance limits. The formula for determining the power reading (in dB):

Power reading = Target power at the source port + specified power cal offset value + prior iteration corr value actual power corr value.

The "actual" value in this equation is returned with SOUR:POW:CORR:DATA?

**Parameters**

- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1
- `<port>` Port number to correct for source power. If unspecified, value is set to 1.
- `<data>` Correction Data

While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
SOURce1:POWer2:CORRection:DATA:PRIor 0.12, -0.34, 0.56
```
Query Syntax  SOURce<ch>:POWer<port>:CORRection:DATA:PRIor? [src]
Return Type  Depends on FORMat:DATA command
Default  Not Applicable

SOURce<ch>:POWer<port>:CORRection:LEVel[:AMPLitude] <num>[,src]

Applicable Models:  N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

(Read-Write) Specifies the power level that is expected at the desired reference plane (DUT input or output). This is not used for segment sweep with independent power levels or power sweeps.

Note: Although this command still works, it is recommended that you specify cal power by setting the test port power and offset value.

Parameters

- `<ch>`  Channel number of the source power cal. If unspecified, value is set to 1
- `<port>`  Port number to correct for source power. If unspecified, value is set to 1.
- `<num>`  Cal power level in dBm. Because this could potentially be at the output of a device-under-test, no limits are placed on this value here. It is realistically limited by the specifications of the device (power sensor) that will be used for measuring the power. The power delivered to the VNA receiver must never exceed VNA specifications for the receiver!

While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

SOUR:POW:CORR:LEV 10
source1:power2:correction:level:amplitude 0 dbm

Query Syntax  SOURce:POWer:CORRection:LEVel[:AMPLitude]? [src]
Return Type  Numeric
Default  0 dBm

SOURce<ch>:POWer<port>:CORRection:OFFSet[:MAGNitude] <num>[,src]

4866
Applicable Models: N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

(Read-Write) Sets or returns a power level offset from the VNA test port power. This can be a gain or loss value (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier at the input of your DUT.

Cal power is the sum of the test port power setting and this offset value. Following the calibration, the VNA power readouts are adjusted to the cal power.

Parameters

- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1
- `<port>` Port number to correct for source power. If unspecified, value is set to 1.
- `<num>` Gain or loss value in dB. Choose a value between -200 and 200

While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

Examples

SOUR:POW:CORR:OFFS 10
source1:power2:correction:offset:magnitude -3

Query Syntax

SOURce:POWer:CORRection:OFFSet[:MAGNitude]? [src]

Return Type

Numeric

Default

0 dB

SOURce<ch>:POWer<port>:CORRection[:STATe] <bool>[,src]
**Applicable Models:** N522xB, N523xB, N524xB, E5080, M980xA, P50xxA, M9485A

*(Read-Write)* Enables and disables source power correction for the specified port on the specified channel.

**Parameters**

- `<ch>`  Channel number of the source power cal. If unspecified, value is set to 1
- `<port>` Port number to correct for source power. If unspecified, value is set to 1.
- `<bool>` ON (or 1) turns source power correction ON.
  
  OFF (or 0) - turns source power correction OFF.

While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

**Examples**

SOUR:POW:CORR ON
source1:power2:correction:state off, "MXG N5183A"

**Query Syntax**

SOURce:POWer:CORRection[:STATe]? "MXG N5183A"

**Return Type**  Boolean (1 = ON, 0 = OFF)

**Default**  OFF (0)
The status registers enable you to query the state of selected events that occur in the analyzer.

**Note:** This documentation requires familiarity with the "Standard Status Data Structure - Register Model" as defined in IEEE Std 488.2-1992. Also, first read [Learn about Status Registers](#).

**STATus:**

OPERation

| AVERaging

| DEFine

| USER

| DEVice

PRESet

QUESTionable

| DEFine
Click on a red keyword to view the command details.

See Also

- Example Programs
- Learn about Status Registers
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

Notes:

- Any bit not shown in the registers is not used but may be reserved for future use.
- The SCPIStringParser can NOT be used with SCPI Status Reporting. However, the *OPC? will work.
STATus:OPERation<keyword>

Applicable Models: All

Summarizes conditions in the Averaging and Operation:Define:User<1|2|3> event registers.

<table>
<thead>
<tr>
<th>&lt;keyword&gt;</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>:CONDition?</td>
<td>STAT:OPER:COND?</td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:OPER:ENAB 1024</td>
</tr>
<tr>
<td>[:EVENT]?</td>
<td>STAT:OPER?</td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt;</td>
<td>STAT:OPER:NTR 1024</td>
</tr>
<tr>
<td>:PTRansition &lt;bits&gt;</td>
<td>STAT:OPER:PTR 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>256</td>
<td>Averaging summary</td>
<td>either enabled bit in the Averaging summary event register is set to 1</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>User Defined summary</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>Device summary</td>
<td>either enabled bit in the Device summary event register is set to 1</td>
</tr>
</tbody>
</table>

STATus:OPERation:AVERaging<n> <keyword>

Applicable Models: All

Monitors and summarizes the status of Averaging on traces 1 to 580. When averaging for a trace is complete, the representative bit is set to 1.

Bit 0 is used to summarize the status in the registers that follow. For example, Average Register 3, bit 0, summarizes the status from registers 4 through 42.

All enable bits are set to 1 by default.

To find the measurement number, use Calc:Par:Mnum.

<table>
<thead>
<tr>
<th>&lt;n&gt;</th>
<th>Averaging Register. Choose from 1 to 42</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;keyword&gt;</td>
<td>Example</td>
</tr>
<tr>
<td>:CONDition?</td>
<td>STAT:OPER:AVER1:COND?</td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:OPER:AVER1:ENAB 1024</td>
</tr>
</tbody>
</table>
### Averaging Register <n>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>...</th>
<th>41</th>
<th>42</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>42</td>
</tr>
<tr>
<td>1</td>
<td>2-42</td>
<td>15</td>
<td>29</td>
<td>43</td>
<td>57</td>
<td>71</td>
<td>85</td>
<td>99</td>
<td>...</td>
<td>575</td>
<td>561</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>2</td>
<td>30</td>
<td>44</td>
<td>58</td>
<td>72</td>
<td>86</td>
<td>100</td>
<td>...</td>
<td>576</td>
<td>562</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>3</td>
<td>17</td>
<td>31</td>
<td>45</td>
<td>59</td>
<td>73</td>
<td>87</td>
<td>101</td>
<td>...</td>
<td>577</td>
<td>563</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>4</td>
<td>18</td>
<td>32</td>
<td>46</td>
<td>60</td>
<td>74</td>
<td>88</td>
<td>102</td>
<td>...</td>
<td>578</td>
<td>564</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>5</td>
<td>19</td>
<td>33</td>
<td>47</td>
<td>61</td>
<td>75</td>
<td>89</td>
<td>103</td>
<td>...</td>
<td>579</td>
<td>565</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>6</td>
<td>20</td>
<td>34</td>
<td>48</td>
<td>62</td>
<td>76</td>
<td>90</td>
<td>104</td>
<td>...</td>
<td>580</td>
<td>566</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>7</td>
<td>21</td>
<td>35</td>
<td>49</td>
<td>63</td>
<td>77</td>
<td>91</td>
<td>105</td>
<td>...</td>
<td>581</td>
<td>567</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>8</td>
<td>22</td>
<td>36</td>
<td>50</td>
<td>64</td>
<td>78</td>
<td>92</td>
<td>106</td>
<td>...</td>
<td>582</td>
<td>568</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>9</td>
<td>23</td>
<td>37</td>
<td>51</td>
<td>65</td>
<td>79</td>
<td>93</td>
<td>107</td>
<td>...</td>
<td>583</td>
<td>569</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>10</td>
<td>24</td>
<td>38</td>
<td>52</td>
<td>66</td>
<td>80</td>
<td>94</td>
<td>108</td>
<td>...</td>
<td>584</td>
<td>570</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>11</td>
<td>25</td>
<td>39</td>
<td>53</td>
<td>67</td>
<td>81</td>
<td>95</td>
<td>109</td>
<td>...</td>
<td>585</td>
<td>571</td>
</tr>
</tbody>
</table>

**Bit is set to 1 when the following conditions exist:**

- **Summary Bit** - If any bit from that register fails, it propagates to the previous register, bit 0.

### Trace Numbers

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>...</th>
<th>41</th>
<th>42</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>29</td>
<td>43</td>
<td>57</td>
<td>71</td>
<td>85</td>
<td>99</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>16</td>
<td>30</td>
<td>44</td>
<td>58</td>
<td>72</td>
<td>86</td>
<td>100</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>8</td>
<td>3</td>
<td>17</td>
<td>31</td>
<td>45</td>
<td>59</td>
<td>73</td>
<td>87</td>
<td>101</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>16</td>
<td>4</td>
<td>18</td>
<td>32</td>
<td>46</td>
<td>60</td>
<td>74</td>
<td>88</td>
<td>102</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>32</td>
<td>5</td>
<td>19</td>
<td>33</td>
<td>47</td>
<td>61</td>
<td>75</td>
<td>89</td>
<td>103</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>64</td>
<td>6</td>
<td>20</td>
<td>34</td>
<td>48</td>
<td>62</td>
<td>76</td>
<td>90</td>
<td>104</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>128</td>
<td>7</td>
<td>21</td>
<td>35</td>
<td>49</td>
<td>63</td>
<td>77</td>
<td>91</td>
<td>105</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>256</td>
<td>8</td>
<td>22</td>
<td>36</td>
<td>50</td>
<td>64</td>
<td>78</td>
<td>92</td>
<td>106</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>512</td>
<td>9</td>
<td>23</td>
<td>37</td>
<td>51</td>
<td>65</td>
<td>79</td>
<td>93</td>
<td>107</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1024</td>
<td>10</td>
<td>24</td>
<td>38</td>
<td>52</td>
<td>66</td>
<td>80</td>
<td>94</td>
<td>108</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>2048</td>
<td>11</td>
<td>25</td>
<td>39</td>
<td>53</td>
<td>67</td>
<td>81</td>
<td>95</td>
<td>109</td>
<td>...</td>
</tr>
</tbody>
</table>

Averaging on this trace is complete.
To determine Register, Bit number, and Weight for trace numbers between 113 and 560 (not shown in the above table) use the following calculations.

The averaging status for trace numbers higher than 580 can NOT be tracked.

The following example calculates the Register, Bit number, and Bit Weight for trace # 400:

- To determine **Register** number, use \(((\text{Trace} \# - 1) / 14) + 1\).
- To determine **Bit Number**, use the remainder +1 of the above calculation.
- \(((\text{400}-1)/14) + 1 = \text{Register}# r+1\text{Bit}\)
  - \(399/14 = 28 \text{ r}7\)
  - \(28+1= \text{Register 29}\)
  - \(7+1= \text{Bit number 8}\)
- To determine **Bit Weight**: Use above table. For example: Bit 8 = **256**

<table>
<thead>
<tr>
<th>Trace #</th>
<th>Register</th>
<th>Bit number</th>
<th>Averaging status</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>4096</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Averaging on this trace is complete</td>
</tr>
</tbody>
</table>

**STATus:OPERation:DEFine<keyword>**

**Applicable Models**: N522xB, N523xB, N524xB, M937xA, P937xA

Summarizes conditions in the OPERation:Define:User<1|2|3> event registers.

<table>
<thead>
<tr>
<th>&lt;keyword&gt;</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>:CONDition?</td>
<td>STAT:OPER:DEF:COND?</td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:OPER:DEF:ENAB &lt;bits&gt;</td>
</tr>
<tr>
<td>[:EVENt]?</td>
<td>STAT:OPER:DEF?</td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt;</td>
<td>STAT:OPER:DEF:NTR &lt;bits&gt;</td>
</tr>
<tr>
<td>:PTRansition &lt;bits&gt;</td>
<td>STAT:OPER:DEF:PTR &lt;bits&gt;</td>
</tr>
</tbody>
</table>
Bit | Weight | Description | Bit is set to 1 when the following conditions exist:
--- | --- | --- | ---
1 | 2 | USER1 | any bit in the USER1 event register is set to 1
2 | 4 | USER2 | any bit in the USER2 event register is set to 1
3 | 8 | USER3 | any bit in the USER3 event register is set to 1

**STATus:OPERation:DEFine:USER<1|2|3><keyword>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

Monitors conditions that you define and map in any of the three OPER:DEF:USER event registers.

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>:ENABle &lt;bits&gt;</td>
</tr>
<tr>
<td>[:EVENT]?</td>
</tr>
<tr>
<td>:MAP &lt;bit&gt;,&lt;error&gt;</td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt;</td>
</tr>
<tr>
<td>:PTRansition &lt;bits&gt;</td>
</tr>
</tbody>
</table>

Bit | Weight | Description | Bit is set to 1 when the following conditions exist:
--- | --- | --- | ---
0 | 1 | for user | user defined
1 | 2 | for user | user defined
2 | 4 | for user | user defined
3 | 8 | for user | user defined
4 | 16 | for user | user defined
5 | 32 | for user | user defined
6 | 64 | for user | user defined
7 | 128 | for user | user defined
8 | 256 | for user | user defined
9 | 512 | for user | user defined
10 | 1024 | for user | user defined
11 | 2048 | for user | user defined
STATus:OPERation:DEVice<keyword>

Applicable Models: All

Summarizes conditions in the OPERation:DEVice event registers.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>Unused</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>Unused</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>Unused</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>Unused</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>Sweep Completed</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>Unused</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>Unused</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>Unused</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>Unused</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>Unused</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>Unused</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>Unused</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>Unused</td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
<td>Unused</td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
<td>Unused</td>
</tr>
</tbody>
</table>

Bit is set to 1 when the following conditions exist:
STATus:PRESet

Applicable Models: All

(Write-only) Initializes all the status registers.

Example  STAT: PRES

STATus:QUESTionable:<keyword>

Applicable Models: All

Summarizes conditions that monitor the quality of measurement data.

<keyword>  Example
:CONDition?  STAT: QUES:COND?
:ENABLE <bits>  STAT: QUES:ENAB 1024
[:EVENT]?  STAT: QUES?
:NTRansition <bits>  STAT: QUES:NTR 1024
:PTRansition <bits>  STAT: QUES:PTR 0

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>512</td>
<td>Integrity Reg summary</td>
<td>any enabled bit in the Integrity event register is set to 1</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>Limit Registers summary</td>
<td>any enabled bit in the Limit event registers is set to 1</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>Define Registers summary</td>
<td>any enabled bit in the Define event registers is set to 1</td>
</tr>
</tbody>
</table>
STATus:QUEStionable:DEFINE<keyword>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

Summarizes conditions in the Questionable:Define:User<1|2|3> event registers.

<table>
<thead>
<tr>
<th>&lt;keyword&gt;</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>:CONDition?</td>
<td>STAT:QUES:DEF:COND?</td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:QUES:DEF:ENAB 1024</td>
</tr>
<tr>
<td>[:EVENt]?</td>
<td>STAT:QUES:DEF?</td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt;</td>
<td>STAT:QUES:DEF:NTR 1024</td>
</tr>
<tr>
<td>:PTRansition &lt;bits&gt;</td>
<td>STAT:QUES:DEF:PTR 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>USER1</td>
<td>any bit in the USER1 event register is set to 1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>USER2</td>
<td>any bit in the USER2 event register is set to 1</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>USER3</td>
<td>any bit in the USER3 event register is set to 1</td>
</tr>
</tbody>
</table>

STATus:QUEStionable:DEFINE:USER<1|2|3><keyword>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

Monitors conditions that you define and map in any of the three QUES:DEF:USER event registers.

<table>
<thead>
<tr>
<th>&lt;keyword&gt;</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:QUES:DEF:USER1:ENABle 1024</td>
</tr>
<tr>
<td>[:EVENt]?</td>
<td>STAT:QUES:DEF:USER1?</td>
</tr>
<tr>
<td>:MAP &lt;bit&gt;,&lt;error&gt;</td>
<td>STAT:QUES:DEF:USER1:MAP 0,-113 'when error -113 occurs, bit 0 in USER1 will set to 1.</td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt;</td>
<td>STAT:QUES:DEF:USER1:NTR 1024</td>
</tr>
<tr>
<td>:PTRansition &lt;bits&gt;</td>
<td>STAT:QUES:DEF:USER1:PTR 0</td>
</tr>
</tbody>
</table>

4877
<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
<td>for user</td>
<td>user defined</td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
<td>for user</td>
<td>user defined</td>
</tr>
</tbody>
</table>

**STATus:QUEStionable:INTegrity <keyword>**

**Applicable Models:** All

Summarizes conditions in the Measurement Integrity register.

<table>
<thead>
<tr>
<th>&lt;keyword&gt;</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>:CONDition?</td>
<td>STAT:QUES:INT:COND?</td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:QUES:INT:ENAB 1024</td>
</tr>
<tr>
<td>[:EVENt]?</td>
<td>STAT:QUES:INT?</td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt;</td>
<td>STAT:QUES:INT:NTR 1024</td>
</tr>
<tr>
<td>Bit</td>
<td>Weight</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

**STATus:QUESTionable:INTegrity:HARDware<keyword>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

Monitors the status of hardware failures.

```
Example

:CONDition?
STAT:QUES:INT:HARD:COND?

:ENABle <bits>
STAT:QUES:INT:HARD:ENAB 1024

[:EVENt]?
STAT:QUES:INT:HARD?

:NTRansition <bits>
STAT:QUES:INT:HARD:NTR 1024

:PTRansition <bits>
STAT:QUES:INT:HARD:PTR 0
```

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>Phase Unlock</td>
<td>the source has lost phaselock, possibly caused by a reference channel open or a hardware failure.</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>Unleveled</td>
<td>the source power is unleveled. This could be caused by a source set for more power than it can deliver at the tuned frequency. Or it could be caused by a hardware failure.</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>Not used</td>
<td>N/A</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>EE Write Failed</td>
<td>an attempted write to the EEPROM has failed, possibly caused by a hardware failure.</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>Not used</td>
<td>N/A</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>Ramp Cal Failed</td>
<td>the analyzer was unable to calibrate the analog ramp generator due to a possible hardware failure.</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>Not used</td>
<td>N/A</td>
</tr>
</tbody>
</table>
STATus:QUEStionable:INTegrity:MEASurement<n> <keyword>

Applicable Models: All

Note: This register can be used ONLY with standard S-parameter measurements.

Monitors the lag between changing a channel setting and when the data is ready to query.

When you change the channel state (start/stop freq, bandwidth, and so forth), then the questionable bit for that channel is set. This indicates that your desired channel state does not yet match the data you would get if querying a data trace. When the next sweep is complete (without aborting in the middle), and the data trace matches the channel state that produced it, the bit is cleared for that channel.

<n> Measurement register number. Choose from 1 to 3

<keyword> Example

:CONDition?
STAT:QUES:INT:MEAS1:COND?

:ENABle <bits>
STAT:QUES:INT:MEAS2:ENAB 1024

[:EVENt]?
STAT:QUES:INT:MEAS3?

:NTRansition <bits>
STAT:QUES:INT:MEAS2:NTR 1024

:PTRansition <bits>
STAT:QUES:INT:MEAS1:PTR 0

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td>Summary from Meas Reg 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>15</td>
<td>29</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>3</td>
<td></td>
<td></td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>4</td>
<td></td>
<td></td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>5</td>
<td></td>
<td></td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>6</td>
<td></td>
<td></td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
</tbody>
</table>
STATus:QUESTionable:LIMit<n> <keyword>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

Monitors and summarizes the status of limit line failures. When a trace fails, the representative bit is set to 1.

Bit 0 is used to summarize failures in the registers that follow. For example, Limit Register 3, bit 0, summarizes the failures from registers 4 through 42.

All enable bits are set to 1 by default.

To find the measurement number, use Calc:Par:Mnum

<n> Limit register: Choose from 1 to 42.

Example

:CONDition? STAT:QUES:LIM4:COND?
:ENABle <bits> STAT:QUES:LIM1:ENAB 1024
<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>43</td>
</tr>
<tr>
<td></td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>46</td>
</tr>
<tr>
<td></td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>49</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>567</td>
</tr>
<tr>
<td></td>
<td>568</td>
</tr>
<tr>
<td></td>
<td>569</td>
</tr>
<tr>
<td></td>
<td>570</td>
</tr>
</tbody>
</table>

### Limit Register <n>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>41</td>
</tr>
<tr>
<td></td>
<td>42</td>
</tr>
</tbody>
</table>

**Bit is set to 1 when the following conditions exist:**

Summary Bit - If any bit from that register fails, it propagates to the previous register, bit 0.

### Trace Numbers

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>561</td>
</tr>
<tr>
<td></td>
<td>562</td>
</tr>
<tr>
<td></td>
<td>563</td>
</tr>
<tr>
<td></td>
<td>564</td>
</tr>
<tr>
<td></td>
<td>565</td>
</tr>
<tr>
<td></td>
<td>566</td>
</tr>
<tr>
<td></td>
<td>567</td>
</tr>
<tr>
<td></td>
<td>568</td>
</tr>
<tr>
<td></td>
<td>569</td>
</tr>
<tr>
<td></td>
<td>570</td>
</tr>
</tbody>
</table>

Any point on trace fails the limit test.
<p>| | | | | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>2048</td>
<td>11</td>
<td>25</td>
<td>39</td>
<td>53</td>
<td>67</td>
<td>81</td>
<td>95</td>
<td>109</td>
<td>...</td>
<td>571</td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>12</td>
<td>26</td>
<td>40</td>
<td>54</td>
<td>68</td>
<td>82</td>
<td>96</td>
<td>110</td>
<td>...</td>
<td>572</td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
<td>13</td>
<td>27</td>
<td>41</td>
<td>55</td>
<td>69</td>
<td>83</td>
<td>97</td>
<td>111</td>
<td>...</td>
<td>573</td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
<td>14</td>
<td>28</td>
<td>42</td>
<td>56</td>
<td>70</td>
<td>84</td>
<td>98</td>
<td>112</td>
<td>...</td>
<td>574</td>
<td>--</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

To determine Register, Bit number, and Weight for trace numbers between 113 and 560 (not shown in the above table) use the following calculations.

The limit status for trace numbers higher than 580 can NOT be tracked.

The following example calculates the Register, Bit number, and Bit Weight for trace # 400:

- To determine Limit **Register** number, use \(((\text{Trace} \# - 1) / 14) + 1\).
- To determine Limit **Bit Number**, use the remainder +1 of the above calculation.
- \(((400-1)/14) + 1 = \text{Register\# r+1Bit}\)
  - 399/14 = 28 r7
  - 28+1= **Register 29**
  - 7+1= **Bit number 8**
- To determine Limit **Bit Weight**: Use above table. For example: Bit 8 = **256**

**STATus:QUEStionable:LSUMmary:<keyword>**

**Applicable Models:** All

Summary register of limit test, ripple test and bandwidth test. bit 0: summary bit for the limit test. bit 1: summary bit for the ripple limit test. bit 2: summary bit for the bandwidth limit test.

```
<keyword>  Example
:CONDition?  STAT:QUES:LSUM:COND?
:ENABle <bits>  STAT:QUES:LSUM:ENAB <bits>
[:EVENt]?  STAT:QUES:LSUM?
:NTRansition <bits>  STAT:QUES:LSUM:NTR <bits>
:PTRansition <bits>  STAT:QUES:LSUM:PTR <bits>
```
STATus:QUEStionable:LSUMmary:BLIMit <n>:<keyword>

Applicable Models: All

Monitors and summarizes the status of bandwidth limit line failures. When a trace fails, the representative bit is set to 1.

Bit 0 is used to summarize failures in the registers that follow. Refer the STATus:QUEStionable:LSUMmary:BLIMit for the trace number information.

All enable bits are set to 1 by default.

To find the measurement number, use Calc:Par:Mnum

<table>
<thead>
<tr>
<th>&lt;n&gt; Bandwidth Limit register. Choose from 1 to 42.</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;keyword&gt; Example</td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt; STAT:QUES:LSUM:BLIM:ENAB 1024</td>
</tr>
<tr>
<td>[:EVENT]? STAT:QUES:LSUM:BLIM?</td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt; STAT:QUES:LSUM:BLIM:NTR 1024</td>
</tr>
<tr>
<td>:PTRansition &lt;bits&gt; STAT:QUES:LSUM:BLIM:PTR 0</td>
</tr>
</tbody>
</table>

STATus:QUEStionable:LSUMmary:LIMit<n>: <keyword>

Applicable Models: All

Monitors and summarizes the status of limit line failures. When a trace fails, the representative bit is set to 1.

Bit 0 is used to summarize failures in the registers that follow. For example, Limit Register 3, bit 0, summarizes the failures from registers 4 through 42.

All enable bits are set to 1 by default.

To find the measurement number, use Calc:Par:Mnum

<table>
<thead>
<tr>
<th>&lt;n&gt; Limit register. Choose from 1 to 42.</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;keyword&gt; Example</td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt; STAT:QUES:LSUM:LIM1:ENAB 1024</td>
</tr>
<tr>
<td>[:EVENT]? STAT:QUES:LSUM:LIM3?</td>
</tr>
</tbody>
</table>
## Limit Register <n>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>...</th>
<th>41</th>
<th>42</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2-42</td>
<td>3-42</td>
<td>4-42</td>
<td>5-42</td>
<td>6-42</td>
<td>7-42</td>
<td>8-42</td>
<td>9-42</td>
<td>...</td>
<td>42</td>
<td>--</td>
<td>Summary Bit - If any bit from that register fails, it propagates to the previous register, bit 0.</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>15</td>
<td>29</td>
<td>43</td>
<td>57</td>
<td>71</td>
<td>85</td>
<td>99</td>
<td>...</td>
<td>561</td>
<td>575</td>
<td>any point on trace fails the limit test</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>2</td>
<td>16</td>
<td>30</td>
<td>44</td>
<td>58</td>
<td>72</td>
<td>86</td>
<td>100</td>
<td>...</td>
<td>562</td>
<td>576</td>
<td>any point on trace fails the limit test</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>3</td>
<td>17</td>
<td>31</td>
<td>45</td>
<td>59</td>
<td>73</td>
<td>87</td>
<td>101</td>
<td>...</td>
<td>563</td>
<td>577</td>
<td>any point on trace fails the limit test</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>4</td>
<td>18</td>
<td>32</td>
<td>46</td>
<td>60</td>
<td>74</td>
<td>88</td>
<td>102</td>
<td>...</td>
<td>564</td>
<td>578</td>
<td>any point on trace fails the limit test</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>5</td>
<td>19</td>
<td>33</td>
<td>47</td>
<td>61</td>
<td>75</td>
<td>89</td>
<td>103</td>
<td>...</td>
<td>565</td>
<td>579</td>
<td>any point on trace fails the limit test</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>6</td>
<td>20</td>
<td>34</td>
<td>48</td>
<td>62</td>
<td>76</td>
<td>90</td>
<td>104</td>
<td>...</td>
<td>566</td>
<td>580</td>
<td>any point on trace fails the limit test</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>7</td>
<td>21</td>
<td>35</td>
<td>49</td>
<td>63</td>
<td>77</td>
<td>91</td>
<td>105</td>
<td>...</td>
<td>567</td>
<td>--</td>
<td>any point on trace fails the limit test</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>8</td>
<td>22</td>
<td>36</td>
<td>50</td>
<td>64</td>
<td>78</td>
<td>92</td>
<td>106</td>
<td>...</td>
<td>568</td>
<td>--</td>
<td>any point on trace fails the limit test</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>9</td>
<td>23</td>
<td>37</td>
<td>51</td>
<td>65</td>
<td>79</td>
<td>93</td>
<td>107</td>
<td>...</td>
<td>569</td>
<td>--</td>
<td>any point on trace fails the limit test</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>10</td>
<td>24</td>
<td>38</td>
<td>52</td>
<td>66</td>
<td>80</td>
<td>94</td>
<td>108</td>
<td>...</td>
<td>570</td>
<td>--</td>
<td>any point on trace fails the limit test</td>
</tr>
</tbody>
</table>
To determine Register, Bit number, and Weight for trace numbers between 113 and 560 (not shown in the above table) use the following calculations.

The limit status for trace numbers higher than 580 can NOT be tracked.

The following example calculates the Register, Bit number, and Bit Weight for trace # 400:

- To determine Limit Register number, use \(((\text{Trace} \# - 1) / 14) + 1\).
- To determine Limit Bit Number, use the remainder +1 of the above calculation.

\[
\text{calculation: } (400-1)/14 + 1 = \text{Register} + 1 \text{Bit}
\]

\[
399/14 = 28 \text{ r}7
\]

- \(28+1 = \text{Register} 29\)
- \(7+1 = \text{Bit number} 8\)

- To determine Limit Bit Weight: Use above table. For example: Bit 8 = 256

**STATus:QUEStionable:LSUMmary:RLIMit <cnum>:<keyword>**

**Applicable Models:** All

Monitors and summarizes the status of ripple limit line failures. When a trace fails, the representative bit is set to 1.

Bit 0 is used to summarize failures in the registers that follow. Refer the STATus:QUEStionable:LSUMmary:LIIMit for the trace number information.

All enable bits are set to 1 by default.

To find the measurement number, use **Calc:Par:Mnum**
Ripple limit channel status register. Choose from 1 to 42.

Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:QUES:LSUM:RLIM:ENAB 1024</td>
</tr>
<tr>
<td>[:EVENt]?</td>
<td>STAT:QUES:LSUM:RLIM?</td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt;</td>
<td>STAT:QUES:LSUM:RLIM:NTR 1024</td>
</tr>
<tr>
<td>:PTRansition &lt;bits&gt;</td>
<td>STAT:QUES:LSUM:RLIM:PTR 0</td>
</tr>
</tbody>
</table>

Standard Event Status Register

Applicable Models: All

Monitors "standard" events that occur in the analyzer. This register can only be cleared by:

- a Clear Command (*CLS).
- reading the Standard Enable Status Register (*ESE?).
- a power-on transition. The analyzer clears the register and then records any transitions that occur, including setting the Power On bit (7).

<table>
<thead>
<tr>
<th>Commands</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*ESE?</td>
<td>Reads the settings of the standard event <strong>ENABLE</strong> register.</td>
</tr>
</tbody>
</table>
| *ESE <bits> | Sets bits in the standard event **ENABLE** register. The current setting is saved in non-volatile memory.  
              <bits> The sum of weighted bits in the register. Use *ESE 0 to clear the enable register. |
<p>| *ESR?    | Reads and clears the <strong>EVENT</strong> settings in the Standard Event Status register. |
| *OPC     | Sets bit 0 when the overlapped command is complete. (see Understanding Command Synchronization / OPC). |
| *OPC?    | Operation complete query - read the Operation Complete bit (0). |</p>
<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>Operation Complete</td>
<td>the two following events occur <strong>in order:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1. the *OPC command is sent to the analyzer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2. the analyzer completes all pending overlapped commands</td>
</tr>
<tr>
<td>1</td>
<td>NA</td>
<td>Request Control</td>
<td>Not Supported - the analyzer application is not configured to control GPIB operation</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>Query Error</td>
<td>a query error is detected indicating:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- an attempt to read data from the output queue when no data was present <strong>OR</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- data in the output queue was lost, as in an overflow</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>Instrument Dependent Error</td>
<td>Set to &quot;1&quot; when an error has occurred and the error is not a command, query, or execution error.</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>Execution Error</td>
<td>an execution error is detected indicating:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- a <code>&lt;PROGRAM DATA&gt;</code> element was outside the legal range or inconsistent with the operation of the analyzer <strong>OR</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- the analyzer could not execute a valid command due to some internal condition</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>Command Error</td>
<td>a command error is detected indicating that the analyzer received a command that:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• did not follow proper syntax</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• was misspelled</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• was an optional command it does not implement</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>Not used</td>
<td>Always 0</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>Power ON</td>
<td>Power to the analyzer has been turned OFF and then ON since the last time this register was read.</td>
</tr>
</tbody>
</table>
Status Byte Register

**Applicable Models:** All

Summarizes the states of the other registers and monitors the VNA output queue. It also generates service requests. The Enable register is called the Service Request Enable Register.

### Commands

<table>
<thead>
<tr>
<th>Description</th>
<th>Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>*CLS</td>
</tr>
<tr>
<td></td>
<td>*STB?</td>
</tr>
<tr>
<td></td>
<td>*SRE?</td>
</tr>
<tr>
<td></td>
<td>*SRE &lt;num&gt;</td>
</tr>
</tbody>
</table>

*CLS*  Clears ALL "event" registers and the SCPI Error / Event queue. The corresponding ENABLE registers are unaffected.

*STB?*  Reads the value of the analyzer's status byte. The byte remains after being read.

*SRE?*  Reads the current state of the Service Request **Enable** Register.

*SRE <num>*  Sets bits in the Service Request **Enable** register. The current setting of the SRE register is stored in non-volatile memory. Use *SRE 0 to clear the enable.

<num> Combined value of the weights for bits to be set.

### Bit Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>Error / Event queue Summary (EAV)</td>
<td>the Error / Event queue is not empty. To read the error message, use SYST:ERR?</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>Questionable Register Summary</td>
<td>any enabled bit in the <strong>questionable</strong> event status register is set to 1</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>Message Available</td>
<td>the output queue is not empty</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>Standard Event Register Summary</td>
<td>any enabled bit in the <strong>standard</strong> event status register is set to 1</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>Request Service</td>
<td>any of the other bits in the status byte register is set to 1 (used to alert the controller of a service request within the analyzer). This bit cannot be disabled.</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>Operation Register Summary</td>
<td>any enabled bit in the standard <strong>operation</strong> event status register is set to 1</td>
</tr>
</tbody>
</table>
## System Commands

Controls and queries settings that affect the VNA system.

```plaintext
<table>
<thead>
<tr>
<th>SYSTem:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABO Rt:THReshold</td>
</tr>
<tr>
<td>ACTive</td>
</tr>
<tr>
<td>CHANnel</td>
</tr>
<tr>
<td>MARKer</td>
</tr>
<tr>
<td>MCLass</td>
</tr>
<tr>
<td>MEASurement</td>
</tr>
<tr>
<td>NUMBer?</td>
</tr>
<tr>
<td>PORT</td>
</tr>
<tr>
<td>SOURRce?</td>
</tr>
<tr>
<td>TEST?</td>
</tr>
<tr>
<td>SHEet?</td>
</tr>
<tr>
<td>TRACe?</td>
</tr>
<tr>
<td>WINDow</td>
</tr>
<tr>
<td>BEEPer</td>
</tr>
<tr>
<td>COMPLETE:IMMediate</td>
</tr>
<tr>
<td>STATe</td>
</tr>
<tr>
<td>VOLUME</td>
</tr>
<tr>
<td>WARNING:IMMediate</td>
</tr>
<tr>
<td>CAL:ALL More commands</td>
</tr>
<tr>
<td>CAL:PHASe More commands</td>
</tr>
<tr>
<td>CAPability More commands</td>
</tr>
<tr>
<td>CHANnels</td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>COUPle[:STATe]</td>
</tr>
<tr>
<td>COUPle:GROup</td>
</tr>
<tr>
<td>COUPle:PARallel[:ENABle]</td>
</tr>
<tr>
<td>COUPle:PARallel:STATe?</td>
</tr>
<tr>
<td>NOISE:PARallel[:ENABle]</td>
</tr>
<tr>
<td>NOISE:PARallel:GROup</td>
</tr>
<tr>
<td>NOISE:PARallel:GROup:LIST</td>
</tr>
<tr>
<td>NOISE:PARallel:STATe?</td>
</tr>
<tr>
<td>DELete</td>
</tr>
<tr>
<td>HOLD</td>
</tr>
<tr>
<td>RESume</td>
</tr>
<tr>
<td>SINGle</td>
</tr>
<tr>
<td>SINGle:COMBine</td>
</tr>
<tr>
<td>CLOCk[:STATe]</td>
</tr>
<tr>
<td>COMMunicate More commands</td>
</tr>
<tr>
<td>CONFigure</td>
</tr>
<tr>
<td>BIT?</td>
</tr>
</tbody>
</table>
```
Click on a red keyword to view the command details.

See Also
SYSTem:ABORt:THReshold <value>

**Applicable Models:** All

(Read-Write) When a VNA setting is made while a sweep is in progress, the sweep is immediately aborted by default. This command allows you to change that behavior by specifying a time threshold. When a setting change is made during a sweep and if the total sweep time is less than the threshold time, then the sweep is allowed to finish instead of immediately aborting.

In general, VNA setting changes that could cause an aborted sweep are changes that affect how a measurement is made, such as changes in stimulus conditions.

For example, with a threshold setting of 60 seconds:

- Sweeps that require 60 seconds or less from start to finish will be allowed to complete if a VNA setting change is made at any time during the sweep.
- Sweeps that require MORE than 60 seconds from start to finish will be immediately aborted when a VNA setting change is made at any time during the sweep.

**Notes:**

- Preset clears this setting.
- Save state saves this setting.
- Sweep times are estimated.
- This setting affects ALL channels.

**Parameters**

<value> Threshold time in seconds. Set to 0 to immediately abort a sweep when a VNA setting is made.
When a setting is made during a sweep, if that sweep requires less than 10 seconds more to complete, it will be allowed to finish instead of aborting.

Example: SYST:ABOR:THR 10

Query Syntax: SYSTem:ABORt:THReshold?

Default: 0 - No threshold time; all sweeps are immediately aborted.

SYSTem:ACTive:CHANnel?

Applicable Models: All

(Read-only) Returns the number of the active channel. The active channel is the channel number that contains the active measurement. The active measurement is the trace that has a highlighted Tr# in the Trace Status area.

If there is no active channel, 0 is returned.

Example: SYST:PRES SYST:ACT:CHAN?

'Returns 1

Return Type: Integer

Default: Not Applicable

SYSTem:ACTive:MARKer <mkr>

Applicable Models: All

(Read-Write) Sets and reads the active marker

Parameters

<mkr> Active marker.

Example: SYST:ACT:MARK 1

Return Type: Integer

Query Syntax: SYSTem:ACTive:MARKer?

Default: 1

SYSTem:ACTive:MCLass <string>
Applicable Models: All

(Read-Write) Sets and reads the active measurement class.

**Parameters**

- `<string>`: Active measurement class.

**Examples**

```
SYST:ACT:MCL "Standard"
```

**Return Type**

String

**Query Syntax**

SYSTem:ACTive:MCLass?

**Default**

Standard

---

SYSTem:ACTive:MEASurement?

**Applicable Models:** All

(Read-only) Returns the name of the active measurement. While looking at the VNA display, the active measurement is the trace that has a highlighted `Tr#` in the Trace Status area. Only displayed measurements can be active.

If there is no active measurement, " " (empty string) is returned.

**Examples**

```
SYST:PRES
SYST:ACT:MEAS?

'Returns "CH1_S11_1"
```

**Return Type**

String

**Default**

Not Applicable

---

SYSTem:ACTive:MEASurement:NUMBer? <mnum>
**Applicable Models:** All

*(Read-only)* Returns the active measurement number.

**Parameters**

- `<mnum>` Measurement number for each measurement. There must be a selected measurement on the trace. If unspecified, `<mnum>` is set to 1.

**Examples**

```
SYST:PRES
SYST:ACT:MEAS:NUMB?
```

**Return Type** Integer

**Default** 1

---

**SYSTem:ACTive:PORT <num>**

**Applicable Models:** All

*(Read-Write)* Sets or gets the active port number.

**Parameters**

- `<num>` The active port number.

**Examples**

```
SYST:ACT:PORT 1
```

**Return Type** Integer

**Query Syntax** SYSTem:ACTive:PORT?

**Default** 1

---

**SYSTem:ACTive:PORT:SOURce?**

**Applicable Models:** All

*(Read-only)* Gets the active source port number.

**Parameters**

- `<num>` The active port number.

**Examples**

```
SYST:ACT:PORT:SOURce?
```

**Return Type** Integer

**Default** 1

---

**SYSTem:ACTive:PORT:TEST?**
### SYStem:ACTive:PORT:TEST?

**Applicable Models:** All

*(Read-only)* Gets the active test port number.

**Parameters**

**Examples** SYST:ACT:PORT:TEST?

**Return Type** Integer

**Default** 1

### SYStem:ACTive:SHEet?

**Applicable Models:** E5080A

*(Read-only)* Returns the active sheet number.

**Examples** SYST:ACT:SHE?

'Returns "1"

**Return Type** String

**Default** 1

### SYStem:ACTive:TRACe?

**Applicable Models:** All

*(Read-only)* Returns the active trace number.

**Examples** SYST:ACT:TRAC?

'Returns "1"

**Return Type** String

**Default** 1

### SYStem:ACTive:WINDow <wnum>

4897
Applicable Models: All

(Read-Write) Sets or gets the active window number.

Parameters

<wnum> The active window number.

Examples SYST:ACT:WIND 1

Return Type Integer

Query Syntax SYSTem:ACTive:WINDow?

Default 1

SYSTem:BEEPer:COMPLETE:IMMediate

Applicable Models: All

(Write-only) This command generates a beep for the notification of the completion of an operation.

Parameters None

Examples SYST:BEEP:COMP:IMM

Query Syntax Not Applicable

Default Not Applicable

SYSTem:BEEPer:STATe <num>

Applicable Models: All

(Read-Write) Sets the beeper on or off.

Parameters

<bool> ON (1) or OFF (0).

Examples SYST:BEEP:STAT 1

Query Syntax SYSTem:BEEPer:STAT?

Default 0

SYSTem:BEEPer:VOLUME <num>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, M9485A, P937xA

*(Read-Write)* Sets and reads the volume of the internal speaker.

**Parameters**

- `<num>` Relative volume of the internal speaker.
  - Choose a volume between 0 (off) and 100.

**Examples**

- `SYST:BEEP:VOL 5`
- `system:beeper:volume`

**Query Syntax**

- `SYSTem:BEEPer:VOLume?`

**Default**

- 0

---

**SYSTem:BEEPer:WARNing:IMMediate**

**Applicable Models:** All

*(Write-only)* This command generates a beep for the notification of warning/limit test results.

**Parameters**

- None

**Examples**

- `SYST:BEEP:WARN:IMM`

**Query Syntax**

- Not Applicable

**Default**

- Not Applicable

---

**SYSTem:CHANnels:CATalog?**

**Applicable Models:** All

*(Read-only)* Returns the channel numbers currently in use.

**Examples**

- `SYST:CHAN:CAT?`
- `system:channels:catalog?`
- `'Returns:
  "1,2,3"`

**Return Type**

- String of comma-separated numbers

**Default**

- Not Applicable
**SYSTem:CHANnels:COUPle[:STATe] <bool>**

**Applicable Models:** All

*(Read-Write)* Sets and reads the state of channel coupling. This causes the VNA to emulate Keysight 8720 channel coupling.

When set to ON, all existing S-parameter channels receive the stimulus settings of the active channel. Subsequent changes made to any coupled channel are changed on all coupled channels.

Channels with applications such as SMC, VMC, GCA, Noise, IMD are not affected.

Coupling is primarily aimed at stimulus settings (such as start, stop, points, power) but also applies to many trigger settings and to Cal Set pointers.

**Parameters**

- **<bool>**
  - ON (or 1) Channels are coupled
  - OFF (or 0) Channels are NOT coupled

**Examples**

- `SYST:CHAN:COUP 1`  
- `system:channels:couple:state OFF`

**Query Syntax**

SYSTem:CHANnels:COUPle[:STATe]?

**Default**

OFF

---

**SYSTem:CHANnels:COUPle:GROup <iarray>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, M9485A, P937xA

*(Read-Write)* Sets and reads the group of channels for the mult-DUT parallel measurement.

**Parameters**

- **<array>**
  - `{<number of group>, <start channel No.>, <end channel No.>, ... }
  
  The first item means number or groups.
  
  Next pairs show start/end channel numbers of the group. 1 pair for 1 group.

  Example:

  - `{0}` Global coupling (default setting)
  - `{1, 1,4}` Couples channel 1-4
  - `{2, 1,3, 5,7}` Couples channel 1-3 and 5-7 independently
### SYSTem:CHANnels:COUPle:GROup? <value>

**Applicable Models:** M9485A, M980xA, P500xA, and E5080B

*(Read-Write)* Sets and reads the Multi DUT parallel measurement state. SYST:CHAN:COUP should be also turned on when Multi DUT parallel measurement is performed.

**Parameters**

- `<value>`
  - **ON** (or `1`) Multi DUT parallel measurement are enabled
  - **OFF** (or `0`) Multi DUT parallel measurement are NOT enabled

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:CHAN:COUP:GRO 1</td>
<td>Multi DUT parallel measurement enabled</td>
</tr>
<tr>
<td>system:channels:couple:group 1 1 4</td>
<td>Multi DUT parallel measurement off</td>
</tr>
</tbody>
</table>

**Query Syntax**

SYStem:CHANnels:COUPle:GROup?

**Default**

OFF

---

### SYSTem:CHANnels:COUPle:PARallel[:ENABle] <bool>

**Applicable Models:** M9485A, M980xA, P500xA, and E5080B

*(Read-Write)* Sets and reads the Multi DUT parallel measurement state. SYST:CHAN:COUP should be also turned on when Multi DUT parallel measurement is performed.

**Parameters**

- `<bool>`
  - **ON** (or `1`) Multi DUT parallel measurement are enabled
  - **OFF** (or `0`) Multi DUT parallel measurement are NOT enabled

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:CHAN:COUP:PAR 1</td>
<td>Multi DUT parallel measurement enabled</td>
</tr>
<tr>
<td>system:channels:couple:parallel OFF</td>
<td>Multi DUT parallel measurement off</td>
</tr>
</tbody>
</table>

**Query Syntax**

SYStem:CHANnels:COUPle:PARallel[:ENABle]?

**Default**

OFF

---

### SYSTem:CHANnels:NOISe:PARallel:STATe? <value>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, M9485A, P937xA

*(Read Only)* Gets the information if the parallel measurement is executed in the last sweep, for the targeted channel.

**Parameters**

- `<value>`
  - Channel number

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:CHAN:COUP:PAR:STAT? 1</td>
<td>Parallel measurement state enabled</td>
</tr>
<tr>
<td>system:channels:couple:parallel:state? 2</td>
<td>Parallel measurement state off</td>
</tr>
</tbody>
</table>

**Query Syntax**

SYStem:CHANnels:COUPle:PARallel:STATe?

**Default**

Not Applicable

---

### SYSTem:CHANnels:NOISe:PARallel[:ENABle] <bool>


Applicable Models: M9485A

(Read-Write) Sets and reads the Noise Figure Dual-band Parallel Measurement state. SYST:CHAN:Coup should be also turned on when Noise Figure Dual-band Parallel Measurement is performed. This function is supported only for M9385A which has options both 028 and 720.

Parameters
<bool> ON (or 1) Noise Figure Dual-band parallel measurement are enabled
OFF (or 0) Noise Figure Dual-band parallel measurement are NOT enabled

Examples
SYST:CHAN:NOIS:PAR 1
system:channels:noise:parallel OFF

Query Syntax
SYSTem:CHANnels:NOISe:PARallel[:ENABle]?

Default OFF

SYSTem:CHANnels:NOISe:PARallel:GROup <iarray>

Applicable Models: M9485A

(Read-Write) Sets and reads the group of channels for the Noise Figure Dual-band Parallel Measurement. This function is supported only for M9385A which has options both 028 and 720.

Parameters
<iarray> <num of groups>, <first channel of group #1>, <second channel of group #1>,<first channel of group #2>, <second channel of group #2>, … ,<first channel of group #n>, <second channel of group #n>

Two consecutive channels can be assigned into one group.

Example:

{1, 1,2} measures NF channels 1 and 2 in parallel.

{2, 3,4, 8,9} measures NF channels 3 and 4, then 8 and 9 in parallel.

Examples
SYST:CHAN:NOIS:PAR:GRO 1,1,2
system:channels:noise:parallel:group 1,1,2

Query Syntax
SYSTem:CHANnels:NOISe:PARallel:GROup?

Default 0
**SYSTem:CHANnels:NOISe:PARallel:GROup:LIST** <iarray>

**Applicable Models:** M9485A

*(Read-Write)* Sets and reads the group of channels for the Noise Figure Dual-band Parallel Measurement. This command allows you to setup any two channels in one group. This function is supported only for M9385A which has options both 028 and 720.

**Parameters**

| <iarray> |  <num of groups>,   |
|         |  <num of channels in group #1>, <channel>, <channel>,... |
|         |  <num of channels in group #2>, <channel>, <channel>,... |

Several channels can be assigned to one group

**Example:**

{1, 2, 1, 3} measures NF channels 1 and 3 in parallel.
{2, 2, 1, 3, 5, 7} measures NF channels 1 and 3, then 5 and 7 in parallel.

| Examples | **SYST:CHAN:NOIS:GRO:PAR:LIST** 2, 3, 1, 3, 2, 5, 7 |
|          | **system:channels:noise:parallel:group:list** 2, 2, 1, 3, 2, 5, 7 |

| Query Syntax | SYSTem:CHANnels:NOISe:PARallel:GROup:LIST? |
| Default | 0 |

**SYSTem:CHANnels:NOISe:PARallel:STATe?** <value>

**Applicable Models:** M9485A

*(Read Only)* Gets the information if the parallel measurement is executed in the last sweep, for the targeted channel. This function is supported only for M9385A which has options both 028 and 720.

**Parameters**

| <value> | Channel number |

| Examples | **SYST:CHAN:NOIS:PAR:STAT?** 1 |
|          | **system:channels:noise:parallel:state?** 2 |

| Query Syntax | SYSTem:CHANnels:NOISe:PARallel:STATe? |
| Default | Not Applicable |

**SYSTem:CHANnels:DELete** <value>
Applicable Models: All

(Write-only) Deletes the specified channel.

Parameters

<value> Channel number to delete

Examples

SYST:CHAN:DEL 2

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CHANnels:HOLD

Applicable Models: All

(Write-only) Places all channels in hold mode. To place a single channel in hold mode, use SENS:SWE:MODE.

Examples

SYST:CHAN:HOLD

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CHANnels:RESume

Applicable Models: All

(Write-only) Resumes the trigger mode of all channels that was in effect before sending SYSTem:CHANnels:HOLD (must be sent before SYST:CHAN:RESume).

Examples

SYST:CHAN:RES

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CHANnels:SINGle <chanNums>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

(Write-only) Sets up multiple channels for manual trigger and provides a method of triggering multiple channels using manual trigger. There are several prerequisites for using this command:

- Manual trigger mode must be used (INIT:CONT OFF)
- All channels in HOLD (SENS:SWE:MODE HOLD), not just the channels being triggered
- No acquisitions currently running (instrument is NOT sweeping - see ABORt:THReshold )
- All specified channels must exist

If the above conditions are not met, then the command will generate an error which describes what is at fault. If the above conditions are met, then the trigger count for the specified channels is set to 1. Issuing an *OPC? query will indicate when the first channel is armed (ready for manual trigger). It is not necessary to wait for *OPC? before sending INIT:IMM to trigger the first channel. After the first channel is triggered, *OPC? will indicate when all armed channels have finished acquiring data.

Channels will be sorted by channel number, and acquire in order from lowest channel number first to highest channel number last.

**Parameters**

<chanNums> Existing comma separated list of channel numbers.

**Examples**

```
SYST:CHAN:SING 1,3,4
System:channels:single 1,3,4
```

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**SYSTem:CHANnels:SINGle:COMBine <chanNums>**

**Applicable Models:** M937xA, P937xA

(Write-only) Sets the trigger count on the list of channels to ONE, and then combines the channels into a single efficient acquisition. The index line stays high during the entire acquisition.

**Parameters**

<chanNums> Existing comma separated list of channel numbers to combine.

**Examples**

```
SYST:CHAN:SING:COMB 1,3,4
System:channels:combination 1,3,4
```

**Query Syntax** Not Applicable

**Default** Not Applicable
SYSTem:CLOCk[:STATe] <bool>

Applicable Models: All

(Read-Write) Sets and reads the clock visibility state in the VNA status bar.

Parameters

<bool>  ON (or 1)  Clock is visible in the VNA status bar.

OFF (or 0)  Clock is NOT visible in the VNA status bar.

Examples

SYST:CLOC 1
system:clock:state OFF

Query Syntax  SYSTem:CLOCk[:STATe]?

Default  ON

SYSTem:CONFigure <model>,<address>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Write-only) Restarts as an "N-port" VNA using the specified multiport test set.

Learn more about VNA Multiport capability.

See other commands to configure multiport test sets.

Parameters

<model>  String - Model of the test set with which to restart.

Use "Native" to restart without a test set.

To see a list of supported test sets, use SENS:MULT:CAT?

<address>  Numeric - GPIB Address of the test set. Ignored when model = "Native".

Examples

SYST:CONF "NATIVE",0
system:configure "N44xx",18

Query Syntax  Not Applicable

Default  Not Applicable

SYSTem:CONFigure:BIT?
**Applicable Models:** All

*(Read-only)* Returns whether VNA FW is 32 bit application or 64 bit application. Returns the word size (32 or 64).

- **Parameters** None
- **Examples** `SYST:CONF:BIT?`
- **Return Type** String
- **Default** Not Applicable

---

**SYSTem:CONFigure:DIRectory? <char>**

**Applicable Models:** All

*(Read-only)* Returns the directory path location for the specified file type.

- **Parameters** None
- `<char>` Type of file. Choose from:
  - **STATe** - This is the location for the storage of state files.
  - **APPLication** - This is the location of the VNA firmware executable files.
  - **SUPPort** - This is the location of private support files for the VNA firmware.

  See these file locations.
- **Example** `SYST:CONF:DIR? SUPP`
- **Return Type** String
- **Default** Not applicable

---

**SYSTem:CONFigure:REVision:CPU?**
Applicable Models: N522xB, N523xB, N524xB, M937xA, E5080A, P937xA

(Read-only) Returns a number that corresponds to the VNA CPU speed that is visible in the Help, About Network Analyzer dialog box. Learn more.

Use the following table to learn the clock speed using the returned value.

<table>
<thead>
<tr>
<th>Reported CPU version</th>
<th>Clock speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>266 MHz (PNA), 2.53 GHz dual core (E5080A)</td>
</tr>
<tr>
<td>2.0</td>
<td>500 MHz</td>
</tr>
<tr>
<td>3.0</td>
<td>1100 MHz</td>
</tr>
<tr>
<td>4.0</td>
<td>1600 MHz</td>
</tr>
<tr>
<td>5.0</td>
<td>2000 MHz</td>
</tr>
<tr>
<td>6.0</td>
<td>2000 MHz dual core</td>
</tr>
<tr>
<td>7.0</td>
<td>2200 MHz dual core</td>
</tr>
<tr>
<td>8.0</td>
<td>1600 MHz quad core (Max Turbo Frequency 2.90 GHz)</td>
</tr>
</tbody>
</table>

**Parameters**  None

**Example**  SYST:CONF:REV:CPU?

**Return Type**  String

**Default**  Not applicable

---

SYSTem:CONFigure:REVision:DSP?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Returns the DSP Revision number that is visible in the Help, About Network Analyzer dialog box. Learn more.

**Parameters**  None

**Example**  SYST:CONF:REV:DSP?

**Return Type**  String

**Default**  Not applicable
SYSTem:CONFigure:REVision:DSPFpga?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Returns the DSP FPGA Revision number that is visible in the Help, About Network Analyzer dialog box. Learn more.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example</td>
<td>SYST:CONF:REV:DSPF?</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

SYSTem:CONFigure:REVision:PNA:SYNThesizer[0,1,2,3]:VERSion?

Applicable Models: N522xB, N524xB

(Read-only) Returns the Synthesizer Revision number.

Synthesizer Versions

5.0 - Direct Digital Synthesizer to generate frequencies from 10 to 250 MHz, one synthesizer per assembly.

6.0 - Improved filtering and reliability improvements, one synthesizer per assembly.

7.0 - Direct Digital Synthesizer to generate frequencies from 10 MHz to 6 GHz, two or four synthesizers per assembly.

Index Definitions for Synthesizer Versions

0 - LO synthesizer

1 - Src1 synthesizer

2 - Src 2 synthesizer

3 - Src 3 synthesizer

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example</td>
<td>SYST:CONF:REV:PNA:SYNT:VERS?</td>
</tr>
<tr>
<td></td>
<td>&quot;6.0&quot;</td>
</tr>
</tbody>
</table>
Return Type: String
Default: Not applicable

SYSTem:CORRection:WiZard[:IMMediate] <char>

Applicable Models: All

(Write-only) Launches either the Calibration Wizard or the Version 2 Calibration Kit File Manager dialog box.

Remote operation returns immediately after the dialog is launched. This is done to avoid timeout issues with I/O protocols such as VISA. Although it is possible to send commands to the VNA while the dialog is open, this is not encouraged. Application programs should wait until the dialog is closed before resuming remote operations.

Parameters

<char> Choose from:

- **MAIN** - Launches the Calibration Wizard which matches the current channel, such as standard S-params, NoiseFigure, GCA, and so forth.
- **CKIT** - Launches the Version 2 Calibration Kit File Manager dialog box.
- **RESP** - Launches the Response Cal Type Selection.
- **BASic** - Launches the Basic Cal dialog.
- **CALL** - Launches the Calibrate All Selected Channels dialog.

These display on the VNA screen.

Examples

SYST:CORR:WIZ MAIN
system:correction:wizard:immediate ckit

Query Syntax: Not Applicable

Default: MAIN

SYSTem:DATA:MEMory:ADD <string>

Applicable Models: All

(Write only) Add a request to copy the contents of the specified measurement into shared memory. Call this command once for each measurement definition. Once the shared memory is setup, there is no need to send more SCPI commands. The shared memory is filled automatically on every sweep.
Parameters

<string> The following elements separated with colons (:).

<Ch>:<MeasNum>:<dataFormat>:<numPoints>:[<repeatCount>]

where:

- <Ch> - Channel number of the measurement to share memory. Use SYST:ACTive:CHAN to return the active channel number. Use SYST:CHAN:CAT? to return the channel numbers in use.
- <MeasNum> - Measurement (Tr) number to share memory. Use CALC<ch>:PAR:MNUM ? just after the trace is created to read the measurement number. See also: Referring to Traces, Measurements, Channels, and Windows Using SCPI.
- <dataFormat> - Choose from:
  - SDATA - Complex measurement data.
    - Reads data from Apply Error Terms (access point 1). Returns TWO numbers per data point. Corrected data is returned when correction is ON. Uncorrected data is returned when correction is OFF.
  - FDATA - Formatted measurement data to or from Data Access Map location Display (access point 2).
    - Corrected data is returned when correction is ON.
    - Uncorrected data is returned when correction is OFF.
    - Returns one number per data point for all other formats.
    - Format of the read data is same as the displayed format.
  - CAL - Access the entire set of calibrated SParameter measurements.
    - Only functional if an SParameter calibration is applied to the channel.
    - Always returns all the SParameters.
    - SParameters are in the following order for a 3 port cal:
      - \{S11, S21, S31, S12, S22, S32, S13, S23, S33\}
- <numPoints> - Number of data points in the measurement trace.
- [<repeatCount>] - This argument is optional. When specified, the buffer will include multiple sweeps of information into the buffer. Each sweep is stored consecutively into the shared data buffer.
### System:Data:Memory:ADD String "2:3:SDATA:201"

Copies the data for channel #2, measurement #3, complex data, 201 points into the shared memory buffer.

### Query Syntax
Not Applicable

### Default
Not Applicable

---

**System:Data:Memory:Catalog?**

**Applicable Models:** All

(Read only) Returns a list of all the allocated shared memory buffers

**Parameters** None

**Examples** System:Data:Mem:Cat?

**Return Type** String

**Default** Not Applicable

---

**System:Data:Memory:Close <memName>**

**Applicable Models:** All

(Write-only) Closes the specified memory mapped buffer.

**Parameters**

<memName> String. Name of the memory mapped buffer.

**Examples** System:Data:Mem:Clos "VNA_MemoryMap"

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**System:Data:Memory:Close:File <fileName>**
Applicable Models: All

(Write-only) Closes the specified memory file.

Parameters
<fileName> Memory file name.

Examples
SYST:DATA:MEM:CLOS "c:\temp\myfile.dat"

Query Syntax
Not Applicable
Default
Not Applicable

SYSTem:DATA:MEMory:COMMit <memName>

Applicable Models: All

(Write only) Allocates the memory mapped buffer.

Parameters
<memName> String. Name of the memory mapped buffer. This must be a unique name, and cannot conflict with other shared memory buffer names. Use this command in your program when connecting to the shared memory.

Examples
SYST:DATA:MEM:COMM "VNA_MemoryMap"

Query Syntax
Not Applicable
Default
Not Applicable

SYSTem:DATA:MEMory:COMMit:FILE <fileName>
Applicable Models: All

(Write only) This command creates the memory buffer and saves the buffer to a file. This command is an alternative to using the SYST:DATA:MEM:COMMIT command.

Parameters
<fileName> String. Name of the memory mapped buffer file. This must be a unique name, and cannot conflict with other shared memory buffer file names. Use this command in your program when connecting to the shared memory.

Examples
SYST:DATA:MEM:COM:FILE "c:\temp\myfile.dat"

Query Syntax Not Applicable
Default Not Applicable

SYSTem:DATA:MEMory:DELete <memName> - Obsolete

Applicable Models: All

(Write only) Deletes the specified memory mapped buffer.

Parameters
<memName> String. Name of the memory mapped buffer. This is the unique name that is used in the COMMit command.

Examples
SYST:DATA:MEM:DEL "VNA_MemoryMap"

Query Syntax Not Applicable
Default Not Applicable

SYSTem:DATA:MEMory:INITialize

Applicable Models: All

(Write only) Initializes the shared memory setup buffers.

Parameters None

Examples SYST:DATA:MEM:INIT

Query Syntax Not Applicable
Default Not Applicable
SYSTem:DATA:MEMory:NAME?

Applicable Models: All

(Read only) Returns a unique, auto-generated name that can be used in the COMMIt command. By using this generated name, a client can be sure not to conflict with any other used shared memory regions.

| Parameters | None |
| Return Type | String |
| Default | Not Applicable |

SYSTem:DATA:MEMory:OFFSet?

Applicable Models: All

(Read only) The shared memory is a contiguous block of memory. Each measurement takes up a subset of this contiguous block. This command returns the offset (in bytes) into the shared memory for the most recently added parameter. The offset is a number that specifies the starting index (in bytes) of the data.

This query can be sent after sending SYST:DATA:MEM:ADD.

| Parameters | None |
| Return Type | Numeric |
| Default | Not Applicable |

SYSTem:DATA:MEMory:REPeat:RESet

Applicable Models: All

(Write only) This command resets the current repeat index to 0.

| Parameters | None |
| Default | Not Applicable |
SYSTem:DATA:MEMory:RESet

Applicable Models: All

(Write only) Deletes all allocated shared memory buffers.

Parameters
None

Examples
SYS:DATA:MEM:RES

Query Syntax
Not Applicable

Default
Not Applicable

SYSTem:DATA:MEMory:SIZE?

Applicable Models: All

(Read only) Returns the size of the memory mapped region. Send this immediately after SYST:DATA:MEM:COMMit. The result is the total size (in bytes) of all the measurements in the shared memory region.

Parameters
None

Examples
SYS:DATA:MEM:SIZE?

ReturnType
Numeric

Default
Not Applicable

SYSTem:DATE?

Applicable Models: All

(Read-only) Returns the system date.

Parameters
None

Example
SYS:DATE?

ReturnType
Comma separated numbers representing year, month, day.

Default
Not applicable

SYSTem:DISK:REVision?
Applicable Models: N522xB, N523xB, N524xB, P937xA, E5080A

(Read-only) Returns the disk drive version. The format is S.XX.YY.ZZ.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example</td>
<td>SYST:DISK:REV?</td>
</tr>
<tr>
<td>Return Type</td>
<td>Comma separated numbers representing year, month, day.</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

SYSTem:ERRor?

Applicable Models: All

(Read-only) Returns the next error in the error queue. Each time the analyzer detects an error, it places a message in the error queue. When the SYSTEM:ERROR? query is sent, one message is moved from the error queue to the output queue so it can be read by the controller. Error messages are delivered to the output queue in the order they were received. The error queue is cleared when any of the following conditions occur:

- When the analyzer is switched ON.
- When the *CLS command is sent to the analyzer.
- When all of the errors are read.

If the error queue overflows, the last error is replaced with a "Queue Overflow" error. The oldest errors remain in the queue and the most recent error is discarded.

See list of all SCPI Errors.

<table>
<thead>
<tr>
<th>Examples</th>
<th>SYST:ERR? system:err?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

SYSTem:ERRor:COUNt?
Applicable Models: All

(Read-only) Returns the number of errors in the error queue. Use SYST:ERR? to read an error.

See list of all SCPI Errors.

Examples

- SYST:ERR:COUN?
- system:err:count?

Default Not Applicable

SYSTem:ERRor:REPort:SUNLeveled <bool>

Applicable Models: All

(Read-Write) Specifies whether or not to report Source Unleveled errors to the SCPI system error buffer.

This setting will NOT revert to the default (OFF) setting on Instrument Preset. Use the SYSTem:PREFerences:DEFault command to reset the preferences to their default settings.

Parameters

- <bool> ON (or 1) Report Source Unleveled Errors. Read errors from the system error buffer using SYST:ERR?
- OFF (or 0) Do NOT report Source Unleveled Errors.

Examples

- SYST:ERR:REP:SUNL 1
- system:err:report:sunleveled ON

Query Syntax

SYSTem:ERRor:REPort:UNLeveled?

Default OFF

SYSTem:FCORrection:CHANnel<cnum>:COUPler[:STATe] <char>
Applicable Models: M937xA, P937xA, E5080B, M980xA, P50xxA, M9485A

(Read-Write) (M937xA/P937xA) Sets and returns the coupler state. This command is not effective for SMC class. (E5080B, M980xA, P50xxA, M9485A) Turn off the system calibration.

Parameters

<char> Choose from:

OFF

AUTO

Examples

SYST:FCOR:CHAN:COUP AUTO
system:fcorrection:channel1:coupler OFF

Query Syntax

SYSTem:FCORrection:CHANnel<cnum>:COUPler[:STATe]?

Return Type

Character

Default

AUTO (for ENA/PXI)

OFF (for PNA)

SYSTem:FPReset

Applicable Models: All

(Write-only) Performs a standard Preset, then deletes the default trace, measurement, and window. The VNA screen becomes blank.

Examples

SYST:FPR
system:fpreset

Default

Not applicable

SYSTem:ISPContol[:STATe] <bool>
**Applicable Models:** E5080A/B, M9485A, M980xA, P50xxA

*(Read-Write)* Sets and reads the status of the Initial Source Port Control feature (to switch the stimulus output in the trigger hold state to a test port).

**Parameters**

- `<bool>`
  - **ON** (or 1) Source is outputted only when measurement is done. Source is not outputted during hold state.
  - **OFF** (or 0) Source is always outputted.

**Examples**

```
SYST:ISPC 1
system:ispcontrol OFF
```

**Query Syntax**

SYSTem:ISPControl[:STATe]?

**Default**

ON (E5080A), OFF (P50xxA, M980xA, M9485A, E5080B)

---

**SYSTem:LOCal**

**Applicable Models:** All

*(Write-only)* Sets the local/remote state to local.

**Examples**

```
SYST:LOC
system:local
```

**Default**

Not applicable

---

**SYSTem:MACRo:COPY:CHANnel<cnum>[:TO] <num>**

**Applicable Models:** All

*(Write-only)* Copies ALL settings from `<cnum>` channel to `<num>` channel. Learn more about copy channels.

Use SENS:PATH:CONF:COPY to copy ONLY mechanical switch and attenuator settings.

**Parameters**

- `<cnum>` Channel number to copy settings from. If unspecified, value is set to 1.
- `<num>` Channel number to copy settings to.

**Examples**

```
SYST:MACR:COPY:CHAN1 2
system:macro:copy:channel2:to 3
```

**Query Syntax**

Not Applicable
SYSTem:MACRo:COPIy:CHANnel<fromChannel>:STATe <toChannel>,<toWindow>,[<copyScope>]

**Applicable Models:** All

*(Write-only)* Copies settings only, or settings and measurements, traces, markers, and limit lines from an existing channel, `<fromChannel>`, to a new channel, `<toChannel>`. Traces can be copied into the Active Window, a user specified window, or a new (next available) window.

**Parameters**
- `<fromChannel>`: Channel number to copy settings from. If unspecified, value is 1.
- `<toChannel>`: 0 for next available channel, or N for channel number to copy settings to.
- `<toWindow>`: -1 will create a new window, 0 will use the active window, and N will use the specified window N. `<toWindow>` is ignored when `<copyScope>` is "stimulus"
- `<copyScope>`: must be "stimulus" which copies only settings, or "state" which copies settings, measurements, traces, markers, and limit lines.

**Examples**

```
SYST:MACR:COPY:CHAN1:STAT 2,0,"stimulus"
```

Copies only settings from channel #1 to channel #2. This is equivalent to `SYST:MACR:COPY:CHAN1 2`

```
SYST:MACR:COPY:CHAN1:STAT 2,-1,"state"
```

Copies settings, measurements, traces, etc. from channel #1 to channel #2. Traces are placed into a new window (next available window), and additional windows will be created as necessary so that all traces are copied.

```
SYST:MACR:COPY:CHAN1:STAT 0,-1
```

Copies settings, measurements, etc. from channel #1 to the next available channel and places traces into the next available new window.

**Query Syntax** Not Applicable

**Default** Not Applicable
SYSTem:MACR: COPY:CHAN<fromChan>:SOUR <fromPort>,<toChan>,<toPort>

**Applicable Models:** N522xB, N523xB, N524xB, E5080A, M9485A, P937xA

*(Write-only)* Copies and applies an existing Source Power Calibration to another channel. Learn more about source power calibration.

**Parameters**
- `<fromChan>` Channel number of the existing source power correction.
- `<fromPort>` Port number of the existing source power correction.
- `<toChan>` Channel number to which the source power correction will be copied.
- `<toPort>` Port number to which the source power correction will be applied.

**Examples**
- `SYST:MACR: COPY:CHAN1:SOUR 1,2,1`
- `system:macro:copy:channel2:source 2,1,2`

**Query Syntax** Not Applicable

**Default** Not Applicable

---

SYSTem:MClass:CATalog?

**Applicable Models:** All

*(Read-only)* Returns measurement classes available on the VNA. Learn more about Measurement Classes.

**Parameters** None

**Examples**
- `SYST:MCClass:CAT?`

**Return Type** String of comma-separated measurement class names. See the complete list of measurement class names.

**Default** Not Applicable

---

SYSTem:MCClass:PARameter:CATalog? <name>
### SYSTem:MEASure:CATalog? [chan]

**Applicable Models:** All

(Read-only) Returns ALL measurement numbers, or measurement numbers from a specified channel.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[chan]</td>
<td>Optional. Channel number to catalog. If not specified, all measurement numbers are returned.</td>
</tr>
</tbody>
</table>

#### Examples

- Returns all measurement numbers
  
  ```
  SYST:MEAS:CAT? 
  ```

- Returns the measurement numbers on channel 2
  
  ```
  system:measurement:catalog? 2 
  ```

<table>
<thead>
<tr>
<th>Return Type</th>
<th>String of comma-separated numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

---

### SYSTem:MEASure<n>:NAME?

**Applicable Models:** All

(Read-only) Returns ALL parameters that are supported by the specified measurement class.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;name&gt;</td>
<td>String. Measurement Class name. See the complete list of measurement class names.</td>
</tr>
</tbody>
</table>

#### Examples

- Returns all parameters for Gain Compression.
  
  ```
  SYST:MCL:PAR:CAT? "Gain Compression"
  ```

<table>
<thead>
<tr>
<th>Return Type</th>
<th>String of comma-separated parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
Applicable Models: All

(Read-only) Returns the name of the specified measurement.

Parameters

<n> Measurement number for which to return the measurement name. If unspecified, value is set to 1.

Examples
'SReturns the name of measurement 2
SYST:MEAS2:NAME?

Return Type String
Default Not Applicable

SYSTem:MEASure<n>:TRACe?

Applicable Models: All

(Read-only) Returns the trace number of the specified measurement number. Trace numbers restart for each window while measurement numbers are always unique.

Parameters

<n> Measurement number for which to return the trace number. If unspecified, value is set to 1.

Examples
'SReturns the trace number of measurement 1
SYST:MEAS1:TRAC?

Return Type Numeric
Default Not Applicable

SYSTem:MEASure<n>:WINDow?
Applicable Models: All

(Read-only) Returns the window number of the specified measurement number.

Parameters

<n> Measurement number for which to return the window number. If unspecified, value is set to 1.

Examples

'SReturns the window number of measurement 2
SYST:MEAS2:WIND?

Default Numeric

Return Type Numeric

Default Not Applicable

SYSTem:PERSona:MANufacturer <string>

Applicable Models: All

(Read-Write) This command allows you to modify the manufacturer name returned by the instrument’s *IDN query response. This is intended to be used for Agilent backward identity compatibility. For example, "Agilent Technologies” or “Agilent”. However, it could be used for other purposes such as emulation of another vendor’s instrument.

The change to the manufacturer string will not take effect until after an instrument reboot.

The manufacturer string does not allow commas in the name. If a comma is detected an error is returned. Also, if an invalid manufacturer is detected, an error is returned.

The manufacturer string used for Keysight is “Keysight Technologies”.

Parameters

<string> Name of the manufacturer.

Examples SYST:PERS:MAN "Keysight Technologies"

Query Syntax SYSTem:PERSona:MANufacturer?

Return Type String

Default Not Applicable

SYSTem:PERSona:MANufacturer:DEFault
**Applicable Models:** All

(Read-Write) Sets and returns the instrument's original manufacturer identification state following the next instrument reboot.

| Parameters | None |
| Parameters | None |

**Examples**

| SYST: PERS: MAN: DEF |
| SYST: PERS: MAN: DEF |

**Query Syntax**

SYSTem:PERSona:MANufacturer:DEFault?
SYSTem:PERSona:MANufacturer?

**Return Type**

String

**Default**

Not Applicable

---

**SYSTem:PERSona:MODel <string>**

**Applicable Models:** All

(Read-Write) This command allows you to modify the product model returned by the instrument’s *IDN query response. This is intended to be used for model compatibility. If not specified, the default model of the instrument is used.

The change to the model string will not take effect until after an instrument reboot.

The model string does not allow commas in the name. If a comma is detected an error is returned. Also, if an invalid model is detected, an error is returned.

**Parameters**

<string> Product model name.

**Examples**

| SYST: PERS: MOD "33220A" |
| SYST: PERS: MOD "33220A" |

**Query Syntax**

SYSTem:PERSona:MANufacturer?

**Return Type**

String

**Default**

Not Applicable

---

**SYSTem:PERSona:MODel:DEFault**
Applicable Models: All

(Read-Write) Sets and returns the instrument's original product model name following the next instrument reboot.

**Parameters** None

**Examples** `SYST: PERS: MOD: DEF`

**Query Syntax** SYSTem:PERSona:MODel:DEFault?

**Return Type** String

**Default** Not Applicable

---

SYSTem:POFF

Applicable Models: All

(Write-only) Shuts down the system.

**Parameters**

<n> Shutdown or restart. Choose from:

1 - Restart.

0 - Shutdown.

**Examples** 'Shuts down the system

`SYST: POFF`

**Default** 0 (Shutdown)

---

SYSTem:POWer<pnum>:LIIMit <value>
**Applicable Models:** All

*(Read-Write)* Sets and returns the power limit for the specified port. Learn more about Power Limit.

**Parameters**

- `<pnum>`  Port number. Choose any VNA port.
- `<value>`  Power limit in dBm

**Examples**

```
SYST:POW1:LIM 5
system:power2:limit 0
```

**Query Syntax**  SYSTem:POWer<pnum>:LIMit?

**Return Type**  Numeric

**Default**  100 dBm

---

**SYSTem:POWer:LIMit:LOCK <bool>**

**Applicable Models:** All

*(Read-Write)* Enables or disables the ability to change the power limit values through the user interface. Learn more about Power Limit.

**Parameters**

- `<bool>`  Power limit lock. Choose from:

  - **ON** or **1** - Disables the ability to change the power limit values from the user interface.
  - **OFF** or **0** - Enables the ability to change the power limit values from the user interface.

**Examples**

```
SYST:POW:LIM:LOCK 1
system:power:limit:lock OFF
```

**Query Syntax**  SYSTem:POWer:LIMit:LOCK?

**Return Type**  Boolean

**Default**  OFF

---

**SYSTem:POWer<pnum>:LIMit:STATe <bool>**
**Applicable Models:** All

**Read-Write** Enables or disables the power limit for the specified port. Learn more about Power Limit.

### Parameters

- `<pnum>` Port number. Choose any VNA port.
- `<value>` Power limit state. Choose from:
  - **ON** or **1** Enables the power limit for the port<pnum>.
  - **OFF** or **0** Disables the power limit for the port<pnum>.

### Examples

```
SYST:POW1:LIM:STAT ON
system:power2:limit:state 0
```

### Query Syntax

```
SYSTem:POWer<pnum>:LIMit:STATe?
```

### Return Type

Boolean

### Default

OFF

---

**SYSTem:PRESet**

**Applicable Models:** All

**(Write-only)** Deletes all traces, measurements, and windows. In addition, resets the analyzer to factory defined default settings and creates a S11 measurement named "CH1_S11_1". For a list of default settings, see Preset.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the VNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

If the VNA display is disabled with DISP:ENAB OFF then SYST:PRES will NOT enable the display.

This command performs the same function as *RST with one exception: Syst:Preset does NOT reset Calc:FORMAT to ASCII as does *RST.

### Examples

```
SYST:PRES
system:preset
```

### Default

Not applicable

---

**SYSTem:SECurity[:LEVe]l <char>**
Applicable Models: All

(Read-Write) Sets and returns the display of frequency information on the VNA screen and printouts.

Learn more about security level.

Parameters

<char> Choose from:

NONE - ALL frequency information is displayed.

LOW - NO frequency information is displayed. Frequency information can be redisplayed using the Security Setting dialog box or this command.

HIGH - LOW setting plus GPIB console is disabled. Frequency information can be redisplayed ONLY by performing a Preset, recalling an instrument state with None or Low security settings, or using this command.

EXTRA - HIGH setting plus:

- ASCII data saving is disabled. Same method to redisplay frequency information as HIGH setting.
- Mixer setup files (*.mxr) can NOT be saved.

Examples

SYST:SEC LOW
system:security:level high

Query Syntax

SYSTem:SECurity[:LEVel]?

Return Type

Character

Default

None

SYSTem:SET <block>
Applicable Models: All

(Read-Write) Sends a definite-length binary block Instrument state and sets the VNA with those settings. This command does the same as saving a *.sta file to the VNA (MMEM:STOR STATE ) and then MMEM:TRAN to transfer the file to the computer.

Parameters

- `<block>` The Instrument state file as definite-length arbitrary binary block.

Examples

```
SYST:SET <block>
```

Query Syntax

SYSTem:SET? (This saves the instrument state file to the remote computer.)

Return Type

Definite-length arbitrary binary block.

Default

Not Applicable

SYSTem:SHEets:CATalog?

Applicable Models: N522xB, N523xB, N524xB, E5080A, M937xA, P937xA

(Read-only) Returns comma separated list of visible sheets.

Parameters

Examples

```
SYST:SHE:CAT?

'Returns:

"1,2,3"
```

Return Type

String of comma-separated numbers

Default

1

SYSTem:SHORtcut<n>:ARGuments<string>
Applicable Models: All

(Read-Write) Reads and writes the arguments for the specified macro. On the Edit Macro Dialog, this is called the "Macro run string parameters".

**Parameters**

- `<n>` Numeric. Number of the macro that is stored in the VNA.
  
  To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all of the macros for the desired macro title.

- `<string>` Arguments for the specified macro.

**Examples**

- SYST:SHOR1:ARG

**Query Syntax**

SYSTem:SHORtcut<n>:ARGuments?

**Default** Not Applicable

---

SYSTem:SHORtcut<n>:DELe

Applicable Models: All

(Write-only) Removes the specified macro from the list of macros in the VNA. Does not delete the macro executable file.

**Parameters**

- `<n>` Numeric. Number of the macro that is stored in the VNA.
  
  To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all of the macros for the desired macro title.

**Examples**

- SYST:SHOR1:DEL

**Query Syntax** Not Applicable

**Default** Not Applicable

---

SYSTem:SHORtcut<n>:EXECute
**Applicable Models:** All

*(Write-only)* Executes (runs) the specified Macro (shortcut) that is stored in the VNA.

**Note:** If Enable Remote Drive Access is unchecked in the Remote Interface dialog, this command will return an error.

**Parameters**

<n> Numeric. Number of the macro that is stored in the VNA.

To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all of the macros for the desired macro title.

**Examples**

SYST:SHOR1:EXEC

**Query Syntax**

Not Applicable

**Default**

Not Applicable

**SYSTem:SHORtcut<n>:PATH <string>**

**Applicable Models:** All

*(Read-Write)* Defines a Macro (shortcut) by linking a path and file name to the Macro number. To be executed, the executable file must be put in the VNA at the location indicated by this command.

**Parameters**

<n> Numeric. Number of the macro to be stored in the analyzer. If the index number already exists, the existing macro is replaced with the new macro.

<string> Full path, file name, and extension, of the existing macro "executable" file.

To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all of the macros for the desired macro title.

**Examples**

SYST:SHOR1:PATH "C:\Program Files(x86)\Keysight\Network Analyzer\Documents\unguideMultiple.vbs"

**Query Syntax**

SYSTem:SHORtcut<n>:PATH?

**Default**

Not Applicable

**SYSTem:SHORtcut<n>:TITLe<string>**
Applicable Models: All

(Read-Write) Reads and writes the name of the specified macro.

**Parameters**

- `<n>` Numeric. Number of the macro that is stored in the VNA.
  
  To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all of the macros for the desired macro title.

- `<string>` The name to be assigned to the macro.

**Examples**

```
SYST:SHOR1:TITL "Guided 4-Port Cal"
```

**Query Syntax**

```
SYSTem:SHORcut<n>:TITLe?
```

**Default** Not Applicable

---

**SYSTem:TDR:PRESet**

Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) Launches TDR application without VNA-TDR GUI.

**Parameters** None

**Example**

```
SYST:TDR:PRES
```

**Return Type** Not applicable

**Default** Not applicable

---

**SYSTem:TIME?**

Applicable Models: All

(Read-only) Returns the system time.

**Parameters** None

**Example**

```
SYST:TIME?
```

**Return Type** Comma separated numbers representing hours, minutes, seconds.

**Default** Not applicable

---

**SYSTem:TOUCHscreen[:STATe] <bool>**
**Applicable Models:** N522xB, N523xB, N524xB, E5080A/B

*(Read-Write)* Enables and disables the touchscreen.

This setting remains until changed again from the front-panel or remotely, or until the hard drive is changed or reformatted.

**Parameters**

<bool> Choose from:

**ON (1)** Enables the touchscreen.

**OFF (0)** Disables the touchscreen.

**Examples**

```
SYST:TOUC 1
system:touchscreen:state OFF
```

**Query Syntax**

SYSTem:TOUChscreen[:STATe]?

**Return Type**

Boolean

**Default**

ON when shipped from factory.

---

**SYSTem:UPReset**

**Applicable Models:** N522xB, N523xB, N524xB, E5080A, M9485A, P937xA

*(Write-only)* Performs a User Preset. There must be an active User Preset state file (see Load and Save) or an error will be returned. Learn more about User Preset.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the VNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

**Examples**

```
SYST:UPReset
system:upreset
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**SYSTem:UPReset:FPANel[:STATe] <bool>**
**Applicable Models:** N522xB, N523xB, N524xB, E5080A, M9485A, P937xA

(Read-Write) 'Checks' and 'clears' the enable box on the User Preset dialog box. This only affects subsequent Presets from the front panel user interface.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the VNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

**Parameters**

<bool> Front Panel User Preset State. Choose from:

0 User Preset OFF

1 User Preset ON

**Examples**

```
SYST:UPR:FPAN 1
system:upreset:fpanel:state 0
```

**Query Syntax**

SYSTem:UPREset:FPANel[:STATe]?

**Return Type**

Boolean

**Default**

0

---

**SYSTem:UPReset:LOAD[:FILE] <file>**

**Applicable Models:** N522xB, N523xB, N524xB, E5080A, M9485A, P937xA

(Write-only) Loads an existing instrument state file (.sta or .cst) to be used for User Preset. Subsequent execution of SYSTem:UPReset will cause the VNA to assume this instrument state.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the VNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

Learn more about User Preset.

**Parameters**

<file> String - Name of the file to be loaded. Change the default folder name using MMEMory:CDIRectory.

**Examples**

```
SYST:UPR:LOAD '1MHzto20GHzUserPreset.cst'
system:upreset:load:file 'C:\Documents and Settings\Administrator\My Documents\NewUserPreset.cst'
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable
SYStem:UPReset:SAVE[:STATE]

Applicable Models: N522xB, N523xB, N524xB, E5080A, M9485A, P937xA

(Write-only) Saves the current instrument settings as UserPreset.sta. Subsequent execution of SYStem:UPReset will cause the VNA to assume this instrument state.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the VNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

Learn more about User Preset.

Examples

SYST:UPR:SAVE
system:upreset:save:state

Query Syntax

Not Applicable

Default

Not Applicable

SYStem:WINDows:CATalog?

Applicable Models: All

(Read-only) Returns the window numbers that are currently being used.

Examples

SYST:WIND:CAT?

system:windows:catalog?

Return Type

String of comma-separated numbers.

For example: "1,2"

Default

Not Applicable
**SYSTEM:CALibrate:ALL Commands**

Contains the settings to configure a "Cal All" Calibration.

Use the Guided Cal interface to perform the calibration.

<table>
<thead>
<tr>
<th>SYSTEM:CALibrate:ALL:</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANnel:</td>
</tr>
<tr>
<td>PORTs[:SELect]</td>
</tr>
<tr>
<td>CSET:</td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>PREFix</td>
</tr>
<tr>
<td>GUIDed:</td>
</tr>
<tr>
<td>CHANnel:LIST?</td>
</tr>
<tr>
<td>CHANnel[:VALue]?</td>
</tr>
<tr>
<td>PORTs?</td>
</tr>
<tr>
<td>IFBW</td>
</tr>
<tr>
<td>INDependent:</td>
</tr>
<tr>
<td>SOURce:</td>
</tr>
<tr>
<td>CALibrate:</td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>RANGE:</td>
</tr>
<tr>
<td>ADD</td>
</tr>
<tr>
<td>CLEar</td>
</tr>
<tr>
<td>COUNt</td>
</tr>
<tr>
<td>PONts</td>
</tr>
<tr>
<td>STARt</td>
</tr>
<tr>
<td>STOP</td>
</tr>
<tr>
<td>MCLass:</td>
</tr>
<tr>
<td>PROPerty:</td>
</tr>
<tr>
<td>NAME:CATalog?</td>
</tr>
<tr>
<td>VALue:</td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>[STATe]</td>
</tr>
<tr>
<td>PATH:</td>
</tr>
<tr>
<td>CONFigure:</td>
</tr>
</tbody>
</table>
Click on a red keyword to view the command details.

See Also

- About Calibrate All Channels
- Example Programs
- Guided Cal commands
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**SYSTem:CALibrate:ALL[1-250]:CHANnel<ch>:PORTs[:SELect] <value>**

**Applicable Models:** All

*(Write-Read)* For each channel to be calibrated, sets and returns the ports to be calibrated. Specify port numbers ONLY for standard channels.

**Parameters**

- *[1-250]* Calibration number. The default is 1.
- *<ch>* Channel number to be calibrated.
- *<value>* Ports to be calibrated for the specified channel. Select any of the native VNA ports (1,2,3,4).

**Examples**

```
SYST:CAL:ALL:CHAN2:PORT 1,2,3
```

**Query Syntax**

SYSTem:CALibrate:ALL:CHANnel<ch>:PORTs[:SELect]?

**Return Type**

Comma-separated port numbers.
SYSTem:CALibrate:ALL[1-250]:CSET:CATalog?

Applicable Models: All

(Read-only) Returns the User Cal Set or cal register names that were produced by the cal all session.

Parameters

[1-250] Calibration number. The default is 1.

Examples

SYST:CAL:ALL:CSET:CATalog?

'returns this format:

"MyCalAll_STD_001, MyCalAll_SMC_002"

See example program

Return Type String of comma-separated Cal Set or cal register names

Default Not Applicable

SYSTem:CALibrate:ALL[1-250]:CSET:PREFix<value>

Applicable Models: All

(Write-Read) Sets and returns the prefix to be used when saving User Cal Sets that result from the Cal All session. The Meas Class and channel number are appended to this prefix for each calibrated channel. Use SYST:CAL:ALL:CSET:CATalog? to read the saved cal set names.

- SENS:CORR:COLL:GUID:SAVE:CSET can also be used to set the Cal Set prefix.
- If a Cal Set prefix is NOT set using either command, the cal data for each channel will be saved only to cal registers. Learn about cal registers.

Parameters

[1-250] Calibration number. The default is 1.

<value> (String) User Cal Set prefix.

Examples

SYST:CAL:ALL:CSET:PREFix "MyCalAll"

Query Syntax SYSTem:CALibrate:ALL:CSET:PREFix?

Return Type String

Default " " (Empty string)
**SYSTem:CALibrate:ALL[1-250]:GUIDed:CHANnel:LIST?**

**Applicable Models:** All

(Read-only) Returns all cal all guided calibration channels.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1-250]</td>
<td>Calibration number. The default is 1.</td>
</tr>
</tbody>
</table>

**Examples**

```
chan = SYST:CAL:ALL:GUID:CHAN:LIST?
```

**Return Type**

Numeric of comma-separated channel numbers

**Default**

Not applicable

---

**SYSTem:CALibrate:ALL[1-250]:GUIDed:CHANnel[:VALue]?**

**Applicable Models:** All

(Read-only) Returns the primary guided calibration channel number if more than one channel exists; otherwise, this command returns the one value. Use this value as the <ch> argument for the subsequent Guided:Cal commands.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1-250]</td>
<td>Calibration number. The default is 1.</td>
</tr>
</tbody>
</table>

**Examples**

```
chan = SYST:CAL:ALL:GUID:CHAN:VAL?
```

**Return Type**

Numeric

**Default**

Not applicable

---

**SYSTem:CALibrate:ALL[1-250]:GUIDed:PORTs?**
Applicable Models: All

(Read-only) Returns the ports to be calibrated during the Cal All Channels calibration. Specify connectors and cal kits for these ports using the Guided:Cal commands.

Specify the ports to be calibrated for each channel using SYST:CAL:ALL:CHAN<ch>:PORT.

Parameters

<table>
<thead>
<tr>
<th>[1-250]</th>
<th>Calibration number. The default is 1.</th>
</tr>
</thead>
</table>

Examples

```
ports = SYST:CAL:ALL:GUID:PORT?
```

Return Type
Comma-separated list of port numbers

Default
Not applicable

SYSTem:CALibrate:ALL[1-250]:IFBW <value>

Applicable Models: All

(Write-Read) Sets and returns the IFBW for a Cal All calibration. Learn more about this setting.

Parameters

<table>
<thead>
<tr>
<th>[1-250]</th>
<th>Calibration number. The default is 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;value&gt;</td>
<td>IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the VNA model. See the list of valid settings. If an invalid number is specified, the VNA will round up to the closest valid setting.</td>
</tr>
</tbody>
</table>

Examples

```
SYST:CAL:ALL:IFBW 10e3
```

Query Syntax
SYSTem:CALibrate:ALL:IFBW?

Return Type
Numeric

Default
1 kHz

SYSTem:CALibrate:ALL[1-250]:INDependent:SOURce:CALibrate:CATalog?
Applicable Models: All

(Read-only) Returns available ports for independent power calibration.

**Parameters**

[1-250] Calibration number. The default is 1.

**Examples**

```
SYST:CAL:ALL:IND:SOUR:CAL:CAT?
```

**Return Type** String

**Default** Not applicable

---


Applicable Models: All

(Write-only) This command adds a power cal range for a specific port <n>. Note that external sources are valid and specifying a source port is the same as other remote commands. By default this will create a range with the preset start/stop frequency and 201 points. The maximum number of ranges that can be added is 100 (same as the maximum number of segments).

**Parameters**

[1-250] Calibration number. The default is 1.

<n> Port number

**Examples**

```
```

**Default** Not applicable

---

SYSTem:CALibrate:ALL[1-250]:INDependent:SOURce<n>:CALibrate:RANGe:CLEar

Applicable Models: All

(Write-only) This command resets all ranges for the given source port <n>.

**Parameters**

[1-250] Calibration number. The default is 1.

<n> Port number

**Examples**

```
```

**Default** Not applicable
SYSTem:CALibrate:ALL[1-250]:INDependent:SOURce<n>:CALibrate:RANGe:COUNt?

Applicable Models: All

(Read-only) This command queries how many ranges are included in the calibration for source port <n>.

Parameters

[1-250] Calibration number. The default is 1.
<n> Port number

Examples


Return Type Numeric
Default Not applicable

SYSTem:CALibrate:ALL[1-250]:INDependent:SOURce<n>:CALibrate:RANGe<m>:POINts <value>

Applicable Models: All

(Write-Read) This command sets and gets the number of points for range <m> for source port <n>.

Parameters

[1-250] Calibration number. The default is 1.
<n> Port number
<m> Range number
<value> Number of points

Examples


Query Syntax
SYSTem:CALibrate:ALL:INDependent:SOURce<n>:CALibrate:RANGe<m>:POINts?

Return Type Numeric
Default Not applicable

SYSTem:CALibrate:ALL[1-250]:INDependent:SOURce<n>:CALibrate:RANGe<m>:STARt <value>
Applicable Models: All

(Write-Read) This command sets and gets the start frequency for range <m> for source port<n>.

Parameters

[1-250] Calibration number. The default is 1.

<n> Port number

<m> Range number

<value> Start frequency for range <m>

Examples


Query Syntax

SYSTem:CALibrate:ALL:INDependent:SOURce<n>:CALibrate:RANGe<m>:START?

Return Type

Numeric

Default Not applicable

SYSTem:CALibrate:ALL[1-250]:INDependent:SOURce<n>:CALibrate:RANGe<m>:STOP

<value>

Applicable Models: All

(Write-Read) This command sets and gets the stop frequency for range <m> for source port<n>.

Parameters

[1-250] Calibration number. The default is 1.

<n> Port number

<m> Range number

<value> Stop frequency for range <m>

Examples


Query Syntax

SYSTem:CALibrate:ALL:INDependent:SOURce<n>:CALibrate:RANGe<m>:STOP?

Return Type

Numeric

Default Not applicable

Applicable Models: All

(Read-only) Returns the unique, settable properties for the current cal all session.

See a list of valid properties and values for each measurement class.

**Parameters**

- **[1-250]** Calibration number. The default is 1.
- **[mclass]** Optional argument. String name of the measurement class for which properties are to be returned. See a list of valid measurement class Application names. The measurement class must be included in the current Cal All calibration.

**Examples**


'with NFX app, returns:

"Noise Cal Method, Noise Tuner, AutoOrient Tuner, Tuner In, Tuner Out, Receiver Characterization Method, ENR File, Noise Source Connector, Noise Source CalKit"

**Return Type**

String of comma-separated properties.

**Default**

Not applicable


Applicable Models: All

(Read-only) Returns the valid property values for a specific property name.

See a list of valid properties and values for each measurement class.

**Parameters**

- **[1-250]** Calibration number. The default is 1.
- **<prop>** (String) Property name for which valid values are to be returned.

**Examples**


'with NFX app, returns:

"Scalar, Vector"

**Return Type**

String of comma-separated values

**Default**

Not applicable
SYSTem:CALibrate:ALL[1-250]:MClass:PROPerty:VALue[:STATe] <prop>,<value>

Applicable Models: All

(Write-Read) Sets and returns the property value for a specific property name.

See a list of valid properties and values for each measurement class.

**Parameters**

- **[1-250]** Calibration number. The default is 1.
- **<prop>** (String) Property name for which value is to be set or returned.
- **<value>** Property value. To read a list of valid values, use SYST:CAL:ALL:MCL:PROP:VAL:CAT?

**Examples**

- **Example 1:**

- **Example 2:**
  SYST:CAL:ALL:MCL:PROP:VAL "Enable Extra Power Cals","Port 1 Src2,Port3"

- **Example 3:**
  SYST:CAL:ALL:MCL:PROP:VAL "Port 1 Src2 Cal Power","-20"

**Query Syntax**

SYSTem:CALibrate:ALL:MClass:PROPerty:VALue[:STATe]? <prop>

**Return Type**

String

**Default**

Varies with the property name.

---

SYSTem:CALibrate:ALL[1-250]:PATH:CONFigure:ELEMENT[:STATe] <element>,<setting>
**Applicable Models:** N522xB, N523xB, N524xB

(Write-Read) Sets and returns the Path Configuration settings for a Cal All calibration.

**Parameters**

- **[1-250]** Calibration number. The default is 1.
- **<element>** (String) Path configuration element to be set. See a list of configurable RF Path elements and settings.
- **<setting>** (String) Path configuration element setting.

**Examples**

```plaintext
SYST:CAL:ALL:PATH:CONFigure:ELEMent "Port1NoiseTuner","Internal"
```

**Query Syntax**

```plaintext
SYSTem:CALibrate:ALL:PATH:CONFigure:ELEMent[:STATe]? <element>
```

**Return Type**

String

**Default**

Not Applicable

---

**Applicable Models:** All

(Write-Read) Sets and returns the Receiver Attenuator setting for a Cal All calibration.

**Parameters**

- **[1-250]** Calibration number. The default is 1.
- **<n>** Receiver port number.
- **<value>** Attenuation value in dB for a Cal All calibration. Choose a valid value for the VNA model. See valid settings.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```plaintext
SYST:CAL:ALL:PORT2:REC:ATT 10
```

**Query Syntax**

```plaintext
SYSTem:CALibrate:ALL:PORT<n>:RECeiver:ATTen<value>[,src]
```

**Return Type**

Numeric

**Default**

0
SYSTem:CALibrate:ALL[1-250]:PORT<n>:RECeiver:ATTen:REFerence<num>

**Applicable Models:** N522xB, N523xB, N524xB, M9485A

**Parameters**

- **[1-250]** Calibration number. The default is 1.
- **<n>** Receiver port number.
- **<num>** Attenuation value in dB for a Cal All calibration. 0dB or 35dB.

If a number other than these is entered, the analyzer will select the next lower valid value. For example, if 19dB is entered, then 0dB attenuation will be selected.

<table>
<thead>
<tr>
<th>Examples</th>
<th>SYST:CAL:ALL:PORT2:REC:ATT:REF 0</th>
</tr>
</thead>
</table>

**Query Syntax**

SYSTem:CALibrate:ALL:PORT<n>:RECeiver:ATTen:REFerence?

**Return Type** Numeric. If querying for the standard port, the return value is 0

**Default** 35

---

SYSTem:CALibrate:ALL[1-250]:PORT<n>:RECeiver:ATTen:TEST<num>

**Applicable Models:** N522xB, N523xB, N524xB, M9485A

**Parameters**

- **[1-250]** Calibration number. The default is 1.
- **<n>** Receiver port number.
- **<num>** Attenuation value in dB for a Cal All calibration. 0dB or 35dB.

If a number other than these is entered, the analyzer will select the next lower valid value. For example, if 19dB is entered, then 0dB attenuation will be selected.

<table>
<thead>
<tr>
<th>Examples</th>
<th>SYST:CAL:ALL:PORT2:REC:ATT:TEST 0</th>
</tr>
</thead>
</table>

**Query Syntax**

SYSTem:CALibrate:ALL:PORT<n>:RECeiver:ATTen:TEST?

**Return Type** Numeric. If querying for the standard (M9376A) port, the return value is 0.
System:CALibrate:ALL[1-250]:PORT<n>:SOURce:POWer:ATTen<value>[,src]

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Write-Read) Sets and returns the Source Attenuator setting for the Cal All calibration.

Parameters

[1-250] Calibration number. The default is 1.
<n> Source port number.
<value> Attenuation value in dB for the Cal All calibration. Choose a valid value for the VNA model. See valid settings.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples


Query Syntax
SYSTem:CALibrate:ALL:PORT<n>:SOURce:POWer:ATTen?

Return Type Numeric

Default 0

System:CALibrate:ALL[1-250]:PORT<n>:SOURce:POWer:OFFSet <value>[,src]
**Applicable Models:** All

*(Write-Read)* Sets and returns the power offset value for a Cal All calibration.

Power Offset provides a method of compensating port power for added attenuation or amplification in the source path. The result is that power at the specified port reflects the added components.

**Parameters**

- **[1-250]** Calibration number. The default is 1.
- **<n>** Source port number.
- **<value>** Power offset value in dB for a Cal All calibration.

- For amplification, use positive offset.
- For attenuation, use negative offset.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port , or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
```

**Query Syntax**

`SYSTem:CALibrate:ALL:PORT<n>:SOURce:POWer:OFFSet?`

**Return Type**

Numeric

**Default**

0

```
SYSTem:CALibrate:ALL[1-250]:PORT<n>:SOURce:POWer[:VALue] <value>[,src]
```
Applicable Models: All

(Write-Read) Sets and returns the power level at which a Cal All calibration is to be performed.

Parameters

- [1-250] Calibration number. The default is 1.
- <n> Source port number.
- <value> Power level at which the calibration is to be performed.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:CAL:ALL:PORT2:SOUR:POW 0</td>
<td>Numeric</td>
<td>Preset power of the VNA model.</td>
</tr>
<tr>
<td>SYST:CAL:ALL:PORT&lt;n&gt;:SOUR:POW[:VALue]?</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

See the data sheet for the power level for each model.

---

SYSTem:CALibrate:ALL[1-250]:RESet

Applicable Models: All

(Write-only) Resets all properties associated with the Cal All session to their default values.

Parameters

- [1-250] Calibration number. The default is 1.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:CAL:ALL:RES</td>
<td></td>
<td>Not Applicable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SYSTem:CALibrate:ALL[1-250]:SELect <value>
Applicable Models: All

(Write-Read) Sets and returns the list of channels to be calibrated during the Cal All session.

**Parameters**

| [1-250] | Calibration number. The default is 1. |
| <value> | Channel numbers to be calibrated. These channels must already exist. |

**Examples**

```
SYST:CAL:ALL:SEL 1, 2, 3
```

**Query Syntax**

SYSTem:CALibrate:ALL:SELect?

**Return Type**

Comma-separated channel numbers.

**Default**

Existing channels
System:Calibrate:Phase

Contains the settings to perform an SMC Phase Reference Calibration.

```
SYSTem:CALibrate:PHASe
   CKit
   CONNector
   DEEMbed
   FREQuency:
   | START
   | STOP
   GUIDed:
   | CHANnel?
   PORT
   POWer:ATTenuator
   REFerence:
   | CATalog?
   RESet
   UNKNown:
   | INCLude
   | INPUT:POWer
   | LO
   | FREQuency
   | POWer
```

Click on a red keyword to view the command details.

**Important Notes**

- It is **NOT** necessary to create an SMC measurement before performing a remote Phase Reference Cal. It is necessary when performed from the user interface.

- Before A..09.90, port selection was made remotely by selecting connectors and Cal Kits for the ports to be included in the SOLT calibration. With A.09.90, port selection is made explicitly with the commands in this node.

**See Also**
SYSTem:CALibrate:PHASe:CKIT <string>

**Applicable Models:** N522xB, N523xB, N524xB

*(Write-Read)* Sets and returns the Cal Kit that will be used to perform the S-parameter Cal.

To read a list of valid Cal Kits, use **SENS:e:CORR:COLL:GUID:CKIT:CAT?**

**Parameters**

- **<string>** Cal Kit.

**Examples**

```text
SYST:CAL:PHAS:CKIT "85052D"
```

**Query Syntax**

SYSTem:CALibrate:PHASE:CKIT?

**Return Type**

String

**Default**

" "

---

SYSTem:CALibrate:PHASe:CONNector <string>

**Applicable Models:** N522xB, N523xB, N524xB

*(Write-Read)* Sets and returns the connector type and gender of your Cal Kit.

To read a list of valid connector types, use **SENS:e:CORR:COLL:GUID:CONN:CAT?**

**Parameters**

- **<string>** Connector type.

**Examples**

```text
SYST:CAL:PHAS:CONN "APC 3.5 female"
```

**Query Syntax**

SYSTem:CALibrate:PHASE:CONNector?

**Return Type**

String

**Default**

" "

---

4955
SYSTem:CALibrate:PHASe:DEEMbed <bool>

Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the state of de-embedding (reversing) the port 2 coupler.

Parameters

<bool> Port 2 coupler de-embed state. Choose from:

ON (or 1) - Configures the calibration to include additional measurements to de-embed the effects of reversing the coupler. (This is the same as clearing the “Omit Coupler” checkbox.)

OFF (or 0) - Excludes additional measurements for de-embedding the effects of reversing the coupler.

Examples SYST:CAL:PHAS:DEEM 1

Query Syntax SYSTem:CALibrate:PHASE:DEEMbed?

Return Type Boolean

Default ON or 1

SYSTem:CALibrate:PHASE:FREQuency:STARt <value>

Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the phase reference cal start frequency.

Parameters

<value> Start frequency. Choose any frequency from 10 MHz to the stop frequency of the VNA.

Examples SYST:CAL:PHAS:FREQ:STAR 17.5e6

Query Syntax SYSTem:CALibrate:PHASE:FREQuency:STARt?

Return Type Numeric

Default Start frequency of the VNA

SYSTem:CALibrate:PHASE:FREQuency:STOP <value>
Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the phase reference cal stop frequency.

Parameters
<value>  Stop frequency. Choose any frequency within the range of the VNA.
Examples  SYST:CAL:PHAS:FREQ:STOP 26.5e9

Query Syntax  SYSTem:CALibrate:PHAs:FRequency:STOP?
Return Type  Numeric
Default  Stop frequency of the VNA

SYSTem:CALibrate:PHAs:GUIDed:CHANnel?

Applicable Models: N522xB, N523xB, N524xB

(Read-only) Reads the channel number of the Phase Reference Calibration. Use this value as the <ch> argument for the subsequent Guided:Cal commands.

Parameters  None
Examples  chan = SYST:CAL:PHAS:GUID:CHAN?
Return Type  Numeric
Default  Not applicable

SYSTem:CALibrate:PHAs:PORT<n> <bool>

Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the enable state for the specified port.

Parameters
<n>  Port number to enable or disable.
<bool>  Port enable state. Choose from:
ON (or 1) - Enable port <n>
OFF (or 0) - Disable port <n>
Examples  SYST:CAL:PHAS:PORT2 1
Query Syntax  SYSTem:CALibrate:PHAs:PORT<n>?
**Return Type**  Boolean

**Default**  Ports 1 and 2 are enabled.

Ports 3 and 4 (if present) are disabled

---

SYSTem:CALibrate:PHASe:POWer:ATTenuator <value>

**Applicable Models:** N522xB, N523xB, N524xB

*(Write-Read)* Sets and returns the Source Attenuator setting for the Phase Reference calibration.

**Note:** This setting MUST match the source attenuator setting at the mixer input port for subsequent SMC+Phase measurements.

**Parameters**

<value>  Attenuation value in dB. Choose a valid value for the VNA model. See valid settings.

**Examples**  SYST:CAL:PHAS:POW:ATT 10

**Query Syntax**  SYSTem:CALibrate:PHASe:POWer:ATTenuator?

**Return Type**  Numeric

**Default**  10 dB

---

SYSTem:CALibrate:PHASe:REFerence <string>

**Applicable Models:** N522xB, N523xB, N524xB

*(Write-Read)* Sets and returns the Phase Reference ID to be used for the Phase Reference calibration. Use SYST:CAL:PHAS:REF:CAT? to read the phase references currently connected to the VNA USB.

**Parameters**

<string>  Phase reference ID string.

**Examples**  SYST:CAL:PHAS:REF "MYPRT0001"

**Query Syntax**  SYSTem:CALibrate:PHASe:REFerence?

**Return Type**  String

**Default**  Not Applicable
SYSTem:CALibrate:PHASe:REFerence:CATalog?

Applicable Models: N522xB, N523xB, N524xB

(Read-only) Reads the ID strings of the phase references that are currently connected to the VNA USB.

Parameters

None

Examples

pRef = SYST:CAL:PHAS:REF:CAT?

Return Type

Comma-separated string

Default

Not Applicable

SYSTem:CALibrate:PHASe:RESet

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Resets all properties associated with the Phase Reference Cal to their default values.

Parameters

None

Examples

SYST:CAL:ALL:PHAS:RES

Query Syntax

Not Applicable

Default

Not Applicable

SYSTem:CALibrate:PHASe:UNKNown:INCLude <bool>

Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the state of Unknown Mixer calibration.

Parameters

<bool> Unknown Mixer cal state. Choose from:

ON (or 1) - Enable Unknown Mixer cal. The start frequency becomes 10 MHz and can NOT be changed.

OFF (or 0) - Disable Unknown Mixer cal.

Examples

SYST:CAL:PHAS:UNKN:INCL 1

Query Syntax

SYSTem:CALibrate:PHASe:UNKNown:INCLude?

Return Type

Boolean

Default

OFF or 0
SYSTem:CALibrate:PHASe:UNKown:INPut:POWer <value>

Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the input power level to the unknown mixer.

Parameters

<value>  Input power level in dBm.

Examples


Query Syntax

SYSTem:CALibrate:PHASe:UNKown:INPut:POWer?

Return Type  Numeric

Default  -15 dBm

---

SYSTem:CALibrate:PHASe:UNKown:LO:FREQuency <value>

Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the LO frequency to the unknown mixer.

Parameters

<value>  LO frequency in Hz. Choose a value between 3 GHz and (Max Frequency minus 1GHz).

For a 26.5 GHz VNA, the range is 3 GHz to 25.5 GHz.

For best results, use the default LO frequency. 3.351Ghz. This frequency produces no spurs from the input/LO frequency. And also the Input frequency will have no band breaks.

Examples

SYST:CAL:PHAS:UNK:LO:FREQ 3.351e9

Query Syntax

SYSTem:CALibrate:PHASe:UNKown:LO:FREQuency?

Return Type  Numeric

Default  3.351 GHz

---

SYSTem:CALibrate:PHASe:UNKown:LO:POWer <value>
Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the LO power level to the unknown mixer.

Parameters

<value>  LO power level in dBm.

Examples

SYSTem:CALibrate:PHASe:UNKown:LO:POWer 10

Query Syntax

SYSTem:CALibrate:PHASe:UNKown:LO:POWer?

Return Type

Numeric

Default

10 dBm
SYSTem:CAPability Commands

Reads various capabilities of the analyzer.

```
SYSTem:CAPability:
  ALC:POWer:
   | MAXimum[:LEVel]?  
   | MINimum[:LEVel]?  
  CHANnels:MAXimum[:COUNT]?  
  DELay
   | TRIGger
    | MAX?
    | MIN?
  FOM:EXISts?
  FREQuency
   | MAXimum?
   | MINimum?
  HARDware:
   | ATTenuator:RECeiver:
    | EXISts?
    | MAXimum?
    | STEP[:SIZE]?
   | ATTenuator:SOURce:
    | MAXimum?
    | STEP[:SIZE]?
   | DC:RECeiver
    | INTernal:CATalog?
    | INTernal:COUNt?
   | DC:SOURce
    | INTernal:CATalog?
    | INTernal:COUNt?
   | IF
    | MAXimum?
    | MINimum?
```
SYSTem:CAPability:HARDware:POWer Commands

These commands provide access to data sheet specified and typical, max and min power levels (in dBm). Max power refers to the maximum leveled source power at the specified port. Min power is calculated by subtracting the power sweep range from the max leveled power. This information is stored by frequency band in a power specification file. These commands provide access to the file’s contents and provide an interface to configure the port number and RF signal path of interest.


No measurement of instrument-specific dynamic range is performed; all power levels are equivalent to power data published in device data sheets. Power levels are valid only for measurement configurations where the front panel jumpers are in their standard positions, as originally shipped. Internal source attenuation and any calibrated external path loss/gain due to cables, fixtures, switches or booster amplifiers are not included in the reported min/max leveled power values. It remains the users’ responsibility to transform the reported factory power range data to a value corresponding to the specific calibration plane of their setups.

The power range data files contain both specified min/max leveled power values and the corresponding “typical” values. Some paths, that are not part of the specifications of the instrument may only have typical data. Only the “Specified” power range data is guaranteed for an instrument with an up-to-date calibration certificate.

Setting Up Power Range

The following describes the command sequence for setting up power range to query the right information.

Setup Power Range to correct path and path settings:

1. Set to the port that you would like to query leveled output power
   SYST:CAP:HARD:POW:PORT <portNum>

2. Find path elements available for the port specified (Optional)

3. Find settings for a path element name given by the above command (Optional)

4. Specify the path element and the value to query power data from

Query power limits for the port using a specific path and path settings:

Discrete frequencies: (CW)

1. Input discrete frequencies of interest

2. Query maximum power limit of each frequency in the list

3. Query minimum power limit of each frequency in the list

4. Query most restrictive maximum of all frequencies in the list
SYST:CAP:HARD:POW:DISC:MAX?

5. Query most restrictive minimum of all frequencies in the list
SYST:CAP:HARD:POW:DISC:MIN?

Frequency range: (Freq sweep)

1. Input start frequency

2. Input stop frequency

3. Query the most restrictive maximum within the specified range
SYST:CAP:HARD:POW:RANG:MAX?

4. Query the most restrictive minimum within the specified range
SYST:CAP:HARD:POW:RANG:MIN?

See Also

- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree
### SYSTem:CAPability:DE Lay:TRIGger:MAX?

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

*(Read-only)* Returns the maximum trigger delay of the analyzer.

**Parameters**
- None

**Examples**
- `SYST:CAP:DEL:TRIG:MAX?`

**Return Type**
- Numeric

**Default**
- Not Applicable

---

### SYSTem:CAPability:DE Lay:TRIGger:MIN?

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

*(Read-only)* Returns the minimum trigger delay of the analyzer.

**Parameters**
- None

**Examples**
- `SYST:CAP:DEL:TRIG:MIN?`

**Return Type**
- Numeric

**Default**
- Not Applicable

---

### SYSTem:CAPability:ALC:POWer:MAXimum[:LEVel]?

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, E5080A, P937xA

*(Read-only)* Returns the maximum leveled source power setting in dB. [Learn more about leveled source power.](#)

**Parameters**
- None

**Examples**
- `SYST:CAP:ALC:POW:MAX?`

**Return Type**
- Numeric

**Default**
- Not Applicable

---

### SYSTem:CAPability:ALC:POWer:MINimum[:LEVel]?

---
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, E5080A, P937xA

*(Read-only)* Returns the minimum leveled source power setting in dB with 0 dB attenuation. Learn more about leveled source power.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Examples</strong></td>
<td><code>SYST:CAP:ALC:POW:MIN?</code></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Numeric</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**SYSTEM:CAPability:CHANnels:MAXimum[:COUNt]??**

**Applicable Models:** All

*(Read-only)* Returns the maximum possible number of channels. Learn more about Channels.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Examples</strong></td>
<td><code>SYST:CAP:CHAN:MAX?</code></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Numeric</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**SYSTEM:CAPability:FOM:EXISSts??**

**Applicable Models:** All

*(Read-only)* Returns whether or not the analyzer has FOM installed. Learn more.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Examples</strong></td>
<td><code>SYST:CAP:FOM:EXIS?</code></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td></td>
<td>1 - Yes, FOM is installed.</td>
</tr>
<tr>
<td></td>
<td>0 - No, FOM is NOT installed.</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**SYSTEM:CAPability:FREQuency:MAXimum??**
**SYStem:CAPability:FREQuency:MAXimum?**

**Applicable Models:** All

*(Read-only)* Returns the maximum frequency of the analyzer, including any over-sweep. Over-sweep frequencies can be set but are not specified.

- **Parameters** None
- **Examples** SYST:CAP:FREQ:MAX?
- **Return Type** Numeric
- **Default** Not Applicable

**SYStem:CAPability:FREQuency:MINimum?**

**Applicable Models:** All

*(Read-only)* Returns the minimum frequency of the analyzer, including any under-sweep. Under-sweep frequencies can be set but are not specified.

- **Parameters** None
- **Examples** SYST:CAP:FREQ:MIN?
- **Return Type** Numeric
- **Default** Not Applicable


**Applicable Models:** N522xB, N523xB, N524xB, M937xA, E5080A, P937xA

*(Read-only)* Returns whether or not there is a receiver attenuator on the specified port.

- **Parameters**
  - `<portNum>` Port number. Choose from the number of test ports on the analyzer.
- **Return Type** Boolean
  - 1 - Yes, the test port has a receiver attenuator.
  - 0 - No, the test port does NOT have a receiver attenuator.
- **Default** Not Applicable

Applicable Models: N522xB, N523xB, N524xB, M937xA, E5080A, P937xA

(Read-only) Returns the maximum amount of receiver attenuation on the specified port.

Parameters
<portNum> Port number. Choose from the number of test ports on the analyzer.

Examples

Return Type Numeric
Default Not Applicable

SYSTem:CAPability:HARDware:ATTenuator:RECeiver:STEP[:SIZE]? <portNum>

Applicable Models: N522xB, N523xB, N524xB, M937xA, E5080A, P937xA

(Read-only) Returns the step size of the receiver attenuator on the specified port.

Parameters
<portNum> Port number. Choose from the number of test ports on the analyzer.

Examples

Return Type Numeric
Default Not Applicable


Applicable Models: N522xB, N523xB, N524xB, M937xA, E5080A, P937xA

(Read-only) Returns the maximum amount of source attenuation on the specified port.

Parameters
<portNum> Port number. Choose from the number of test ports on the analyzer.

Examples

Return Type Numeric
Default Not Applicable

SYSTem:CAPability:HARDware:ATTenuator:SOURce:STEP[:SIZE]? <portNum>

4970
Applicable Models: N522xB, N523xB, N524xB, M937xA, E5080A, P937xA

(Read-only) Returns the step size of the source attenuator on the specified port.

**Parameters**

- `<portNum>` Port number. Choose from the number of test ports on the analyzer.

**Examples**

```
```

**Return Type** Numeric

**Default** Not Applicable

---


Applicable Models: N522xB, N523xB, N524xB, M937xA, E5080A/B, P937xA

(Read-only) Returns a list of names of the internal DC receivers.

**Parameters** None

**Examples**

```
```

**Return Type** String of internal DC receivers separated by commas.

For example, "AI1, AI2, AIG, AOS1, AOS2"

**Default** Not Applicable

---

**SYSTem:CAPability:Hardware:DC:RECeiver:INTernal:COUNt?**

Applicable Models: N522xB, N523xB, N524xB, M937xA, E5080A/B, P937xA

(Read-only) Returns the number of internal DC receivers in the analyzer.

**Parameters** None

**Examples**

```
```

**Return Type** Numeric

**Default** Not Applicable

---

**SYSTem:CAPability:Hardware:DC:SOURce:INTernal:CATalog?**
### Applicable Models:
N522xB, N523xB, N524xB, M937xA, E5080A/B, P937xA

(Read-only) Returns a list of names of the internal DC sources.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>String of internal DC sources separated by commas.</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

#### SYSTem:CAPability:HARDware:DC:SOURce:INTernal:COUNt?

### Applicable Models:
N522xB, N523xB, N524xB, M937xA, E5080A/B, P937xA

(Read-only) Returns the number of internal DC sources in the analyzer.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

#### SYSTem:CAPability:HARDware:IF:MAXimum?

### Applicable Models:
N522xB, N523xB, N524xB

(Read-only) Returns the maximum IF frequency the instrument supports.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>SYST:CAP:HARD:IF:MAX?</td>
</tr>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

#### SYSTem:CAPability:HARDware:IF:MINimum?
### Applicable Models:
N522xB, N523xB, N524xB

(Read-only) Returns the minimum IF frequency the instrument supports.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td><code>SYST:CAP:HARD:IF:MIN?</code></td>
</tr>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

### SYStem:CAPability:HARDware:LFEXtension:EXISts?

Applicable Models: N5222B, N5227B, N5242B, N5247B, N5290A, N5291A

(Read-only) Returns whether or not the VNA has the low frequency extension (LFE) installed. Learn more.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td><code>SYST:CAP:HARD:LFEX:EXISts?</code></td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

### SYStem:CAPability:HARDware:PORTs:CATalog?

Applicable Models: All

(Read-only) Returns a list of test port names including external testset ports.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td><code>SYST:CAP:HARD:PORT:CAT?</code></td>
</tr>
<tr>
<td>Return Type</td>
<td>String of port names separated by commas. For example, &quot;Port 1,Port 2,Port 3,Port 4&quot;.</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

### SYStem:CAPability:HARDware:PORTs:COUNt?

4973
Applicable Models: All

(Read-only) Returns the number of test ports including external testset ports.

Parameters: None

Examples: SYST:CAP:PORT:COUN?

Return Type: Numeric

Default: Not Applicable

SYSTem:CAPability:HARDware:PORTs:INTernal:CATalog?

Applicable Models: All

(Read-only) Returns a list of internal test port names.

Parameters: None

Examples: SYST:CAP:PORT:INT:CAT?

Return Type: String of port names separated by commas.
For example, "Port 1,Port 2,Port 3,Port 4".

Default: Not Applicable

SYSTem:CAPability:HARDware:PORTs:INTernal:COUNt?

Applicable Models: All

(Read-only) Returns the number of internal test ports.

Parameters: None

Examples: SYST:CAP:PORT:INT:COUN?

Return Type: Numeric

Default: Not Applicable

SYSTem:CAPability:HARDware:PORTs:PNUMber? <portName>
**Applicable Models:** All

(Read-only) Returns the port number associated with the specified port name.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
</table>

**Examples**

```plaintext
SYST:CAP:HARD:PORT:PNUM? "Port 1"
```

**Return Type** Numeric

**Default** Not Applicable

**SYSTem:CAPability:HARDware:PORTs:SOURce:CATalog?**

**Applicable Models:** All

(Read-only) Returns a list of source port names, including any configured external sources.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>String of source port names separated by commas.</td>
</tr>
<tr>
<td></td>
<td>For example, &quot;Port 1,Port 2,Port 3,Port 4, Port 1 Src 2,Source3&quot;.</td>
</tr>
</tbody>
</table>

**Default** Not Applicable

**SYSTem:CAPability:HARDware:PORTs:SOURce:COUNt?**

**Applicable Models:** All

(Read-only) Returns the number of source ports, including any configured external sources.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Numeric</td>
</tr>
</tbody>
</table>

**Default** Not Applicable

**SYSTem:CAPability:HARDware:PORTs:SOURce:INTernal:CATalog?**
Applicable Models: All

(Read-only) Returns a list of internal source port names.

**Parameters** None


**Return Type** String of internal source port names separated by commas.

For example, "Port 1,Port 2,Port 3,Port 4, Port 1 Src 2,Source3"

**Default** Not Applicable

---

`SYSTem:CAPability:HARDware:PORTs:SOURce:INTernal:COUNt?`

Applicable Models: All

(Read-only) Returns the number of internal source ports.

**Parameters** None

**Examples** `SYST:CAP:HARD:PORT:SOUR:INT:COUN?`

**Return Type** Numeric

**Default** Not Applicable

---


Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets or returns the list of discrete frequencies corresponding to the powers returned by the discrete min and max power list functions.

**Parameters**

- `<freqList>` List of frequencies for which power is returned.

**Examples** `SYST:CAP:HARD:POW:DISC:FREQ:LIST 1e9,2e9,3e9,4e9`

`system:capability:hardware:power:discrete:frequency:list 10e6,100e6,1e9,10e9`

**Query Syntax** `SYSTem:CAPability:HARDware:POWer:DISCrete:FREQuency:LIST?`

**Return Type** Array

**Default** Not Applicable
SYSTem:CAPability:HARDware:POWer:DISCrete:MAXimum?

**Applicable Models:** N522xB, N523xB, N524xB

*(Read-only)* Returns a single max leveled power value (in dBm) indicating the most restrictive maximum for all discrete maximum powers (the minimum of all max leveled powers).

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
</table>
| Examples   | SYST:CAP:HARD:POW:DISC:MAX?  
System:capability:hardware:power:discrete:maximum? |
| Return Type| Double |
| Default    | Not Applicable |

SYSTem:CAPability:HARDware:POWer:DISCrete:MAXimum:LIST?

**Applicable Models:** N522xB, N523xB, N524xB

*(Read-only)* Returns an array of max leveled power values (in dBm), where each element corresponds to the maximum leveled power possible for CW stimulus at the corresponding frequency set by the SYSTem:CAPability:HARDware:POWer:DISCrete:FREQuency:LIST command.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
</table>
System:capability:hardware:power:discrete:maximum:list? |
| Return Type| Array |
| Default    | Not Applicable |

SYSTem:CAPability:HARDware:POWer:DISCrete:MINimum?
**Applicable Models:** N522xB, N523xB, N524xB

*(Read-only)* Returns a single minimum power value (in dBm) indicating the most restrictive minimum for all discrete minimum powers (the maximum of all minimum powers).

**Parameters** None

**Examples**

```
SYST:CAP:HARD:POW:DISC:MIN?

system:capability:hardware:power:discrete:minimum?
```

**Return Type** Double

**Default** Not Applicable

---

**SYSTem:CAPability:HARDware:POWer:DISCrete:MINimum:LIST?**

**Applicable Models:** N522xB, N523xB, N524xB

*(Read-only)* Returns an array of minimum power values (in dBm), where each element corresponds to the minimum power possible for CW stimulus at the corresponding frequency set by the SYSTem:CAPability:HARDware:POWer:DISCrete:FREQuency:LIST command.

**Parameters** None

**Examples**

```

System:capability:hardware:power:discrete:minimum:list?
```

**Return Type** Array

**Default** Not Applicable

---

**SYSTem:CAPability:HARDware:POWer:PATH:CONFig:ELEMent:CATalog?**

**Applicable Models:** N522xB, N523xB, N524xB

*(Read-only)* Returns a string with the names of all valid RF path configuration elements.

**Parameters** None

**Examples**

```

```

**Return Type** String

**Default** Not Applicable
SYSTem:CAPability:HARDware:POWer:PATH:CONFig:ELEMent[:STATe]
<element>,<setting>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Returns the name of the value for the given path element name or sets the value of a path element.

Parameters
<element>  String - Choose from all path elements listed with SYST:CAP:_HARD:POW:PATH:CONF:ELEM:CAT?.

Examples
SYST:CAP:_HARD:POW:PATH:CONF:ELEM "Src1Out1LowBand","HiPwr"
SYST:CAP:_HARD:POW:PATH:CONF:ELEM:STAT "Port1Bypass","Thru"

Query Syntax
SYSTem:CAPability:HARDware:POWer:PATH:CONFig:ELEMent:STATe?
“Src2Out1LowBand”

Returns the setting for the specified element.

Return Type
String
Default  Not Applicable


Applicable Models: N522xB, N523xB, N524xB

(Read-only) Returns all valid values for the given path configuration element.

Parameters
<PathElementName>  (String) Chose from all path elements (see PATH:CONF:ELEM:CAT?)

Examples

Return Type
String
Default  Not Applicable

SYSTem:CAPability:HARDware:POWer:PORT <portNum>
**Applicable Models:** N522xB, N523xB, N524xB

*(Read-Write)* Sets and reads the port number for power range data. When two sources are available in combiner mode, refer to the portNum selections below.

### Parameters

<portNum> Port number. Choose from:

1 - Port 1.
2 - Port 2.
3 - Port 3 (4-port instrument) or Src2-Out1 (2-port instrument with option 224).
4 - Port 4 (4-port instrument) or Src2-Out2 (2-port instrument with option 423).
5 - Src2Out1LowBand (2-port with option 224 or 4-port with option 423).
6 - Source 3

### Examples

SYST:CAP:HW:POW:PORT 1

### Query Syntax

SYSTem:CAPability:HW:POW:PORT?

### Return Type

Integer

### Default

Not Applicable

SYSTem:CAPability:HW:POW:RANGe:FREQuency:STARt <num>
**Applicable Models:** N522xB, N523xB, N524xB

*(Read-Write)* Sets or returns the lower bound of the frequency range used for range based power min and max.

**Parameters**

| <num> | Start frequency. Choose a number within the frequency limits of the analyzer. Units are Hz. |

**Examples**

| system:capability:hardware:power:range:frequency:start 1e9 |

**Query Syntax**

SYSTem:CAPability:HARDware:POWer:RANGe:FREQuency:STARt?

**Return Type**

Double

**Default**

Not Applicable

---

**SYSTem:CAPability:HARDware:POWer:RANGe:FREQuency:STOP <num>**

**Applicable Models:** N522xB, N523xB, N524xB

*(Read-Write)* Sets or returns the upper bound of the frequency range used for range based power min and max.

**Parameters**

| <num> | Stop frequency. Choose a number within the frequency limits of the analyzer. Units are Hz. |

**Examples**


**Query Syntax**

SYST:CAP:HW:POW:RANG:FREQ:STOP?

**Return Type**

Double

**Default**

Not Applicable

---

**SYSTem:CAPability:HARDware:POWer:RANGe:MAXimum?**
Applicable Models: N522xB, N523xB, N524xB

(Read-only) Returns the minimum of all max leveled power values (in dBm) from range start frequency to stop frequency (inclusive).

Parameters
None

Examples
SYST: CAP: HARD: POW: RANG: MAX?

Return Type
Double
Default
Not Applicable

System: Capability: Hardware: Power: Range: Minimum?

Applicable Models: N522xB, N523xB, N524xB

(Read-only) The maximum of all min power values (in dBm) from range start frequency to stop frequency (inclusive).

Parameters

Examples
SYST: CAP: HARD: POW: RANG: MIN?

Return Type
Double
Default
Not Applicable

System: Capability: Hardware: Power: Reset

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Resets all power range properties to default values, as if the instrument had been preset. Power range type is set to SPECified, port number is set to 1 with all path configuration elements in their default states.

Parameters
None

Examples
SYST: CAP: HARD: POW: RES
system: capability: hardware: power: reset

Default
Not Applicable

System: Capability: Hardware: Power: Type <enum>
Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets and reads the type of power range data (specified or typical) to be returned.

Parameters
<enum> Choose from:

**SPECified** - Warranted performance.

**TYPical** - Typical performance.

Examples

<table>
<thead>
<tr>
<th>SYST:CAP:HARD:POW:TYPE SPEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>system:capability:hardware:power:type typical</td>
</tr>
</tbody>
</table>

Query Syntax
SYSTem:CAPability:HARDware:POWer:TYPE?

Return Type
Enumeration

**Default** SPECified

---


Applicable Models: All

(Read-only) Returns whether or not the specified port number has a reference bypass switch.

Parameters

<portNum> Port number. Choose from the number of test ports on the analyzer.

Examples


Return Type

Boolean

1 - Yes, the test port has a reference bypass switch.

0 - No, the test port does NOT have a reference bypass switch.

**Default** Not Applicable

---

SYSTem:CAPability:HARDware:RECeiver:INTernal:COUNt?
Applicable Models: All

(Read-only) Returns the number of receivers in the analyzer.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td><code>SYST:CAP:Hard:Rec:Int:Coun?</code></td>
</tr>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**SYSTem:CAPability:Hardware:RECeiver:DACCess?**

Applicable Models: N522xB, N523xB, N524xB, E5080A

(Read-only) Returns whether or not the analyzer has direct receiver access (front-panel jumpers).

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
</table>
| <bool>     | Choose from:
| 1          | Yes, the analyzer has direct receiver access. |
| 0          | No, the analyzer does NOT have direct receiver access. |
| Examples   | `SYST:CAP:Hard:Rec:Dacc?` |
| Return Type| Boolean |
| Default    | Not Applicable |

**SYSTem:CAPability:Hardware:SOURce:COUNt?**

Applicable Models: All

(Read-only) Returns the number of sources in the analyzer.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td><code>SYST:CAP:Hard:Sour:Coun?</code></td>
</tr>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**SYSTem:CAPability:IFBW:CATalog?**
<table>
<thead>
<tr>
<th>Application Models: All</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>(Read-only)</strong> Returns the list of supported IFBW values.</td>
</tr>
<tr>
<td><strong>Parameters</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
</tr>
<tr>
<td><strong>Default</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System:Capability:IFBW:Maximum?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Applicable Models: All</strong></td>
</tr>
<tr>
<td><strong>(Read-only)</strong> Returns the maximum IFBW for the standard IF filter.</td>
</tr>
<tr>
<td><strong>Parameters</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
</tr>
<tr>
<td><strong>Default</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System:Capability:IFBW:Minimum?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Applicable Models: All</strong></td>
</tr>
<tr>
<td><strong>(Read-only)</strong> Returns the minimum IFBW for the standard IF filter.</td>
</tr>
<tr>
<td><strong>Parameters</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
</tr>
<tr>
<td><strong>Default</strong></td>
</tr>
</tbody>
</table>
Applicable Models: All

*(Read-only)* Returns the list of licenses. See a list of common licenses.

**Parameters**

Choose from:

**VALID** - Return a list of licenses which have enabled VNA features. For PXI/USB VNA, only software licenses are returned. For PNA/ENA, all options including hardware are returned.

**ALL** - Return a list of all installed licenses/options in the Keysight License Manager including the ones not related to the VNA software.

**IGNORED** - Return a list of VNA software licenses which are either invalid or ignored. This can occur when a transportable license is transported to an instrument that does not support the license feature. In addition, this can occur when multiple licenses for the same base feature are installed and only the least restrictive license is used (the more restrictive licenses are ignored). For example, when transporting multiple Spectrum Analyzer licenses to the same instrument, the license with the greatest frequency range is used and the other licenses are ignored.

**Note:** Licenses not related to the VNA software but installed on the instrument are not reported as ignored when using **IGNORED**.

**Examples**

```
SYST:CAP: LIC: CAT? ALL
"N5242B-423, N5242B-020, N5242B-021, N5242B-022, S93029A/B-1FP"
```

**Return Type**

Variant array of string values

**Default**

Not Applicable

```
SYSTem:CAPability:NBW:NOISe:CATalog?
```
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA, M980xA, P50xxA

(Read-only) Returns the list of supported Noise Bandwidth values when using a noise receiver (option 029). Learn more about Opt. 029.

Parameters  None
Examples  SYST:CAP:NBW:NOIS:CAT?
Return Type  Variant array of string values
Default  Not Applicable

SYSTem:CAPability:NBW:STD:CATalog?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Returns the list of supported Noise Bandwidth values when using the NA receiver for noise measurements (option 028). Learn more about Opt 028.

Parameters  None
Examples  SYST:CAP:NBW:STD:CAT?
Return Type  Variant array of string values
Default  Not Applicable

SYSTem:CAPability:POINts:MAXimum?

Applicable Models: All

(Read-only) Returns the maximum number of points.

Parameters  None
Examples  SYST:CAP:POIN:MAX?
Return Type  Numeric
Default  Not Applicable

SYSTem:CAPability:POINts:MINimum?
Applicable Models: All

(Read-only) Returns the minimum number of points.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>SYST:CAP:POIN:MIN?</td>
</tr>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

SYSTem:CAPability:PRESet:FREQuency:MAXimum?

Applicable Models: All

(Read-only) Returns the maximum specified frequency of the analyzer. Does not include any over-sweep. See also: SYST:CAP:FREQ:MAX?

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>SYST:CAP:PRES:FREQ:MAX?</td>
</tr>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

SYSTem:CAPability:PRESet:FREQuency:MINimum?

Applicable Models: All

(Read-only) Returns the minimum specified frequency of the analyzer. Does not include any under-sweep. See also: SYST:CAP:FREQ:MIN?

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>SYST:CAP:PRES:FREQ:MIN?</td>
</tr>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

SYSTem:CAPability:RBW:IMS:CATalog?
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Returns the list of supported Resolution BW values for the IMSpectrum channel. Learn more about IMSpectrum.

Parameters

None

Examples

SYST:CAP:RBW:IMS:CAT?

Return Type

Variant array of string values

Default

Not Applicable

SYSTem:CAPability:RBW:SA:CATalog?

Applicable Models: All with Spectrum Analysis Options (S9x09xA/B, S9x090A/B)

(Read-only) Returns the list of supported Resolution BW values for the SA channel. Learn more about the SA application.

Parameters

None

Examples

SYST:CAP:RBW:SA:CAT?

Return Type

Variant array of string values

Default

Not Applicable

SYSTem:CAPability:SUPPort:DWELl? <ifBandwidth>

Applicable Models: VNAs with S93070xA/B or S95070A/B

(Read-only) Returns whether a given IF Bandwidth value supports dwell time (dwell time between each sweep point). Dwell time can be used for all IF Bandwidth values.

Parameters

<ifBandwidth> IF Bandwidth value.

Examples

SYST:CAP:SUPP:DWEL? 1E3

Return Type

Boolean

0 = False

1 = True

Default

Not Applicable
SYSTem:CAPability:SUPPort:RIFBw?

Applicable Models: VNAs with S93070xA/B or S95070A/B

(Read-only) Returns whether the VNA supports the Reduce IF BW at Low Frequencies feature. The noise floor increases at low frequencies. By reducing the IFBW automatically, the noise floor is kept constant. Otherwise, measurements get noisier at low frequencies. When a constant bandwidth is required (for pulse measurement for instance), then the Reduce IF BW at Low Frequencies is turned off.

Parameters None

Examples SYST:CAP:SUPP:RIFB?

Return Type Boolean

0 = False

1 = True

Default Not Applicable


SYSTem:CAPability:WINDows:MAXimum[:COUNt]?

Applicable Models: All

(Read-only) Returns the maximum number of windows. Learn more.

Parameters None

Examples SYST:CAP:WIND:MAX?

Return Type Numeric

Default Not Applicable


SYSTem:CAPability:WINDows:TRACes:MAXimum[:COUNt]?

Applicable Models: All

(Read-only) Returns the maximum number of traces per window. Learn more.

Parameters None

Examples SYST:CAP:WIND:TRAC:MAX?

Return Type Numeric

Default Not Applicable
Controls and queries settings that affect the VNA system.

SYSTem: COMMunicate Commands

SYSTem: COMMunicate:

DRIVE
  | ENABle?

ECAL
  | CATalog?
  | CLIST?
  | COUNT?

DMEMory
  | CLEar
  | IMPort
  | EXPort
  | SNP
  | INFormation?
  | KNAMe:INFormation?
  | LIST?
  | PATH:COUNt?

GPIB
  | PMETer
    | CATalog?
  | ADDRess
  | RDEVice
    | CLOSe
    | OPEN
    | READ?
    | RESet
    | WBINary
    | WBLock
    | WRITe

LAN: HOSTname

PSENsor
TCPip: CONTrol?

USB: PMETer: CAT?

VISA
  | RDEVice
    | CLOSe
    | FIND?
    | OPEN
    | READ?
    | RESet
    | TIMEout
    | WBINary
Click on a keyword to view the command details.

See Also

- Referring to Traces Channels Windows and Meas Using SCPI
- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

SYSTem:COMMunicate:DRIVe:ENABle?

Applicable Models: All

(Read-only) Returns whether or not the Enable Remote Drive Access is checked in the Remote Interface dialog. When checked allows access to the hard disk.

<table>
<thead>
<tr>
<th>Examples</th>
<th>SYST:COMM:DRIV:ENAB?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

SYSTem:COMMunicate:ECAL:CATalog?

Applicable Models: All

(Read-only) Returns the ID string of ECals that are connected to the VNA USB. Use the list to select a Ecal for Ecal calibration.

<table>
<thead>
<tr>
<th>Examples</th>
<th>SYST:COMM:ECAL:CAT?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>system:communicate:ecal:catalog?</td>
</tr>
<tr>
<td>Return Type</td>
<td>Comma-delimited strings.</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

SYSTem:COMMunicate:ECAL<mod>:CLIS?
**SyStem:ComMuNicate:ECAL:CLiST?**

**Applicable Models:** All

*(Read-only)* Returns a list of characterizations stored in the specified ECal module.

**Parameters**

<mod> ECal module from which to read user characterization numbers. Choose from 1 to 50. If unspecified, value is set to 1.

**Examples**

```
Module 1 contains User Characterizations 1 and 3.
SYST:COMM:ECAL:CLIST?
'REturns the following (0 always indicates the factory characterization):
0,1,3
```

**Return Type** Numeric list, separated by commas.

**Default** Not Applicable

**SyStem:ComMuNicate:ECAL:COUNt?**

**Applicable Models:** All

*(Read-only)* Returns the number of installed cal kits.

**Examples**

```
SYST:COMM:ECAL:COUNt?
```

**Query Syntax**

SYST:COMM:ECAL:COUNt?

**Return Type** Numeric

**Default** Not Applicable

**SyStem:ComMuNicate:ECAL:DMEMory:CLEar <kitName>**
Applicable Models: All

(Write-only) Deletes user characterizations from VNA disk memory.

Parameters

<kitName>  Optional String argument. ECal Model, User Characterization name + " ECal", and serial number of the ECal module, separated by spaces. See examples below.

If unspecified, ALL User Characterizations that are stored in VNA disk memory are deleted.

Examples

'SThese examples all use "MyUserChar" as the User characterization name.

'The "My User Char" characterization is deleted from disk memory.

SYST:COMM:ECAL:DMEM:CLE "N4433A MyUserChar ECal 00001"

'All User characterizations are deleted from disk memory.

SYST:COMM:ECAL:DMEM:CLE

Query Syntax

Not Applicable

Default

Not Applicable

SYSTem:COMMunicate:ECAL:DMEMory:IMPort <file>

Applicable Models: All

(Write-only) After the VNA disk memory is Exported to a file, use this command to Import the file into VNA disk memory, which allows the User Characterization to be used with the VNA and ECal module.

Note: An ECal confidence check can NOT be performed remotely from User Characterizations that are stored on the VNA disk.

Parameters

<file>  String. Full path and file name of file that was exported.

Examples

SYST:COMM:ECAL:DMEM:IMP "c:\users\public\network analyzer\ECal User Characterizations/myDiskUserChar.euc"

Query Syntax

Not Applicable

Default

Not Applicable
SYSTem:COMMunicate:ECAL:EXP ort <kit>,[<file>],[<NewName>]]

Applicable Models: All

(Write-only) Saves an existing ECal characterization to a file. Use this command to archive the user characterization or to move the characterization to a different VNA for use with the specified ECal module. After exporting the user characterization, use SYST:COMM:ECAL:DMEM:IMPort to make the user characterization available for use.

Parameters

- **<kit>** String. Not case sensitive. ECal Model, User char name + " ECal", and serial number of the ECal module used for the characterization, separated by spaces. See examples below.

  If the model and serial number of the module is not found, an error is returned.

- **[<file>]** Optional String argument. Path and filename of the user characterization. If not specified, the file is saved using characterization name + ".euc". If the path is not specified, it is stored in C:\Program Files(x86)\Keysight\Network Analyzer\E Cal User Characterizations\ The extension ".euc" is appended if one is not specified.

- **[<NewName>]** Optional String argument. This allows you to change the name for the User Characterization. When specified, the new name is saved in the file with the characterization. If unspecified, the existing user characterization name is saved.

  **Note:** If this argument is specified, the second argument (<file>) must also be specified.

Examples

- These examples all use "MyUserChar" as the User characterization name.

  - All parameters specified

    SYST:COMM:ECAL:EXP "N4433A MyUserChar ECal 00001","myUserChar.euc","NewUserChar"

  - First two parameters are specified

    system:communicate:ecal:export "N4691B MyUserChar ECal 00500","myUserChar.euc"

  - Only first parameter is specified

    SYST:COMM:ECAL:EXP "N4433A MyUserChar ECal 00001"

Query Syntax

Not Applicable
SYSTem:COMMunicate:ECAL:EXPort:SNP <kit>,<ecalState>,<snpFileName>

**Applicable Models:** All

(Write-only) Read S parameter of ECal Thru from the ECal memory and save it as s2p file.

**Parameters**

- **<kit>**  String. Not case sensitive. ECal Model, User char name + " ECal", and serial number of the ECal module used for the characterization, separated by spaces. See examples below.
  
  If the model and serial number of the module is not found, an error is returned.

- **<ecalState>**  ECal transmission path. Choose from AB, AC, AD, BA, BC, BD, CA, CB, CD, DA, DB or DC. Not case sensitive.

- **<snpFileName>**  Path and filename of the output s2p file name.

**Examples**  

```
SYST:COMM:ECAL:EXP "N4433A ECal 00001","BC","D:\ecal thru.s2p"
```

**Query Syntax**  

Not Applicable

**Default**  

Not Applicable

---

SYSTem:COMMunicate:ECAL<mod>:INFormation? [<char>]

---

4997
Applicable Models: All

(Read-only) Reads the identification and characterization information from the specified ECal module.

Note: To read user-characterization information that is stored in VNA disk memory, then use SYST:COMM:ECAL:KNAM:INF?

Parameters

<mod> ECal module from which to read characterizations. Choose from 1 through 50. If unspecified, value is set to 1.

Do NOT assume the <mod> number is the order in which ECal modules were connected.

Use SYST:COMM:ECAL:LIST? to read a list of <mod> numbers of currently-attached ECal modules.

<char> Optional argument. Specifies which characterization to read information from. If not specified, value is set to CHAR0.

Choose from:

- CHAR0 Factory characterization (data that was stored in the ECal module by Keysight)
- CHAR1 User characterization #1
- CHAR2 User characterization #2
- ... through ...
- CHAR12 User characterization #12

Examples

SYST:COMM:ECAL2:INFormation? char5

'Example return string:

"ModelNumber: 85092-60007, SerialNumber: 01386, ConnectorType: N5FN5F, PortAConnector: Type N (50) female, PortBConnector: Type N (50) female, MinFreq: 30000, MaxFreq: 9100000000, NumberOfPoints: 250, Calibrated: July 4 2002"

Return Type Character

Default Not Applicable

SYSTem:COMMunicate:ECAL:KNAMe:INFormation? <kitName>
Applicable Models: All

(Read-only) Reads the identification and characterization information from the specified ECal module or VNA disk memory.

Learn more about User Characterization in VNA Disk Memory.

Parameters

<kitName> String. ECal model and characterization to read information from, enclosed in quotes, in the following format:

<model> <name> ECal <serial number>

Where:

<model>: Always required

<name>:

- For the factory characterization, do not specify.
- For a user-characterization stored in the module, use User <n> in the string, where <n> is the user-characterization number. Not case sensitive. Separate User and <n> with a space.
- For a user-characterization stored in VNA disk memory, use <charName> from SENS:CORR:CKIT:ECAL:CHAR:DMEM:SAVE <charName>

ECal - not case sensitive

<serial number>: Optional. Include when two or more ECal modules with same model number are attached to the VNA,

Each item is separated with a space.

Examples

'For a factory characterization in module memory:

'For user characterization in module memory with optional serial number:
SYST:COMM:ECAL:KNAM:INF? "N4433A User 1 ECal 00028"

'For user characterization "foo" in disk memory:
SYST:COMM:ECAL:KNAM:INF? "N4433A foo ECal 00028"

'Example return string:

"ModelNumber: N4433A, SerialNumber: 00028, ConnectorType: N5FN5F, PortAConnector: Type N (50) female, PortBConnector: Type N (50) female, MinFreq: 30000, MaxFreq: 9100000000,"
**SYSTem:COMMunicate:ECAL:LIST?**

**Applicable Models:** All

(Read-only) Returns a list of index numbers for ECal modules that are currently attached to the VNA. Use these numbers (called <mod> in VNAHelp) to refer to the ECal module using SCPI commands.

**Examples**

```
SYST:COMM:ECAL:LIST?

'If 2 modules are attached to the VNA
'then the returned list will be:
+1,+2

'If NO modules are attached to the VNA
'then the returned list will be:
+0
```

**Return Type** Numeric list, separated by commas.

**Default** Not Applicable

---

**SYSTem:COMMunicate:ECAL<n>:PATH:COUNt? <path>**

**Applicable Models:** All

(Read-only) Returns the number of unique states that exist for the specified path name on the selected ECal module.

This command performs exactly the same function as `CONT:ECAL:MOD:PATH:COUNt?`

Use the `CONT:ECAL:MOD:PATH:STAT` command to set the module into one of those states.

Use `SENS:CORR:CKIT:ECAL:PATH:DATA?` to read the data for a state.

**Parameters**

- `<n>` USB number of the ECal module. Choose from 1 to 50.

  If unspecified (only one ECal module is connected to the USB), `<n>` is set to 1. If two or more modules are connected, use `SYST:COMM:ECAL:LIST?` to determine how many, and `SYST:COMM:ECAL:INF?` to verify their identities.
<path> Name of the path for which to read number of states. Choose from:

Reflection paths

- A
- B
- C (4-port modules)
- D (4-port modules)

Transmission paths

- AB
- AC (4-port modules)
- AD (4-port modules)
- BC (4-port modules)
- BD (4-port modules)
- CD (4-port modules)

Examples

SYST:COMM:ECAL:PATH:COUNt?
system:communicate:ecal:path:count?

Return Type

- Integer

Default

- Not Applicable

SYSTem:COMMunicate:GPIB:PMETer[:ADDRESS] <num> Superseded
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

**Note:** This command is replaced with `SYST:COMM:PSEnson`.

*(Read-Write)* Specifies the GPIB address of the power meter to be used in a source power calibration. When performing a source power cal, the VNA will search VISA interfaces that are configured in the Keysight IO Libraries on the VNA.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;num&gt;</td>
<td>GPIB address of the power meter. Choose any integer between 0 and 30.</td>
</tr>
</tbody>
</table>

**Examples**

```plaintext
SYST:COMM:GPIB:PMET 13
```

```
system:communicate:gpi:pmeter:address 14
```

**Query Syntax**

```
SYSTem:COMMunicate:GPIB:PMETer[:ADDRess]?
```

**Return Type**

Numeric

**Default**

13


**Applicable Models:** All

*(Read-only)* Returns the ID string of power meters / sensors that are connected to the VNA via GPIB. Optionally, the visa address is returned. Use the list to select a power sensor for a *source power cal*.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;optional bool&gt;</td>
<td>Choose from:</td>
</tr>
</tbody>
</table>

| ON or 1               | Returns the visa address.                                                     |
| OFF or 0              | Does not return the visa address.                                             |

**Examples**

```plaintext
SYST:COMM:USB:PMET:CAT? ON
```

```
system:communicate:usb:pmeter:catalog?
```

**Return Type**

Comma-delimited strings. Two power sensor strings are separate by a semicolon.

**Default**

OFF

**SYSTem:COMMunicate:GPIB:RDEVice:CLOSE <ID>**

5002
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Write only) Closes the remote GPIB session. This command should be sent when ending every successful OPEN session.

Parameters

- `<ID>` Session identification number that was returned with the OPEN? command.

Examples

See an example program

Query Syntax

Not Applicable

Default

Not Applicable

SYSTem:COMMunicate:GPIB:RDEVice:OPEN <bus>, <addr>, <timeout>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Initiates a GPIB pass-through session. First send this OPEN command, then send the OPEN query to read the session ID number. An existing GPIB pass-through session remains open after an instrument preset.

To learn more about GPIB pass-through capability, see the example program.

Parameters

- `<bus>` Bus ID number.

You can find the USB-GPIB adapter bus number by looking at the dialog that appears when the USB-GPIB device is connected. Error 1073 indicates the bus or address number is incorrect.

Use 0 (zero) when connected using a GPIB cable to the VNA controller port.

- `<addr>` GPIB Address of the device to be controlled

- `<timeout>` The amount of time (in milliseconds) to wait for a response from the remote device after sending a command. A "timeout" error is displayed after this time has passed without a response.

Examples

See an example program

Query Syntax

SYSTem:COMMunicate:GPIB:RDEVice:OPEN?

Returns the session identification number that is used when communicating with this device.

Return Type

Numeric
SYSTem:COMMunicate:GPIB:RDEVice:READ? <ID>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Returns data from the GPIB pass-through device.

Parameters

- **<ID>** Session identification number that was returned with the OPEN? command.

Examples

- See an example program

Return Type String

- **Default** Not Applicable

SYSTem:COMMunicate:GPIB:RDEVice:RESet

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Write-only) Performs the same function as SYST:COMM:GPIB:RDEV:CLOS except that ALL pass-through sessions are closed.

Examples

- SYST:COMM:GPIB:RDEV:RES

- **Default** Not Applicable

SYSTem:COMMunicate:GPIB:RDEVice:WBINary <ID>,<data>
(Write-only) Sends data to a GPIB pass-through device. This command requires a header that specifies the size of the data to be written. The header (described below) is not passed along to the device.

Use this command if too many embedded quotes prevent you from using `SYST:COMM:GPIB:RDEV:WRIT`.

Use `SYST:COMM:GPIB:RDEV:OPEN` to open the pass through session.

**Parameters**

- `<ID>`: Session identification number that was returned with the `OPEN?` command.
- `<data>`: Data to be sent to the GPIB pass-through device. Use the following syntax:
  
  #<num digits><byte count><data bytes><NL><END>

  * `<num_digits>` specifies how many digits are contained in `<byte_count>`
  
  * `<byte_count>` specifies how many data bytes will follow in `<data bytes>`

**Examples**

```
SYSTem:COMMunicate:GPIB:RDEVice:WBINary 101,#17ABC+XYZ<nl><end>
```

# - always sent before data.

1 - specifies that the byte count is one digit (7).

7 - specifies the number of data bytes that will follow, not counting `<NL><END>`.

`ABC+XYZ` - Data block

`<nl><end>` - always sent at the end of block data.

The following example sends a line feed at the end.

```
SYST:COMM:GPIB:RDEV:WBIN 1,#210SYST:PRES<EOL>
```

The `<EOL>` represents your linefeed character.

**Query Syntax**

Not Applicable

**Default**

Not Applicable

```
SYSTem:COMMunicate:GPIB:RDEVice:WBLock <ID>,<data>
```
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Write-only)* Same as `SYSTem:COMM:GPIB:RDEV:WBIN` (above) but the header IS passed along to the device.

Use this command if too many embedded quotes prevent you from using `SYST:COMM:GPIB:RDEV:WRIT`.

**Parameters**

- `<ID>` Session identification number that was returned with the `OPEN?` command.
- `<data>` Data to be sent to the GPIB pass-through device. See previous command.

**Examples**

See previous example.

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

`SYSTem:COMMunicate:GPIB:RDEVice:WRITe <ID>,<string>`

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Write-only)* Sends ASCII string data to the GPIB pass-through device.

A line feed is NOT appended to the string data. To send a line feed, see the example in `SYST:COMM:GPIB:RDEV:WBIN`.

**Parameters**

- `<ID>` Session identification number that was returned with the `OPEN?` command.
- `<string>` Commands to be sent to the GPIB pass-through device.

**Examples**

See an example program

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

`SYSTem:COMMunicate:LAN:HOSTname?`
Applicable Models: All

(Read-only) Returns the LAN hostname that is visible in the Help, About Network Analyzer dialog box. Learn more. This is the same information that is visible on the LXI compliance dialog.

Parameters
None

Example
SYST:COMM:LAN:HOSTname?

Return Type
String

Default
Not applicable

SYSTem:COMMunicate:PSENsor <char>, <string>

Applicable Models: All

This command replaces SYST:COMM:GPIB:PMET:ADDR.

(Read-Write) Specifies the type and location of the power meter to be used in a source power calibration.

Parameters

<char> Type of power meter/ sensor. Choose from:

- GPIB  GPIB power meter
- USB  USB power sensor or USB power sensor
- LAN  LAN enabled power meter
- ANY  Any VISA resource string or a visa alias

<string> For GPIB, address of the power meter. Choose any integer between 0 and 30.

For USB, the ID string of the power meter or power sensor. Use SYST:COMM:USB:PMET:CAT? to see a list of ID strings of connected power meters and sensors.

For LAN, the hostname or IP address of the power meter.

For ANY, any VISA resource string or a visa alias.

Examples
SYST:COMM:PSEN gpib, "14"
system:communicate:psensor usb, "Keysight Technologies,U2000A,MY12345678"
syst:comm:psen lan, "mymeter.Keysight.com"

**Applicable Models:** All

*(Read-only)* Returns the ID string of power meters / sensors that are connected to the VNA USB. Optionally, the visa address is returned. Use the list to select a power sensor for a source power cal.

These meter/sensor ID strings can NOT be used as the resource string for configuring a USB-based PMAR (*SYST:CONF:EDEV:IOConfig)*.

**Parameters**

Choose from:

- **ON** or **1** - Returns the visa address.
- **OFF** or **0** - Does not return the visa address.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

**Return Type**

Comma-delimited strings. Two power sensor strings are separate by a semicolon.

**Default**

OFF

---

SYSTem:COMMunicate:TCPip:CONTrol?
Applicable Models: All

(Read-only) Queries the TCP/IP port number to use for opening a TCP/IP socket control connection to the VNA. The control connection is used for two purposes:

1. To perform a Device Clear operation on the VNA
2. To detect when a Service Request (SRQ) event occurs on the VNA.

The port number can range from 5000 to 5099. The VNA will skip over 5025 as it is being used for the primary socket connection.

To detect an SRQ, your program sends the appropriate commands via the regular socket connection to set up for an SRQ event to occur the same sequence of commands as if you were sending them via GPIB. You write your program so that while your program is doing SCPI transactions on the standard socket connection, a second thread of execution in your program detects the SRQ on the control connection and responds to the event. When the SRQ event occurs, the VNA sends a SRQ +xxx/n message on the control connection (where /n is linefeed character, ASCII value 10 decimal). The xxx value in the SRQ +xxx/n string is the IEEE 488.2 status byte at the time the SRQ was generated. So listening for that on the control connection is how your program detects the event. If for your socket communication you're using a software API that provides for asynchronous communication via a callback mechanism (for example, if you're using Microsoft's winsock API, or their .NET Socket class as in the example program below), in that case your listener execution thread is created implicitly for you so your program doesn't have to create one explicitly.

Note: If this SCPI query is sent to the VNA via a SCPI parser other than a TCP/IP socket connection (for example, if sent via GPIB), the query is not applicable in that case and will return value of 0.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example</td>
<td>See example program</td>
</tr>
<tr>
<td>Return Type</td>
<td>Integer</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

SYSTem:COMMunicate:VISA:RDEVice:CLOSE <ID>
**Applicable Models:** N522xB, N523xB, N524xB, E5080A

(Write only) Closes the specified remote VISA session. VISA sessions should always be closed when you are finished communicating with the remote device. Use this command to close (end) each VISA session that was opened successfully using the OPEN command. If you have more than one open session, and need to close them all at the same time, it may be faster and easier to use the RESet command.

**Parameters**

- `<ID>` VISA session number (see SYST:COMM:VISA:RDEV:OPEN[?]).

**Examples**

```
SYST:COMM:VISA:RDEV:CLOS 1
system:communicate:visa:rdevice:close 2
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**SYSTem:COMMunicate:VISA:RDEVice:FIND? <VISA regex> [,<ADDRess|ALIas>]**

**Applicable Models:** All

(Read-only) Returns a comma separated list of either VISA address strings or aliases.

**Parameters**

- `<VISA regex>` (String) VISA regular expressions are expressions defined by the user to find devices that have been set up on the VISA interface. The following are examples of VISA regular expressions:

<table>
<thead>
<tr>
<th>Interface</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPIB</td>
<td>GPIB[0-9]*::?*INSTR</td>
</tr>
<tr>
<td>PXI</td>
<td>PXI?*INSTR</td>
</tr>
<tr>
<td>VXI</td>
<td>VXI?*INSTR</td>
</tr>
<tr>
<td>GPIB-VXI</td>
<td>GPIB-VXI!?*INSTR</td>
</tr>
<tr>
<td>GPIB and GPIB-VXI</td>
<td>GPIB?*INSTR</td>
</tr>
<tr>
<td>All VXI</td>
<td>?<em>VXI[0-9]</em>::?*INSTR</td>
</tr>
<tr>
<td>ASRL</td>
<td>ASRL[0-9]*::?*INSTR</td>
</tr>
<tr>
<td>All</td>
<td>?<em>INSTR or ?</em></td>
</tr>
</tbody>
</table>

Note that using "INSTR" in the VISA regular expression finds "instruments." To search all interfaces, use "?*".
<ADDRess|ALIas>  Optional. Determines whether addresses or aliases are returned.

**Note:** The list of aliases may have less or more entries than the list of addresses because not all addresses will have aliases, and one address can have more than one alias.

### Examples

```
SYST:COMM:VISA:RDEV:FIND? "?*", ADDR
system:communicate:visa:rdevice:find? '??*INSTR', alias
```

### Return Type

**Default** Addresses returned if no return-type specified

---

**SYSTem:COMMunicate:VISA:RDEVice:OPEN <addr>, <timeout>**

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

*Read-Write* Initiates a VISA pass-through session for a device. Immediately after successfully sending this command, send an OPEN? query to retrieve the unique session ID that is to be used whenever communicating with the device. Pass-through sessions can be closed by using the CLOSE command, the RESet command, or by properly shutting down the instrument or the analyzer application. Presetting the instrument will not close existing pass-through sessions.

**Note:** When opening a socket session (addresses of type: “TCP/IP[board]:host address::port::SOCKET”), you must use the appropriate VISA Address for the identifier argument. Using an alias to open a socket session is not currently supported. Aliases are allowed for all other types of supported sessions.

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;addr&gt;</td>
<td>VISA Address or alias name of the device to be controlled</td>
</tr>
<tr>
<td>&lt;timeout&gt;</td>
<td>The amount of time (in milliseconds) to wait for a response from the remote device after sending a command. A &quot;timeout&quot; error is displayed after this time has passed without a response.</td>
</tr>
</tbody>
</table>

### Examples

```
SYST:COMM:VISA:RDEV:OPEN 'TCPIP0::A-N5242A-10096::hislip1::INSTR', 1000
system:communicate:visa:rdevice:open 'MyAliasName', 7000
```

### Query Syntax

**SYSTem:COMMunicate:VISA:RDEVice:OPEN?**

Returns the VISA session identification number to be used when communicating with this device.

### Return Type

**Default** Numeric

**Not Applicable**
SYStem:COMMunicate:VISA:RDEVice:READ? <ID>

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

(Read-only) Returns data from the VISA pass-through device.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ID&gt;</td>
<td>VISA session number (see SYST:COMM:VISA:RDEV:OPEN[?]).</td>
</tr>
</tbody>
</table>

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:COMM:VISA:RDEV:READ? 1</td>
</tr>
</tbody>
</table>

**Return Type** String

**Default** Not Applicable

---

SYStem:COMMunicate:VISA:RDEVice:RESet

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

(Write-only) Closes all currently open VISA pass-through sessions. See also CLOSe to close only one session at a time.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:COMM:VISA:RDEV:RES</td>
</tr>
</tbody>
</table>

**Query Syntax** Not Applicable

**Default** Not Applicable

---

SYStem:COMMunicate:VISA:RDEVice:TIMeout <ID>, <timeout>

**Applicable Models:** All

(Read-Write) Sets or returns the timeout value (in milliseconds) for VISA pass-through commands for the specified VISA session ID.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ID&gt;</td>
<td>VISA session number that was returned with the OPEN? command.</td>
</tr>
<tr>
<td>&lt;timeout&gt;</td>
<td>The amount of time (in milliseconds) to wait for a response from the remote device after sending a command. A &quot;timeout&quot; error is displayed after this time has passed without a response.</td>
</tr>
</tbody>
</table>

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:COMM:VISA:RDEV:TIM 1,6000</td>
</tr>
<tr>
<td>system:communicate:visa:rdevice:timeout 3,6000</td>
</tr>
</tbody>
</table>
### Query Syntax

SYSTem:COMMunicate:VISA:RDEVice:TIMeout? <ID>

Returns the timeout value for the specified session ID.

**Return Type**

*Numeric*

**Default**

2000

### SYSTem:COMMunicate:VISA:RDEVice:WBINary <ID>,<data>

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

*(Write-only)* Sends data to a VISA pass-through device. This command requires a header that specifies the size of the data to be written. The header (described below) is not passed along to the device.

**Parameters**

- **<ID>** VISA session number (see `SYST:COMM:VISA:RDEV:OPEN[?]`).
- **<data>** Data to be sent to the VISA pass-through device. Use the following syntax:

  ```
  #<num_digits><byte_count><data_bytes><NL><END>
  ```

  - `<num_digits>` specifies how many digits are contained in `<byte_count>`.
  - `<byte_count>` specifies how many data bytes will follow in `<data_bytes>`.

**Examples**

SYSTem:COMMunicate:VISA:RDEVice:WBINary 1,#17ABC+XYZ<nl><end>

- # - always sent before data.
- 1 - specifies that the byte count is one digit (7).
- 7 - specifies the number of data bytes that will follow, not counting `<NL><END>`.
- ABC+XYZ - Data block
- `<nl><end>` - always sent at the end of block data.

The following example sends a line feed at the end.

SYST:COMM:VISA:RDEV:WBIN 1,#210SYST:PRES<EOL>

The `<EOL>` represents your linefeed character.

**Query Syntax**

Not Applicable
**SYSTem:COMMunicate:VISA:RDEVice:WRIte <ID>,<string>**

**Applicable Models:** N522xB, N523xB, N524xB, E5080A

*(Write-only)* Sends ASCII string data to the VISA pass-through device. If sending a command that returns data, follow with the `READ?` query.

**Parameters**

- **<ID>** VISA session number (see `SYST:COMM:VISA:RDEV:OPEN[?]`).
- **<string>** Commands to be sent to the VISA pass-through device.

**Examples**

```
SYST:COMM:VISA:RDEV:WRIT 1,'*IDN?'
```

```
SYSTEM:COMMunicate:visa:rdevice:write 2,'SYST:PRES'
```

**Query Syntax** Not Applicable

**Default** Not Applicable
SYSTem:CONFigure:MWAVe Commands

Controls and queries settings that affect Millimeter Wave measurements.

Millimeter Module Configuration
Banded Configuration Dialog

Broadband Configuration Dialog
SYSTem:CONFigure:MWAVe:CONF:ACTive <string>

**Applicable Models:** N5261A, N5262A, N5252A, N5292A

*(Read-Write)* Set and return the active configuration.

**Note:** Any changes to configuration settings are not active until the SYSTem:CONFigure:MWAVe:CONF:ACTive command is executed.

**Parameters**

<string> The name of the mmWave configuration to activate.

**Examples**

```
SYST:CONF:MWAV:CONF:ACT "N5291A Broadband"  'Set to broadband configuration
SYST:CONF:MWAV:CONF:ACT "WR10"  'Set to banded configuration
```

See Also

- N5251A Millimeter Module Configuration
- N5290A/N5291A Millimeter Module Configuration
**standard PNA configuration**

**Query Syntax**
SYSTem:CONFigure:MWAve:CONF:ACTive?

**Return Type**
String

**Default**
Not applicable

**SYSTem:CONFigure:MWAve:CONF:ACTive:CALibration:DATE?**

**Applicable Models:** N5293A, N5295A

*(Read-only)* Return the calibration date of the active configuration's test set. Three integers are returned (year, month, day). All three integers will be 0 if there is no calibration data for the active test set.

**Parameters**
None

**Examples**
system:configure:mwave:conf:active:calibration:date?

**Return Type**
Integer

**Default**
Not applicable

**SYSTem:CONFigure:MWAve:CONF:ACTive:CALibration:TIME?**

**Applicable Models:** N5293A, N5295A

*(Read-only)* Return the calibration time of the active configuration's test set. Three integers are returned (hour, minute, second). All three integers will be 0 if there is no calibration data for the active test set.

**Parameters**
None

**Examples**
system:configure:mwave:conf:active:calibration:time?

**Return Type**
Integer

**Default**
Not applicable

**SYSTem:CONFigure:MWAve:CONF:ACTive:MODE?**
Applicable Models: N5293A, N5295A

(Read-only) Return the model number of the active test set.

Parameters None
Examples
SYST:CONF:MWAV:CONF:ACT:MOD?
system:onfigure:mwave:conf:active:model?

Return Type String
Default Not applicable

SYSTem:CONFigure:MWAVe:CONF:ACTive:OPTion?

Applicable Models: N5293A, N5295A

(Read-only) Return the option number of the active test set.

Parameters None
Examples
SYST:CONF:MWAV:CONF:ACT:OPT?
system:onfigure:mwave:conf:active:option?

Return Type String
Default Not applicable

SYSTem:CONFigure:MWAVe:CONF:ACTive:PORT{1:4}:CALibration:DATE?

Applicable Models: N5293A, N5295A

(Read-only) Return the calibration date of the active configuration's port. Three integers are returned (year,month,day). All three integers will be 0 if there is no calibration data for the active port.

Parameters None
Examples
system:onfigure:mwave:conf:active:port1:calibration:date?

Return Type Integer
Default Not applicable

SYSTem:CONFigure:MWAVe:CONF:ACTive:PORT{1:4}:CALibration:TIME?
### Applicable Models: N5293A, N5295A

(Read-only) Return the calibration time of the active configuration's port. Three integers are returned (hour, minute, second). All three integers will be 0 if there is no calibration data for the active port.

**Parameters** None

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>system:onfigure:mwave:conf:active:port1:calibration:time?</td>
</tr>
</tbody>
</table>

**Return Type** Integer

**Default** Not applicable

---

### SYSTem:CONFigure:MWAve:CONF:ACTive:PORT{1:4}:MODei?

**Applicable Models: N5293A, N5295A**

(Read-only) Return the model number of the frequency extender module connected to the specified port number.

**Parameters** None

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>system:onfigure:mwave:conf:active:port1:model?</td>
</tr>
</tbody>
</table>

**Return Type** String

**Default** Not applicable

---

### SYSTem:CONFigure:MWAve:CONF:ACTive:PORT{1:4}:OPTiOn?

**Applicable Models: N5293A, N5295A**

(Read-only) Return the option number of the frequency extender module connected to the specified port number.

**Parameters** None

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>system:onfigure:mwave:conf:active:port1:option?</td>
</tr>
</tbody>
</table>

**Return Type** String

**Default** Not applicable
SYSTem:CONFigure:MWAVe:CONF:ACTive:PORT{1:4}:SERial?

Applicable Models: N5293A, N5295A

(Read-only) Return the serial number of the frequency extender module connected to the specified port number.

Parameters
None

Examples
system:configure:mwave:conf:active:port1:serial?

Return Type
String
Default Not applicable

SYSTem:CONFigure:MWAVe:CONF:ACTive:SERial?

Applicable Models: N5292A

(Read-only) Return the serial number of the test set.

Parameters
None

Examples
SYST:CONF:MWAV:CONF:ACT:SER?
system:configure:mwave:conf:active:serial?

Return Type
String
Default Not applicable

SYSTem:CONFigure:MWAVe:CONF:ADD <string>

Applicable Models: N5261A, N5262A, N5252A, N5292A

(Read-Write) Add a banded mmWave configuration.

Parameters
<string> The name of the mmWave configuration to add.

Examples
SYST:CONF:MWAV:CONF:ADD "WR10"
system:configure:mwave:conf:add "wr10"

Query Syntax
SYSTem:CONFigure:MWAVe:CONF:ADD?

Return Type
String
Default Not applicable
SYSTem:CONFigure:MWAVe:CONF:CATalog?

Applicable Models: N5261A, N5262A, N5252A, N5292A

(Read-only) Returns the list of mmWave configurations.

Parameters

Examples

SYST:CONF:MWA:CONF:CAT?

system:configure:mwave:config:catalog?

Return Type  Comma-delimited string.

Default  Not applicable

---

SYSTem:CONFigure:MWAVe:CONF:REMove <string>

Applicable Models: N5261A, N5262A, N5252A, N5292A

(Write-only) Remove a mmWave configuration.

Parameters

<string>  The name of the mmWave configuration to remove.

Examples

SYST:CONF:MWA:CONF:REM "WR10"

system:configure:mwave:conf:remove "wr10"

Return Type  Not applicable

Default  Not applicable

---

SYSTem:CONFigure:MWAVE:FREQuency:LO:MULTiplier <string>,<value>
**Applicable Models:** N5261A, N5262A, N5252A, N5292A

*(Read-Write)* Set and read the LO multiplier value for the specified configuration. The LO Frequency Range multiplied by this value equals the test port frequency. Learn more about frequency settings.

**Note:** Any changes to configuration settings are not active until the SYSTem:CONFigure:MWAVe:CONF:ACTive command is executed.

### Parameters
- **<string>** The name of the mmWave configuration.
- **<value>** Choose a value within the range of the analyzer.

### Examples
- `system:configure:mwave:frequency:lo:multiplier "WR10",8`

### Query Syntax
- `SYSTem:CONFigure:MWAVe:FREQuency:LO:MULTiplier? "WR10"`

### Return Type
- **Integer**
- **Default** 1

**SYSTem:CONFigure:MWAVe:FREQuency:LO:SOURce <string>,<source>**

**Applicable Models:** N5261A, N5262A, N5292A

*(Read-Write)* Set and read the LO source for the specified configuration. Learn more.

**Note:** Any changes to configuration settings are not active until the SYSTem:CONFigure:MWAVe:CONF:ACTive command is executed.

### Parameters
- **<string>** The name of the mmWave configuration.
- **<source>** The name of the LO source.

### Examples
- `SYST:CONF:MWAV:FREQ:LO:SOUR "WR10","LO_Source"
- `system:configure:mwave:frequency:lo:source "WR10","LO_Source"

### Query Syntax
- `SYSTem:CONFigure:MWAVe:FREQuency:LO:SOURce? "WR10"

### Return Type
- **String**
- **Default** Internal LO source

**SYSTem:CONFigure:MWAVe:FREQuency:LO:STARt? <string>**
(Read-only) Return the calculated LO start frequency using the LO multiplier for the specified configuration. Learn more about frequency settings.

**Parameters**

- `<string>`: The name of the mmWave configuration.

**Examples**

```

system:configure:mwave:frequency:lo:start? "WR10"
```

**Return Type**

- Double

**Default**

Dependent on the range of the analyzer

---

(Read-only) Return the calculated LO stop frequency using the LO multiplier for the specified configuration. Learn more about frequency settings.

**Parameters**

- `<string>`: The name of the mmWave configuration.

**Examples**

```

system:configure:mwave:frequency:lo:stop? "WR10"
```

**Return Type**

- Double

**Default**

Dependent on the range of the analyzer

---

SYSTem:CONFigure:MWAVe:FREQuency:RF:MULTiplier <string>,<value>
Applicable Models: N5261A, N5262A, N5252A, N5292A

(Read-Write) Set and read the RF multiplier value for the specified configuration. The RF Frequency Range multiplied by this value equals the test port frequency range. Learn more about frequency settings.

Note: Any changes to configuration settings are not active until the SYSTem:CONFigure:MWAVe:CONF:ACTive command is executed.

Parameters
- <string> The name of the mmWave configuration.
- <value> Choose a value within the range of the analyzer.

Examples
- system:configure:mwave:frequency:rf:multiplier "WR10", 6

Query Syntax
SYSTem:CONFigure:MWAVe:FREQuency:RF:MULTiplier? "WR10"

Return Type
Integer
Default 1

System:CONFigure:MWAVe:FREQuency:RF:SOURce <string>,<source>

Applicable Models: N5261A, N5262A, N5292A

(Read-Write) Set and read the RF source for the specified configuration. Learn more.

Note: Any changes to configuration settings are not active until the SYSTem:CONFigure:MWAVe:CONF:ACTive command is executed.

Parameters
- <string> The name of the mmWave configuration.
- <source> The name of the RF source.

Examples
- SYST:CONF:MWAV:FREQ:RF:SOUR "WR10", "RF_Source"
- system:configure:mwave:frequency:rf:source "WR10", "RF_Source"

Query Syntax
SYSTem:CONFigure:MWAVe:FREQuency:RF:SOURce? "WR10"

Return Type
String
Default Internal RF source

System:CONFigure:MWAVe:FREQuency:RF:STARt? <string>
Applicable Models: N5261A, N5262A, N5252A, N5292A

(Read-only) Return the calculated RF start frequency using the RF multiplier for the specified configuration. Learn more about frequency settings.

Parameters

<string> The name of the mmWave configuration.

Examples

system:conffigure:mwave:frequency:rf:start? "WR10"

Return Type Double

Default Dependent on the range of the analyzer

SYSTem:CONFigure:MWAVe:FREQuency:RF:STOP? <string>

Applicable Models: N5261A, N5262A, N5252A, N5292A

(Read-only) Return the calculated RF stop frequency using the RF multiplier for the specified configuration. Learn more about frequency settings.

Parameters

<string> The name of the mmWave configuration.

Examples

system:conffigure:mwave:frequency:rf:stop? "WR10"

Return Type Double

Default Dependent on the range of the analyzer

SYSTem:CONFigure:MWAVe:FREQuency:STARt <string>,<value>
Applicable Models: N5261A, N5262A, N5252A, N5292A

(Read-Write) Set and read the start frequency for the specified configuration. Learn more about frequency settings.

Note: Any changes to configuration settings are not active until the SYStem:CONFigure:MWAVe:CONF:ACTive command is executed.

Parameters
- <string> The name of the mmWave configuration.
- <value> The start frequency.

Examples
- SYST:CONF:MWAV:FREQ:STAR "WR10",75e9
- system:configure:mwave:frequency:start "WR10",75e9

Query Syntax
- SYSTem:CONFigure:MWAVe:FREQuency:STAR? "WR10"

Return Type
- Double
- Default Dependent on the range of the analyzer

SYSTem:CONFigure:MWAVe:FREQuency:STOP <string>,<value>

Applicable Models: N5261A, N5262A, N5252A, N5292A

(Read-Write) Set and read the stop frequency for the specified configuration. Learn more about frequency settings.

Note: Any changes to configuration settings are not active until the SYStem:CONFigure:MWAVe:CONF:ACTive command is executed.

Parameters
- <string> The name of the mmWave configuration.
- <value> The stop frequency.

Examples
- SYST:CONF:MWAV:FREQ:STOP "WR10",110e9
- system:configure:mwave:frequency:stop "WR10",110e9

Query Syntax
- SYSTem:CONFigure:MWAVe:FREQuency:STOP? "WR10"

Return Type
- Double
- Default Dependent on the range of the analyzer

SYSTem:CONFigure:MWAVe:SERial?
**Applicable Models:** N5290A, N5291A

*(Read-only)* Return the serial number of the mmWave system.

**Parameters** None

**Parameters** None

**Examples**

```
SYST:CONF:MWAV:SER?
```

```
System:configure:mwave:serial?
```

```
"M42147691962"
```

**Return Type** String

**Default** Not applicable

**SYSTem:CONFigure:MWAVe:TSET:ALC** `<string>,<bool>`

**Applicable Models:** N5261A, N5262A

*(Read-Write)* Enable or disable automatic power leveling for the specified configuration. Learn more about test set properties.

**Note:** Any changes to configuration settings are not active until the SYSTem:CONFigure:MWAVe:CONF:ACTive command is executed.

**Parameters**

- `<string>` The name of the mmWave configuration.
- `<bool>` Choose from:
  - 0 - **OFF** - Disable automatic power leveling control.
  - 1 - **ON** - Enable automatic power leveling control.

**Examples**

```
SYST:CONF:MWAV:TSET:ALC "WR10",1
```

```
System:configure:mwave:tset:alc "WR10",1
```

**Query Syntax** SYSTem:CONFigure:MWAVe:TSET:ALC? "WR10"

**Return Type** Boolean

**Default** 1

**SYSTem:CONFigure:MWAVe:TSET:CATalog?**
Applicable Models: N5261A, N5262A, N5292A

(Read-only) Returns the list of available test set names.

Parameters

Examples

```
SYST:CONF:MWAV:TSET:CAT?

system:configure:mwave:tset:catalog?
```

Return Type  Comma-delimited string.
Default    Not applicable

SYSTem:CONFigure:MWAVe:TSET:MIXer <string>,<bool>

Applicable Models: N5261A, N5262A, N5292A

(Read-Write) Enable or disable mmWave mixer mode to allow mixer testing using the specified configuration. Learn more about test set properties.

Note: Any changes to configuration settings are not active until the SYSTem:CONFigure:MWAVe:CONF:ACTive command is executed.

Parameters

```
<string>   The name of the mmWave configuration.
<bool>     Choose from:

0 - OFF - Disable mmWave mixer mode.
1 - ON  - Enable mmWave mixer mode.
```

Examples

```
SYST:CONF:MWAV:TSET:MIXer "WR10",0

system:configure:mwave:tset:mixer "WR10",0
```

Query Syntax

SYSTem:CONFigure:MWAVe:TSET:MIXer? "WR10"

Return Type  Boolean
Default    0

SYSTem:CONFigure:MWAVe:TSET:NAME <string>,<name>
Applicable Models: N5261A, N5262A, N5292A

(Read-Write) Set and read the test set name for the specified configuration.

Note: Any changes to configuration settings are not active until the SYStem:CONFigure:MWAVe:CONF:ACTive command is executed.

Setting the test set name may reset the test set properties to default values, which includes the following:

- Enabled Modules
- Mixer Mode
- IF Gain
- Route to Rear Panel
- ALC Enabled
- Power Limit
- Power Offset
- Power Slope

**Parameters**

- `<string>` The name of the mmWave configuration.
- `<name>` The name of the test set.

**Examples**

```
SYST:CONF:MWAV:TSET:NAME "WR10","N5262A"
```

**Query Syntax**

```
SYSTem:CONFigure:MWAVe:TSET:NAME? "WR10"
```

**Return Type** String

**Default** Not applicable

```
SYSTem:CONFigure:MWAVe:TSET:PORT<port> <configuration>,<val>
```
**Applicable Models:** N5261A, N5262A, N5292A

*(Read-Write)* This command enables or disables modules for individual ports on a particular configuration. If an N5292A test set is attached it will detect if a module is physically present and only allow this command on those ports.

**Note:** Any changes to configuration settings are not active until the SYStem:CONFigure:MWAVe:CONF:ACTive command is executed.

**Parameters**

- `<port>` Test set port number.
- `<configuration>` The name of the mmWave configuration. Use SYStem:CONFigure:MWAVe:CONF:CATalog? for a list of configuration names.
- `<val>` Enable/disable test set port. Choose from:
  - **0 - OFF** - Disable test set port.
  - **1 - ON** - Enable test set port.

**Examples**

```
SYST:CONF:MWAV:TSET:PORT1 "WR10",1
system:configure:mwave:tset:port1 "WR10",1
```

**Query Syntax**

SYStem:CONFigure:MWAVe:TSET:PORT:COUNT? <string>

**Applicable Models:** N5261A, N5262A, N5292A

*(Read-only)* Return the number of ports on a test set. Only the test set ports having frequency extenders attached are detected.

**Parameters**

- `<string>` The name of the mmWave configuration.

**Examples**

```
```

**Return Type**

- **Boolean**
- **Default** 0
SYSTem:CONFigure:MWAVe:TSET:POWer:LIMit <string>,<value>

Applicable Models: N5261A, N5262A, N5252A, N5292A

(Read-Write) Set and read the test set power limit for the specified configuration. Learn more about test set properties.

Note: Any changes to configuration settings are not active until the SYSTem:CONFigure:MWAVe:CONF:ACTive command is executed.

Parameters

<string> The name of the mmWave configuration.

<value> The power limit. Choose a value between -90 and +13 dBm.

Examples


Query Syntax

SYSTem:CONFigure:MWAVe:TSET:POWer:LIMit? "WR10"

Return Type Double

Default 11 dBm

---

SYSTem:CONFigure:MWAVe:TSET:POWer:OFFSet <string>,<value>

Applicable Models: N5261A, N5262A, N5252A, N5292A

(Read-Write) Set and read the test set power offset for the specified configuration. Learn more about test set properties.

Note: Any changes to configuration settings are not active until the SYSTem:CONFigure:MWAVe:CONF:ACTive command is executed.

Parameters

<string> The name of the mmWave configuration.

<value> The power offset. Choose a value between -30 and +30 dB.

Examples

SYST:CONF:MWAV:TSET:POW:OFFS "WR10",0
system:configure:mwave:tset:power:offset "WR10",0

Query Syntax

SYSTem:CONFigure:MWAVe:TSET:POWer:OFFSet? "WR10"

Return Type Double

Default 0 dB
SYSTem:CONFigure:MWAVe:TSET:POWer:SLOPe <string>,<value>

**Applicable Models:** N5261A, N5262A, N5252A, N5292A

*(Read-Write)* Set and read the test set power slope for the specified configuration. Learn more about test set properties.

**Note:** Any changes to configuration settings are not active until the SYSTem:CONFigure:MWAVe:CONF:ACTive command is executed.

**Parameters**
- `<string>` The name of the mmWave configuration.
- `<value>` The power slope. Choose a value between -2 dB/GHz and +2 dB/GHz.

**Examples**
- `SYST:CONF:MWAV:TSET:POW:SLOP "WR10",.113`
- `system:onfigure:mwave:tset:power:slope "WR10",.113`

**Query Syntax**
- `SYSTem:CONFigure:MWAVe:TSET:POWer:SLOpe? "WR10"`

**Return Type**
- Double
- **Default** 0.113 dB/GHz

---

SYSTem:CONFigure:MWAVe:TSET:RPANel <string>,<bool>

**Applicable Models:** N5261A, N5262A

*(Read-Write)* Set and read the status of the RF power routing to the rear panel "SW SRC OUT" connector for the specified configuration.

**Note:** Any changes to configuration settings are not active until the SYSTem:CONFigure:MWAVe:CONF:ACTive command is executed.

**Parameters**
- `<string>` The name of the mmWave configuration.
- `<bool>` Choose from:
  - **0 - OFF** - Disable RF power routing to rear panel.
  - **1 - ON** - Enable RF power routing to rear panel "SW SRC OUT" connector.

**Examples**
- `SYST:CONF:MWAV:TSET:RPAN "WR10",1`
- `system:onfigure:mwave:tset:rpanel "WR10",1`

**Query Syntax**
- `SYSTem:CONFigure:MWAVe:TSET:RPANel? "WR10"`
Return Type  Boolean
Default  1
External Device Commands

Configures and makes settings for an external device.

SYSTem:CONFigure:EDEVice:
| ADD
| CAT?
| DRIVer
| DTYPE
| EXISts?
| IOConfig
| IOENable
| LOAD
| REMove
| SAVE
| SMU
  | CHANnel
    | STATE
| STATE
| TOUT

| DC  More commands
| PMAR  More commands
| PULSe  More commands

| SOURce:
| DPP
| MODulation
  | CONTrol
    | :STATE
| TMODE
| TPORt

Click on a red keyword to view the command details.

See Also

- Learn about: Configure an External Source
- Learn about: Configure a PMAR Device
Configure an External Source

Configure a PMAR Device

SYST:PREF:ITEM:EDEV:DPOL - Determines whether external devices remain activated or are de-activated when the VNA is Preset or when a Instrument State is recalled.

Synchronizing the Analyzer and Controller

SCPI Command Tree

**SYSTem:CONFigure:EDEVice:ADD <name>**

**Applicable Models:** All

*(Write-only)* Adds an external device to the list of configured devices. This is the same as pressing New on the Select an external device dialog.

Upon creation, all settings on the new device are set to the defaults. The device is not active until set using **SYST:CON:EDev:STAT**

**Parameters**

- `<name>` String - Model and type of the external device.

To see a list of configured external devices, use **SYST:CONF:EDEV:CAT?**

**Examples**

- **SYST:CONF:EDEV:ADD "myDevice"**
- `system:configure:edevice:add "myDevice"`

**Query Syntax**

- Not Applicable

**Default**

- Not Applicable

**SYSTem:CONFigure:EDEVice:CAT?**
Applicable Models: All

(Read-only) Returns a list of names of all configured devices. These are devices that appear in the external devices dialog.

Use SENS:FOM:CAT? to report all active devices.

Use Source:CAT? to report all active sources.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>String of comma-separated devices. &quot;Device0:Driver0, Device1:Driver1&quot;</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

SYSTem:CONFigure:EDEVice:DRIVer <name>,<value>

Applicable Models: All

(Read-Write) Sets and returns the external device driver (model).

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;name&gt;</td>
</tr>
<tr>
<td>&lt;value&gt;</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:CONF:EDEV:DRIV &quot;myDevice&quot;,&quot;AGPM&quot;</td>
</tr>
<tr>
<td>system:configure:edevice:driver &quot;myDevice&quot;,&quot;AGESG&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:CONFigure:EDEVice:DRIVer? &lt;name&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;AGGeneric&quot;</td>
</tr>
</tbody>
</table>
SYStem:CONFigure:EDEVice:DTYPe <name>,<type>

Applicable Models: All

(Read-Write) Sets and returns the Device Type for the external device.

Parameters

- `<name>` String - Name of the device to modify.
- `<type>` String - Device type - not case sensitive. Choose from:
  - "None"
  - "Source" - external source
  - "Power Meter" - power meter
  - "DC Meter" - DC voltmeter
  - "DC Source" - DC power supply
  - "Pulse Generator" - external pulse generator
  - "SMU" - Source Measure Unit

Examples

SYST:CONF:EDEV:DTYP "myDevice","Power Meter"

System:configure:edevice:dtype "myDevice","Source"

Query Syntax

SYSTem:CONFigure:EDEVice:DTYPE? <name>

Return Type

- String

  Default

  Source

SYSTem:CONFigure:EDEVice:EXISts? <string>
### Applicable Models:

All

(Read-only) Returns whether the named device is present on the bus for which it is configured.

#### Parameters

- `<string>`: Name of the external device.

#### Example

```
SYST:CONF:EDEV:EXIS? "MyPowerMeter"
```

#### Return Type

Boolean

- **0**: The device is not in the collection or the device fails to respond and times out when communication is attempted.
- **1**: The device responds when communication is attempted.

**Default**: Not applicable

---

### SYStem:CONFigure:EDEVice:IOConfig <name>,<value>

**Applicable Models**: All

(Read-Write) Sets and return the configuration path for the specified external device.

#### Parameters

- `<name>`: String - Name of the device.
- `<value>`: String - Configuration path. Any valid VISA resource shown in the IO Configuration field of the external devices dialog, enclosed in quotes.

Use the `SYST:COMM:VISA:RDEV:FIND?` command to return IO devices that the visa subsystem can identify. This command will return addresses for devices found on GPIB, USB, or LAN.

#### Examples

```
SYST:CONF:EDEV:IOC "myDevice","GPIB0::13::INSTR"

SYST:CONF:EDEV:IOC "myDevice","TCPIP0::141.121.148.119::inst0::INSTR"

system:configure:edevice:ioconfig
"myDevice","USB0::10893::19457::MZ00703073::0::INSTR"
```

#### Query Syntax

```
SYSTem:CONFigure:EDEVice:IOConfig? <name>
```

#### Return Type

String

**Default**

"" Empty String
SYSTem:CONFigure:EDEVice:IOENable <name>,<value>

Applicable Models: All

(Read-Write) Enable or disable communication with an external device.

When disabled (OFF), the VNA will NOT attempt to connect to the external device regardless of the instrument state command (SYST:CONF:EDEV:STATe). Therefore, no errors will be produced if the device is not connected.

This command is useful for debugging and testing states when the external device is not connected. This command is unnecessary in ordinary operation (when the device is connected).

Parameters

- <name> String - Name of the device.
- <value> Boolean - Choose from:
  - OFF or 0 - Device communication disabled
  - ON or 1 - Device communication enabled

Examples

SYST:CONF:EDEV:IOEN "myDevice", ON
system:configure:edev:ioenable "myDevice", 0

Query Syntax

SYSTem:CONFigure:EDEVice:IOENable? <name>

Return Type

Boolean

Default

ON

SYSTem:CONFigure:EDEVice:LOAD <file>,<name>

Applicable Models: All

(Write-only) Recalls an external device configuration file from the VNA hard drive.

Currently, only DC Supply and DC Meter configuration files are supported. See more DC Device commands.


Parameters

- <file> String - Filename of the external device configuration file.

1. If <file> is not a full path, it is assumed to be a relative path based at "D:\Drivers".
2. The file specified using <file> cannot be the same file as "D:\Drivers\<name>.xml".
### SYStem:CONFigure:EDEVice:REMove <name>

**Applicable Models:** All  

(Write-only) Removes the specified device from the list of configured devices. If the device is a Source and both Active and I/O Enabled is checked (ON), then the RF power state is set to OFF. Learn more.

**Parameters**

- `<name>`  

**Examples**

```
SYST:CONF:EDEV:REM "myDevice"
```

**Query Syntax**  
Not Applicable

**Default**  
Not Applicable

### SYStem:CONFigure:EDEVice:SAVE <file>,<name>

**Applicable Models:** All

(Write-only) Saves an external device configuration file to the VNA hard drive.

Currently, only DC Supply and DC Meter configuration files are supported. See more DC Device commands.

Use `SYST:CONF:EDEV:LOAD` to recall a configuration file.

**Parameters**

- `<file>`  
  String - Filename of the external device configuration file.

- `<name>`  
  String - Name of the external device. Currently, only DC Supply and DC Meter configuration files are supported.

**Examples**

```
SYST:CONF:EDEV:SAVE "myDevice.xml","MyDCSupply"
```
SYSTem:CONFigure:EDEVice:SMU:CHANnel[1-4]:STATe <name>,<state>

**Applicable Models:** All

*(Read-Write)* Set and return the state of activation of the SMU device channel. Learn more about configuring an external SMU.

When `SYST:CONF:EDEV:IOEN` = ON, and this command is set to ON, the VNA will attempt communication with the external device.

Send this command AFTER sending other external device settings (especially `SYST:CONF:EDEV:DTYP`) to avoid communicating with the device before it has been fully configured.

See Also: `SYST:PREF:ITEM:EDEV:DPOL` - Determines whether external devices remain activated or are de-activated when the VNA is Preset or when a Instrument State is recalled.

**Parameters**

- `<name>`  String - Name of the SMU device.
- `<state>`  Boolean - Choose from:

  - **OFF** or **0** - SMU Device channel is NOT activated
  - **ON** or **1** - SMU Device channel is activated.

**Examples**

```
SYST:CONF:EDEV:SMU:CHAN2:STAT "myDevice", ON
system:configure:edevice:smu:channel2:state "myDevice", 0
```

**SYSTem:CONFigure:EDEVice:STATe <name>,<state>**

**Query Syntax**

Not Applicable

**Default**

Not Applicable

**Return Type**

Boolean

**Default**

OFF
**Applicable Models:** All

*(Read-Write)* Set and return the state of activation of the device. When `SYST:CONF:EDEV:IOEN = ON`, and this command is set to ON, the VNA will attempt communication with the external device.

Send this command AFTER sending other external device settings (especially `SYST:CONF:EDEV:DTYP`) to avoid communicating with the device before it has been fully configured.

See Also: `SYST:PREF:ITEM:EDEV:DPOL` - Determines whether external devices remain activated or are deactivated when the VNA is Preset or when a Instrument State is recalled.

**Parameters**

- `<name>` String - Name of the device.
- `<state>` Boolean - Choose from:
  - **OFF** or 0 - Device is NOT activated
  - **ON** or 1 - Device is activated.

**Examples**

```plaintext
SYST:CONF:EDEV:STAT "myDevice", ON
system:configure:edevice:state "myDevice", 0
```

**Query Syntax**

SYSTem:CONFigure:EDEVice:STATe? `<name>`

**Return Type**

Boolean

**Default**

OFF - When configured using the front panel user interface, the device is ON (activated) by default.

---

**SYSTem:CONFigure:EDEVice:TOUT `<name>`,<value>**

**Applicable Models:** All

*(Read-Write)* Set and return the time out value for the specified external device. This is the time allowed for communication with the device before an error is generated.

**Parameters**

- `<name>` String - Name of the device.
- `<value>` Time out value in seconds.

**Examples**

```plaintext
SYST:CONF:EDEV:TOUT "myDevice",2
system:configure:edevice:tout "myDevice",5
```

**Query Syntax**

SYSTem:CONFigure:EDEVice:TOUT? `<name>`

**Return Type**

Numeric

5042
SYSTem:CONFigure:EDEVice:SOURce:DPP <name>,<value>

**Applicable Models:** All

(Read-Write) Sets and returns the amount of time the VNA should wait after for an external source to settle before making a measurement at each data point. This setting applies to all channels that use this external source.

**Parameters**
- **<name>** String - Name of the device.
- **<value>** Dwell time in seconds.

**Examples**

```
SYST:CONF:EDEV:SOUR:DPP "myDevice",2
system:configure:edevice:source:dpp "myDevice",1
```

**Query Syntax**

SYSTem:CONFigure:EDEVice:SOURce:DPP? <name>

**Return Type** Numeric

**Default** 3.114 e-3

---

SYSTem:CONFigure:EDEVice:SOURce:MODulation:CONTrol:STATe <name>,<state>

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, E5080A

(Read-Write) Sets and reads the state of the modulation control. Modulation control must be ON to control the modulation of an external source.

**Parameters**
- **<name>** String - Name of the device.
- **<state>** ON (or 1) Enable control of external modulation.
- **OFF** (or 0) Disable control of external modulation.

**Examples**

```
SYST:CONF:EDEV:SOUR:MOD:CONT:STAT "qasmxg",1
```

**Query Syntax**

SYSTem:CONFigure:EDEVice:SOURce:MODulation:CONTrol:STATe? "qasmxg"

---

5043
SYS TEM:CONFigure:EDEVice:SOURce:TMODe <name>,<value>

Applicable Models: All

(Read-Write) Sets and returns the trigger mode for an external source. Learn more.

Parameters
- <name> String - Name of the device.
- <value> Trigger Mode. Choose from:
  - CW - Software CW mode
  - HW - Hardware list mode

Examples
SYST:CONF:EDEV:SOUR:TMOD "myDevice",CW
system:configure:edevice:source:tmode "myDevice",hw

Query Syntax
SYSTem:CONFigure:EDEVice:SOURce:TMODe? "myDevice"

Return Type Character
Default Depends on Source and VNA Model

SYS TEM:CONFigure:EDEVice:SOURce:TPORt <name>,<value>

Applicable Models: N522xB, N523xB, N524xB, M937xA, M9485A, E5080A

(Read-Write) Sets and returns the VNA port through which an external source is to be triggered.

Parameters
- <name> String - Name of the device.
- <value> Trigger Port. Choose from aux1 or aux2

Examples
system:configure:edevice:source:tport "myDevice",aux1

Query Syntax
SYSTem:CONFigure:EDEVice:SOURce:TPORt? <name>

Return Type Character
Default aux1
SYSTem:CONF:EDEVice:DC Commands

Configures external SMU, DC Meter, and DC Source properties.

```plaintext
SYST:CONF:EDEVice:DC
  | COMMand
  |   | EXIT
  |   | INIT
  |   | LIMit
  |   | POINt
  |   |   | SET
  |   | SWEep
  |       | ABORt
  |       | AFTer
  |       | BEFore
  |   | CORRection
  |   | DPOint
  |   | DSWeep
  |   | LIMit
  |       | CURRent
  |       | VOLTage
  |   | MAX
  |       | [:STATe]
  |       | VALue
  |   | MIN
  |       | [:STATe]
  |       | VALue
  |   | OFFSet
  |   | QUERy
  |       | ERRor
  |       | ID
  |       | SCALe
  |       | TYPE
```
See Also

- All SYST:CONF:EDEV commands
- SOURce:DC commands (make DC sweep settings)
- Learn about: Configure an External DC Device
- Learn about Configure an External Device
- SYST:PREF:ITEM:EDEV:DPOL - Determines whether External Devices remain activated or are de-activated when the VNA is Preset or when a Instrument State is recalled.
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**SYSTem:CONFigure:EDEVice:DC:COMMand:EXIT <name>,<value>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets and returns the Disable I/O command for an external DC Source and an external DC Meter.

**Parameters**

- `<name>`  
  String - Name of the device.
- `<value>`  
  String - The SCPI command used to disable the DC Source and DC Meter.

**Examples**

```
SYST:CONF:EDEV:DC:COMM:EXIT "myDCDevice","OUTP OFF"

system:configure:edevice:dc:command:exit "myDCDevice","OUTP OFF"
```

**Query Syntax**  
SYSTem:CONFigure:EDEVice:DC:COMMan:EXIT? <name>

**Return Type**  
String

**Default**  
"" Empty String

**SYSTem:CONFigure:EDEVice:DC:COMMan:INIT <name>,<value>**
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets and returns the **Enable I/O command** for an external DC Source and an external DC Meter.

**Parameters**

- `<name>` String - Name of the device.
- `<value>` String - The SCPI command used to enable the DC Source and DC Meter.

**Examples**

```
SYST:CONF:EDEV:DC:COMM:INIT "myDCDevice","OUTP ON"
system:configure:edevice:dc:command:init "myDCDevice","OUTP ON"
```

**Query Syntax**

`SYSTem:CONFigure:EDEVice:DC:COMMand:INIT? <name>`

**Return Type**

String

**Default**

" " Empty String

---

SYSTem:CONFigure:EDEVice:DC:COMMand:LIMit <name>, <cmd>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Sets and returns a user-defined command string that is used to set the DC limit of the external DC source. The actual limit value is set using either the `SYST:CONF:EDEV:DC:LIM:VOLT` (voltage) or `SYST:CONF:EDEV:DC:LIM:CURR` (current). The limit command is sent to the external DC source at the beginning of a sweep for each channel. The firmware automatically selects the current or voltage limit value depending on the external DC source type.

**Parameters**

- `<name>` String - Name of the device.
- `<cmd>` String - User-defined command name: Include `{%f}` in the command string which will be substituted by the actual limit value.

**Examples**

```
system:configure:edevice:dc:command:limit "myDCDevice","Limit Command {%f}"
```

**Query Syntax**

`SYSTem:CONFigure:EDEVice:DC:COMMand:LIMit? <name>`

**Return Type**

String

**Default**

Not Applicable

---

SYSTem:CONFigure:EDEVice:DC:COMMand:POINt:SET <name>,<value>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the Point Read Commands for an external DC Meter or Point Set Commands for an external DC Source.

Parameters
  <name>  String - Name of the device.
  <value> String -

For DC Source, sets the Point Set Commands. Use {%f} to specify a double value and {%d} to specify an integer.

For DC Meter, sets the Point Read Commands (for example, meas:volt?).

Examples
SYST:CONF:EDEV:DC:COMM:POIN:SET "myDCDevice","sour1:volt {%f}"


Query Syntax
SYSTem:CONFigure:EDEVice:DC:COMManD:POINt:SET? <name>

Return Type  String
  Default  " " Empty String

SYSTem:CONFigure:EDEVice:DC:COMManD:SWEep:ABORt <name>,<value>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the Abort Sweep command for an external DC Source and an external DC Meter.

Parameters
  <name>  String - Name of the device.
  <value> String - The SCPI command used to abort or reset the DC Source or DC Meter.

Examples
SYST:CONF:EDEV:DC:COMM:SWE:ABOR "myDCDevice","ABORt"

system:configure:edevice:dc:command:sweep:abort "myDCDevice","ABORt"

Query Syntax
SYSTem:CONFigure:EDEVice:DC:COMManD:SWEep:ABORt? <name>

Return Type  String
  Default  " " Empty String

5048
SYSTem:CONFigure:EDEVice:DC:COMManD:SWEep:AFTer <name>,<value>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the After Sweep command for an external DC Source and an external DC Meter.

Parameters
- <name> String - Name of the device.
- <value> String - The SCPI command to be sent at the end of a sweep.

Examples

"myDCDevice","OUTP OFF"

Query Syntax
SYSTem:CONFigure:EDEVice:DC:COMManD:SWEep:AFTer? <name>

Return Type String
Default " " Empty String

SYSTem:CONFigure:EDEVice:DC:COMManD:SWEep:BEFore <name>,<value>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the Before Sweep command for an external DC Source and an external DC Meter.

Parameters
- <name> String - Name of the device.
- <value> String - The SCPI command to be sent at the beginning of a sweep.

Examples

"myDCDevice","OUTP ON"

Query Syntax
SYSTem:CONFigure:EDEVice:DC:COMManD:SWEep:BEFore? <name>

Return Type String
Default " " Empty String

SYSTem:CONFigure:EDEVice:DC:CORRection <name>,<value>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the correction ON/OFF state for a DC Meter and a DC Source.

Parameters
- `<name>`  String - Name of the device.
- `<value>` Correction ON/OFF state. Choose from:
  - **ON or 1** - Turn Correction ON
  - **OFF or 0** - Turn Correction OFF

Examples

```
SYST:CONF:EDEV:DC:CORR "myDCDevice",1
system:configure:edevice:dc:correction "myDCDevice",OFF
```

Query Syntax
SYSTem:CONFigure:EDEVice:DC:CORRection? <name>

Return Type
Boolean
Default OFF

SYSTem:CONFigure:EDEVice:DC:DPOint <name>,<value>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the "Dwell Before/After Point" value for an external DC Device which can be configured as either a DC Meter or a DC Source.

Parameters
- `<name>`  String - Name of the device.
- `<value>` For DC Meter, the dwell time (in seconds) before making a data point measurement.
  For DC Source, the dwell time (in seconds) after making a data point setting.

Examples

```
SYST:CONF:EDEV:DC:DPO "myDCDevice",10e-3
system:configure:edevice:dc:dpoint "myDCDevice",.01
```

Query Syntax
SYSTem:CONFigure:EDEVice:DC:DPOint? <name>

Return Type
Numeric
Default 3 milliseconds

SYSTem:CONFigure:EDEVice:DC:DSWeep <name>,<value>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the "Dwell Before Sweep" value for an external DC Device which can be configured as either a DC Meter or a DC Source.

Parameters
- <name> String - Name of the device.
- <value> The dwell time (in seconds) before making a new sweep.

Examples

```
SYST:CONF:EDEV:DC:DSW "myDCDevice",10e-3
system:configure:edevice:dc:dsweep "myDCDevice",.01
```

Query Syntax
SYSTem:CONFigure:EDEVice:DC:DSWeep? <name>

Return Type
Numeric
Default 1 millisecond

---

SYSTem:CONFigure:EDEVice:DC:LIMit:CURRent <name>, <value>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the maximum output current value of the external DC Source. This command supports Keysight B2900A and N6700 series devices only.

Parameters
- <name> String - Name of the device.
- <value> Current limit value.

Examples

```
SYST:CONF:EDEV:DC:LIM:CURR "myDCDevice",4
```

Query Syntax
SYSTem:CONFigure:EDEVice:DC:LIMit:CURRent? <name>

Return Type
Double
Default 0

---

SYSTem:CONFigure:EDEVice:DC:LIMit:VOLTage <name>, <value>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the maximum output voltage value of the external DC Source. This command supports Keysight B2900A and N6700 series devices only.

Parameters
- `<name>`: String - Name of the device.
- `<value>`: Voltage limit value.

Examples

Query Syntax
- `SYSTem:CONFigure:EDEVice:DC:LIMit:VOLTage? <name>`

Return Type: Double
- Default: 0

SYSTem:CONFigure:EDEVice:DC:MAX[:STATe] <name>,<value>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the "Define Max As" ON/OFF state for an external DC Source.

Parameters
- `<name>`: String - Name of the device.
- `<value>`: "Define Max As" ON/OFF state. Choose from:
  - **ON or 1** - Turn "Define Max As" ON
  - **OFF or 0** - Turn "Define Max As" OFF

Examples
- `SYST:CONF:EDEV:DC:MAX "myDCDevice",1`
- `system:configure:edevice:dc:max:state "myDCDevice",OFF`

Query Syntax
- `SYSTem:CONFigure:EDEVice:DC:MAX:STATe? <name>`

Return Type: Boolean
- Default: ON

SYSTem:CONFigure:EDEVice:DC:MAX:VALue name, value
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the "Define Max As" value for an external DC Source.

Parameters

- `<name>` String - Name of the device.
- `<value>` Maximum value for the external DC Source (in volts).

Examples

```
system:configure:edevice:dc:max:value "myDCDevice",10
```

Query Syntax

```
SYSTem:CONFigure:EDEVice:DC:MAX:VALue? <name>
```

Return Type: Double

Default: 10

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the "Define Min As" ON/OFF state for an external DC Source.

Parameters

- `<name>` String - Name of the device.
- `<value>` "Define Min As" ON/OFF state. Choose from:
  - **ON or 1** - Turn "Define Min As" ON
  - **OFF or 0** - Turn "Define Min As" OFF

Examples

```
SYST:CONF:EDEV:DC:MIN "myDCDevice",1
system:configure:edevice:dc:min:state "myDCDevice",OFF
```

Query Syntax

```
SYSTem:CONFigure:EDEVice:DC:MIN:STATe? <name>
```

Return Type: Boolean

Default: ON

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the "Define Min As" value for an external DC Source.

Parameters

- `<name>` String - Name of the device.
- `<value>` Maximum value for the external DC Source (in volts).

Examples

```
SYST:CONF:EDEV:DC:MIN:VALue <name>, <value>
```

5053
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the "Define Min As" value for an external DC Source.

**Parameters**
- `<name>` String - Name of the device.
- `<value>` Minimum value for the external DC Source (in volts).

**Examples**
```
SYST:CONF:EDEV:DC:MIN:VAL "myDCDevice",-10
system:configure:edevice:dc:min:value "myDCDevice",-10
```

**Query Syntax**
```
SYSTem:CONFigure:EDEVice:DC:MIN:VALue? <name>
```

**Return Type** Double

**Default** -10

---

**SYSTem:CONFigure:EDEVice:DC:OFFSet <name>,<value>**

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the offset correction value for an external DC Device which can be configured as either a DC Meter or a DC Source.

**Parameters**
- `<name>` String - Name of the device.
- `<value>` DC offset value.

The VNA will display readings from a DC Meter as:
```
Display = (Meas'd value - Offset) * Scale
```

The VNA will adjust the output from a DC Source as:
```
Output = (Set value - Offset) * Scale
```

**Examples**
```
SYST:CONF:EDEV:DC:OFFS "myDCDevice",4
system:configure:edevice:dc:offset "myDCDevice",1.25
```

**Query Syntax**
```
SYSTem:CONFigure:EDEVice:DC:OFFSet? <name>
```

**Return Type** Numeric

**Default** 0

---

**SYSTem:CONFigure:EDEVice:DC:QUERy:ERRor <name>,<value>**
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the Error Query command for an external DC Source and an external DC Meter.

Parameters
- `<name>` String - Name of the device.
- `<value>` String - The SCPI command for returning DC Source and DC Meter errors.

Examples

Query Syntax
- `SYSTem:CONFigure:EDEVice:DC:QUERy:ERRor? <name>`

Return Type
- String
- Default "SYST:ERR?"

---

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the ID Query command for an external DC Source and an external DC Meter.

Parameters
- `<name>` String - Name of the device.
- `<value>` String - The SCPI command for returning DC Source and DC Meter ID string.

Examples
- `SYST:CONF:EDEV:DC:QUER:ID "myDCDevice","*IDN?"
- `system:configure:edevice:dc:query:error "myDCDevice","*IDN?"`

Query Syntax
- `SYSTem:CONFigure:EDEVice:DC:QUERy:ID? <name>`

Return Type
- String
- Default "*IDN?"

---

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

SYSTem:CONFigure:EDEVice:DC:SCALe <name>,<value>
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the scale correction value for an external DC Device which can be configured as either a DC Meter or a DC Source.

Parameters

- `<name>`: String - Name of the device.
- `<value>`: DC Scale value.

The VNA will display readings from a DC Meter as:

\[ \text{Display} = (\text{Meas'd value} - \text{Offset}) \times \text{Scale} \]

The VNA will adjust the output from a DC Source as:

\[ \text{Output} = (\text{Set value} - \text{Offset}) \times \text{Scale} \]

Examples

- `SYST:CONFigure:EDEVice:DC:SCALe "myDCDevice",1.2`
- `system:configure:edevice:dc:scale "myDCDevice",.5`

Query Syntax

`SYSTem:CONFigure:EDEVice:DC:SCALe? <name>`

Return Type

- Numeric
- Default: 1

---

`SYSTem:CONFigure:EDEVice:DC:TYPE <name>,<value>`

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and returns the DC Type for an external DC Device which can be configured as either a DC Meter or a DC Source. This setting is used as the units for display on the VNA X-axis.

Parameters

- `<name>`: String - Name of the device.
- `<value>`: DC type. Choose from:

  "dBm", "A", "V", "W", "K", "F", "C"

Examples

- `SYST:CONFigure:EDEVice:DC:TYPE "myDCDevice","A"
- `system:configure:edevice:dc:type "myDCDevice","W"

Query Syntax

`SYSTem:CONFigure:EDEVice:DC:TYPE? <name>`

Return Type

- String
- Default: "V"
SYST:CONF:EDEVice:PMAR Commands

Configures and makes settings for an external Power Meter as Receiver.

```
SYSTem:CONFigure:EDEVice:PMAR
 | CALibrate
 | FLIMit
 | FMAXimum
 | FMINimum
 | CFACtors
 | STATe
 | READing:
 | COUNT
 | NTOLerance
 | SENSor
 | CATalog?
 | TABLe:
 | CFAC:
 | DATA
 | FREQuency
 | LOSS:
 | DATA
 | FREQuency
 | STATe
 | RFACtor
 | UNCertainty
 | CATalog?
 | FILE
 | MODel
 | PLEVEL?
 | READ?
 | ZERO
```

Click on a keyword to view the command details.

**See Also**

- Learn about: Configure a Power Meter As Receiver
- See root SYST:CONF:EDEV commands
Learn about Configure and External Device

SYST:PREF:ITEM:EDEV:DPOL - Determines whether External Devices remain activated or are de-activated when the VNA is Preset or when a Instrument State is recalled.

Synchronizing the Analyzer and Controller

SCPI Command Tree

SYSTem:CONFigure:EDEVice:PMAR:CALibrate <name>

Applicable Models: All

(Write-only) Performs a calibration of the power sensor. Calibration usually involves connecting the power sensor to the meter's 1 mW reference.

- Keysight P-Series sensors have an internal reference so you can calibrate them using this command without connecting to the meters reference port.
- Keysight USB power sensors do not require calibrating.
- For other sensors, refer to the documentation to determine if it has calibration capability.

This command is always synchronous, so *OPC? is the only way to determine that the operation is complete. Set an I/O timeout of at least 20 seconds.

Parameters

- <name> String - Name of the power meter.

Examples

SYST:CONF:EDEV:PMAR:CAL "myDevice"

system:configure:edevice:pmar:calibrate "myDevice"

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CONFigure:EDEVice:PMAR:FLIMit <name>,<value>
Applicable Models: All

(Read-Write) Enable or disable the power meter min and max frequencies.

Parameters

- `<name>` String - Name of the power meter.
- `<value>` Boolean. State of min and max frequency. Choose from:
  - **OFF** or 0 - Min and max frequencies disabled.
  - **ON** or 1 - Min and max frequencies enabled.

Examples

```
SYST:CONF:EDEV:PMAR:FLIM "myDevice", 0
system:configure:edevice:pmar:flimit "myDevice", ON
```

See example program

Query Syntax

SYSTem:CONFigure:EDEVice:PMAR:FLIMit? <name>

Return Type

Boolean

Default

OFF

SYSTem:CONFigure:EDEVice:PMAR:FMAXimum <name>,<value>

Applicable Models: All

(Read-Write) Set and return the maximum frequency of the power meter.

Parameters

- `<name>` String - Name of the power meter.
- `<value>` Numeric - Max frequency in Hz.

Examples

```
SYST:CONF:EDEV:PMAR:FMAX "myDevice", 1e10
system:configure:edevice:pmar:fmaximum "myDevice", 3e9
```

See example program

Query Syntax

SYSTem:CONFigure:EDEVice:PMAR:FMAXimum? <name>

Return Type

Numeric

Default

Not Applicable

SYSTem:CONFigure:EDEVice:PMAR:FMINimum <name>,<value>
Applicable Models: All

(Read-Write) Set and return the minimum frequency of the power meter.

**Parameters**
- `<name>` String - Name of the power meter.
- `<value>` Numeric - Min frequency in Hz.

**Examples**
```
SYST:CONF:EDEV:PMAR:FMIN "myDevice", 1e10

system:configure:edevice:pmar:fminimum "myDevice", 3e9
```

See example program

**Query Syntax**
```
SYSTem:CONFigure:EDEVice:PMAR:FMAXimum? <name>
```

**Return Type** Numeric

**Default** Not Applicable

---

SYSTem:CONFigure:EDEVice:PMAR:CFACtors:STATe <name> <bool>

Applicable Models: All

(Read-Write) Enables/disables use of internal calibration factors for power sensors with built-in calibration factors and reads the current state.

**Parameters**
- `<name>` String - Name of the power meter.
- `<bool>` Choose from:
  - 0 - OFF - Disables the use of internal calibration factors.
  - 1 - ON - Enables the use of internal calibration factors.

Learn about these settings.

**Examples**
```
' This example script demonstrates the SCPI set/get of the "Use Internal Cal Factors" property
' for an existing PMAR named 'MyPMAR'.

Option Explicit

dim app

Set app = CreateObject("AgilentPNA835x.Application")

dim scpi
```
Set scpi = app.ScpiStringParser
Dim opcReply
opcReply = scpi.Parse("SYST:PRES;*OPC?")
opcReply = scpi.Parse("SENS1:SWE:MODE HOLD;*OPC?")
scpi.Parse "SENS1:FREQ: CW 1E9"
scpi.Parse "SENS1:SWE: TYPE CW"
scpi.Parse "SENS1:SWE: POIN 3"
' Activate the PMAR and change the default trace to measure that PMAR connected to port 3
scpi.Parse "SYST:CONF:EDEV:STAT 'MyPMAR', ON"
scpi.Parse "CALC1:PAR:SEL 'CH1_S11_1'"
scpi.Parse "CALC1:PAR:MOD:EXT 'MyPMAR, 3'"
scpi.Parse "CALC1:FUNC:TYPE MEAN"
' Disable use of the sensor's internal cal factors, take a sweep and report the Mean
scpi.Parse "SYST:CONF:EDEV:IOENable 'MyPMAR', ON"
opcReply = scpi.Parse("SENS1:SWE:MODE SING;*OPC?")
' Enable use of the sensor's internal cal factors, take another sweep and report the Mean again
scpi.Parse "SYST:CONF:EDEV:IOENable 'MyPMAR', OFF"
scpi.Parse "SYST:CONF:EDEV:PMAR:CFAC:STAT 'MyPMAR', ON"
scpi.Parse "SYST:CONF:EDEV:IOENable 'MyPMAR', ON"
opcReply = scpi.Parse("SENS1:SWE:MODE SING;*OPC?")

Query Syntax  SYSTem:CONFigure:EDEVice:PMAR:CFACtors:STATe?
SYSTem:CONFigure:EDEVice:PMAR:READing:COUNt <name>,<value>

**Applicable Models:** All

*(Read-Write)* This command, along with SYST:CONF:EDEV:PMAR:READ:NTOL, allows for settling of the power sensor READINGS.

Set and return the maximum number of power readings that are taken at each stimulus point to allow for measurement settling. Each reading is averaged with the previous readings at that stimulus point.

When this average meets the Average:NTOLerance value or this number of readings has been made, the average is returned as the valid reading.

**Parameters**

- `<name>`: String - Name of the power meter.
- `<value>`: Number of readings. Choose a value between 1 and 1000

**Examples**

```
```

See example program

**Query Syntax**

SYSTem:CONFigure:EDEVice:PMAR:READing:COUNt? <name>

**Return Type** Numeric

**Default** 3

SYSTem:CONFigure:EDEVice:PMAR:READing:NTOLerance <name>,<value>
Applicable Models: All

(Read-Write) This command, along with SYST:CONF:EDEV:PMAR:READ:COUN, allows for settling of the power sensor READINGS.

Each power reading is averaged with the previous readings at each stimulus point. When the average meets this nominal tolerance value, or the max number of readings has been made, the average is returned as the valid reading.

Parameters

- `<name>` String - Name of the power meter.
- `<value>` Power measurement settling tolerance value in dB. Choose any number between 0 and 5.

Examples

```plaintext
system:configure:edevice:pmar:reading:ntolerance "myDevice",.01
```

See example program

Query Syntax

SYSTem:CONFigure:EDEVice:PMAR:READing:NTOLerance? <name>

Return Type

Numeric

Default

.05

SYSTem:CONFigure:EDEVice:PMAR:SENSor <name>,<value>

Applicable Models: All

(Read-Write) Sets and returns the power sensor channel (1 or 2) to be used. This performs the same function as the **Use this sensor only** checkbox.

Parameters

- `<name>` String - Name of the power meter.
- `<value>` Power Meter channel.

Examples

```plaintext
SYST:CONF:EDEV:PMAR:SENS "myDevice",2
system:configure:edevice:pmar:sensor "myDevice",1
```

See example program

Query Syntax

SYSTem:CONFigure:EDEVice:PMAR:SENSor? <name>
SYStem:CONFigure:EDEVice:PMAR:SENSor:CATalog? <name>

Applicable Models: All

(Read-only) Returns the power sensor channel assignment of the specified power meter.

Parameters

=name> String - Name of the power meter.

Examples


Return Type Numeric

Default 1

---

SYSTem:CONFigure:EDEVice:PMAR:TABLE:CFAC:DATA <name>,<value>[,value]

Applicable Models: All

(Read-Write) Sets and returns the cal factor data for the power sensor.

Parameters

=<name> String - Name of the power meter.

<value>[,value] Cal factor data in percent. For each frequency used with
SYST:CONF:EDEV:PMAR:TABLE:CFAC:FREQ, enter a cal factor number
between 1 and 100.

Examples


97,97,97

See example program

Query Syntax

SYSTem:CONFigure:EDEVice:PMAR:TABLE:CFAC:DATA? <name>

Return Type Numeric - one number per table segment.

Default Not Applicable

---

SYSTem:CONFigure:EDEVice:PMAR:TABLE:CFAC:FREQuency <name>,<value>[,value]
Applicable Models: All

(Read-Write) Sets and returns the cal factor frequencies for the power sensor.

**Parameters**

- `<name>` String - Name of the power meter.
- `<value>[,value]` Cal factor frequencies in Hz.

**Examples**

```plaintext
SYST:CONF:EDEV:PMAR:TABL:CFAC:FREQ "myDevice", 1e7,1e8,1e9
system:configure:edevice:pmar:table:cfac:frequency "myDevice", 5e7,5e8,5e9
```

See example program

**Query Syntax**

SYSTem:CONFigure:EDEVice:PMAR:TABLe:CFAC:FREQuency?<name>

**Return Type**

Numeric - one number per table segment.

**Default**

Not Applicable

---

SYSTem:CONFigure:EDEVice:PMAR:TABLe:LOSS:DATA <name>,<value>[,value]

Applicable Models: All

(Read-Write) Sets and returns the power loss data for the power sensor.

Each table can contain up to 9999 segments. Values can also be loaded using the Characterize Adapter macro.

**Parameters**

- `<name>` String - Name of the power meter.
- `<value>[,value]` Loss data in dB. POSITIVE values in dB are interpreted as LOSS. To compensate for gain, use negative values.

For each frequency used with SYST:CONF:EDEV:PMAR:TABL:LOSS:FREQ, enter a cal factor number between 1 and 100.

**Examples**

```plaintext
SYST:CONF:EDEV:PMAR:TABL:LOSS:DATA "myDevice",.01,.02,.03
system:configure:edevice:pmar:table:loss:data "myDevice",.04,.05,.06
```

See example program

**Query Syntax**

SYSTem:CONFigure:EDEVice:PMAR:TABLe:CFAC:DATA? <name>
**SYStem:CONFigure:EDEVice:PMAR:TABLe:LOSS:FREQuency** <name>,<value>[,value]

**Applicable Models:** All

*(Read-Write)* Sets and returns frequencies for the power loss data.

**Parameters**

- `<name>`: String - Name of the power meter.
- `<value>[,value]`: Power Loss frequencies in Hz.

**Examples**

```language
SYST:CONF:EDEV:PMAR:TABLE:LOSS:FREQ "myDevice",1e7,1e8,1e9

"myDevice",5e7,5e8,5e9
```

See example program

**Query Syntax**

SYSTem:CONFigure:EDEVice:PMAR:TABLe:LOSS:FREQuency? <name>

**Return Type** Numeric - one number per table segment.

**Default** Not Applicable

---

**SYStem:CONFigure:EDEVice:PMAR:TABLe:LOSS:STATe** <name>,<value>

**Applicable Models:** All

*(Read-Write)* Sets and returns whether to use the power loss table.

**Parameters**

- `<name>`: String - Name of the power meter.
- `<value>`: Boolean. State of the power loss table. Choose from:
  - OFF or 0 - Power loss table not used.
  - ON or 1 - Power loss table used.

**Examples**

```language

```

See example program

**Query Syntax**

SYSTem:CONFigure:EDEVice:PMAR:TABLe:LOSS:STATe? <name>
**SYSTem:CONFigure:EDEVice:PMAR:TABLE:RFACtor** <name>,<value>

**Applicable Models:** All

*(Read-Write)* Sets and returns the reference cal factor for the power sensor.

Note: If the sensor connected to the power meter contains cal factors in EPROM (such as the Keysight E-series power sensors), those will be the cal factors used. The reference cal factor value associated with this command, and any cal factors entered into the VNA for that sensor channel, will not be used.

**Parameters**

- **<name>** String - Name of the power meter.
- **<value>** Reference cal factor in percent. Choose any number between 1 and 150.

**Examples**

```
SYST:CONF:EDEV:PMAR:TABL:RFAC "myDevice", 1
system:configure:edevice:pmar:table:rfactor "myDevice", 1
```

[See example program]

**Query Syntax**

```
SYSTem:CONFigure:EDEVice:PMAR:TABLE:RFACtor? <name>
```

**Return Type** Numeric

**Default** 100

---

**SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:CATalog?**

**Applicable Models:** N522xB, N523xB, N524xB (requires Option S93015A)

*(Read-only)* Returns a list of available power meters that have power uncertainty.

**Parameters**

- **<name>** String - Name of the device used as power meter which has uncertainty data available from the external device list.

**Examples**

```
SYST:CONF:EDEV:PMAR:UNC:CAT?
system:configure:edevice:pmar:uncertainty:catalog?
```

**Return Type** Comma separated string

**Default** Not Applicable
SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:FILE <name>,<FilePath>

**Applicable Models:** N522xB, N523xB, N524xB (requires Option S93015A)

*(Read-Write)* Sets and returns a custom model uncertainty file containing all of the power meter uncertainty properties. When this command is executed, the model name is automatically set to "CustomFile".

**Parameters**
- `<name>`: String - Name of the device used as the power meter.
- `<FilePath>`: Full path to the custom file.

**Examples**
```
SYST:CONF:EDEV:PMAR:UNC:FILE
"myDevice","C:\U8485A_MY55140018.dat"
```
```
system:configure:edevice:pmar:uncertainty:file
"myDevice","C:\U8485A_MY55140018.dat"
```

**Query Syntax**
SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:FILE? <name>

**Return Type**
String

**Default**
Not Applicable

---

SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:MODel <name>,<PwrMtrModel>

**Applicable Models:** N522xB, N523xB, N524xB (requires Option S93015A)

*(Read-Write)* Sets and returns the name assigned to a specific power meter model among those available for uncertainty (see SYSTem:CONFigure:EDEVice:PMAR:UNC:CAT?).

**Parameters**
- `<name>`: String - Name of the device used as the power meter.
- `<PwrMtrModel>`: String - Specific power meter model.

**Examples**
```
SYST:CONF:EDEV:PMAR:UNC:MOD "myPowerMeter","N8488A"
```
```
"myPowerMeter","N8488A"
```

**Query Syntax**
SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:MODel? <name>

**Return Type**
String

**Default**
Not Applicable

---

SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:PLEVel? <name>
**Applicable Models:** N522xB, N523xB, N524xB (requires Option S93015A)

*(Read-only)* Returns the power level associated with the best accuracy for a specific power meter.

**Note:** This is typically the level where the calfactor was obtained.

**Parameters**

- `<name>` String - Name of the device used as the power meter.

**Examples**

```
```

**Return Type**

- Double (dBm)

**Default**

Not Applicable

---

**SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:READing? <name>,<frequency>,<power>**

**Applicable Models:** N522xB, N523xB, N524xB (requires Option S93015A)

*(Read-only)* Returns the power uncertainty associated with the specific power meter at the specified frequency and power. The returned value is the variance of the power expressed in [mW]^2.

**Parameters**

- `<name>` String - Name of the device used as the power meter.
- `<frequency>` Frequency (Hz).
- `<power>` Power (dBm).

**Examples**

```
SYST:CONF:EDEV:PMAR:UNC:READ? "myDevice",10e9,0.0
system:configure:edevice:pmar:uncertainty:read? "myDevice",10e9,0.0
```

**Return Type**

- Double (W^2)

**Default**

Not Applicable

---

**SYSTem:CONFigure:EDEVice:PMAR:ZERO <name>[,SYNC,<value>]**
**Applicable Models:** All

*(Write-only)* Performs a zeroing of the PMAR device.

This command is always synchronous, so *OPC?* is the only way to determine that the operation is complete. Set an I/O timeout of at least 20 seconds.

Keysight P-Series sensors do ONLY Internal zeroing. These, and Keysight USB power sensors when Internal is selected, do NOT require disconnecting from the measurement path before zeroing.

All other Keysight sensors do ONLY External zeroing.

**Parameters**

- `<name>` String - Name of the power meter.

- `[SYNC,<value>]` Optional argument for use with power sensors that support both internal and external types of zeroing such as Keysight USB power sensors.

  Choose from:

  - **SYNC,Internal** - Internal zeroing. Power is automatically removed from the sensor input before zeroing occurs (Default setting).

  - **SYNC,External** - External zeroing. First remove the sensor input, then send this command. External zeroing is recommended for powers below -30 dBm with the U2000-Series sensors (-20 dBm for the H models).

**Examples**

```
SYST:CONF:EDEV:PMAR:ZERO "myDevice"
```

```
SYSTEM:configure:edevice:pmar:zero "myDevice",sync,internal
```

**Query Syntax** Not Applicable

**Default** Not Applicable
SYSTem:CONF:EDEVice:PULSe Commands

Configures and makes settings for an external Keysight 81110A Pulse Generator.

```
SYST:CONF:EDEVice:PULSe
  | CHAN
  | HAMP
  | LAMP
  | LIMP
  | PMODE
  | SIMP
```

Click on a keyword to view the command details.

**See Also**

- Root SYST:CONF:EDEV commands
- All Integrated Pulse App commands
- Learn about: Configure an External Pulse Generator
- Learn about Configure and External Device
- **SYST:PREF:ITEM:EDEV:DPOL** - Determines whether External Devices remain activated or are de-activated when the VNA is Preset or when a Instrument State is recalled.
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

**SYSTem:CONFigure:EDEVice:PULSe:CHAN <name>,<value>**
**Applicable Models:** All

(Read-Write) Sets and returns the output channel of the pulse generator.

**Parameters**

- `<name>`: String - Name of the external pulse generator.
- `<value>`: Pulse Generator output port. Choose from 1 or 2.

**Examples**

```
SYST:CONF:EDEV:PULS:CHAN "81110",1
system:configure:edevice:pulse:chan "myPG",2
```

**Query Syntax**

```
SYSTem:CONFigure:EDEVice:DC:PULSe:CHAN? <name>
```

**Return Type**

Numeric

**Default**

1

---

**SYSTem:CONFigure:EDEVice:PULSe:HAMP <name>,<value>**

(Read-Write) Sets and returns the High amplitude (voltage) of the pulse generator.

**Parameters**

- `<name>`: String - Name of the external pulse generator.
- `<value>`: Pulse Generator high amplitude voltage.

**Examples**

```
SYST:CONF:EDEV:PULS:HAMP "81110",3
system:configure:edevice:pulse:HAMP "myPG",4
```

**Query Syntax**

```
SYSTem:CONFigure:EDEVice:DC:PULSe:HAMP? <name>
```

**Return Type**

Numeric

**Default**

5

---

**SYSTem:CONFigure:EDEVice:PULSe:LAMP <name>,<value>**
Applicable Models: All

(Read-Write) Pulse Generator low amplitude voltage.

Parameters

- `<name>` String - Name of the external pulse generator.
- `<value>` Pulse Generator low amplitude voltage.

Examples

```
SYST:CONF:EDEV:PULS:LAMP "81110",.2
```
```
system:configure:edevice:pulse:lamp "myPG",1
```

Query Syntax

```
SYSTem:CONFigure:EDEVice:DC:PULSe:LAMP? <name>
```

Return Type Numeric

Default 0

---

SYSTem:CONFigure:EDEVice:PULSe:LIMP <name>,<value>

Applicable Models: All

(Read-Write) Sets and returns the load impedance of the pulse generator.

Parameters

- `<name>` String - Name of the external pulse generator.
- `<value>` Pulse generator load impedance.

Examples

```
SYST:CONF:EDEV:PULS:LIMP "81110",52
```
```
system:configure:edevice:pulse:limp "myPG",49
```

Query Syntax

```
SYSTem:CONFigure:EDEVice:DC:PULSe:LIMP? <name>
```

Return Type Numeric

Default 50

---

SYSTem:CONFigure:EDEVice:PULSe:PMODE <name>,<bool>
Applicable Models: All

(Read-Write) Sets and returns the primary (On/Off) setting of the external pulse generator. The ON setting allows the external pulse generator to set the primary clock frequency for the other pulse generators.

Parameters

- `<name>` String - Name of the external pulse generator.
- `<bool>` Primary setting. Choose from:
  - **ON** or 1 - Use the external pulse generator becomes the primary clock frequency.
  - **OFF** or 0 - Use the internal pulse generator as the primary clock frequency.

Examples

```
SYST:CONF:EDEV:PULS:PMOD "81110",OFF
system:configure:edevice:pulse:pmode "myPG",1
```

Query Syntax

```
SYSTem:CONFigure:EDEVice:DC:PULSe:PMODe? <name>
```

Return Type

Boolean

Default

OFF or 0

---

SYSTem:CONFigure:EDEVice:PULS:SIMP <name>,<value>

Applicable Models: All

(Read-Write) Sets and returns the source impedance of the pulse generator.

Parameters

- `<name>` String - Name of the external pulse generator.
- `<value>` Pulse generator source impedance.

Examples

```
SYST:CONF:EDEV:PULS:SIMP "81110",52
system:configure:edevice:pulse:simp "myPG",49
```

Query Syntax

```
SYSTem:CONFigure:EDEVice:DC:PULSe:SIMP? <name>
```

Return Type

Numeric

Default

50
The five SYSTem:CORRection:INTerpolate:LINear commands are used as a sequence. They are not meant to be used independent of the others. The commands perform linear interpolation on a scalar data of (x,y) pairs based on a primary set of x and y values to a new mapping based on a desired set of x values. The desired set of x values (or range) must fall within the primary set of x values (or range) and can have a different number of points than the primary data set.

Definition:

Linear interpolation operates by drawing a straight line between each two adjacent data points on the primary (x,y) pairs that fall on either side of the new desired data point represented by (x’,y’). In other words if

\[(xi, yi)\] represents data pairs on the primary data set and \[(xj’,yj’)\] represents a data point on the interpolated result data set then:

\[X_i < X_j' < X_{i+1} \quad \text{and} \quad Y_j' = Y_i + \frac{(Y_{i+1} - Y_i)}{(X_{i+1} - X_i)}(X_j' - X_i)\]

Note: The primary data set must represent a function on the Cartesian coordinate system. In other words, for each x value in the primary data set, there can be only one corresponding Y value.

There are five steps in the sequence:

1. SYSTem:CORRection:INTerpolate:LINear:INPut:X - loads in the primary X values
2. SYSTem:CORRection:INTerpolate: LINear:INPut:Y - loads in the primary Y values
3. SYSTem:CORRection:INTerpolate: LINear:OUTPut:X - loads in the desired interpolated X values
4. SYSTem:CORRection:INTerpolate: LINear:CALCulate - calculates the interpolated Y values

Example

The following function uses the SYSTem:CORRection:INTerpolate: LINear commands:

```vbnet
Function InterpolateData_Single(inputX() As Double, inputY() As Single, outputX() As Double, ByVal interpData() As Single)
    x = visa_io.ag_send_binBlock64("SYST:CORR:INT:LIN:INP:X ", inputX)
    x = visa_io.ag_send_binBlock("SYST:CORR:INT:LIN:INP:Y ", inputY)
    x = visa_io.ag_send_binBlock64("SYST:CORR:INT:LIN:OUTP:X ", outputX)
End Function
```
 Dim BResp_freqList() As Double
 Dim BResp_Re() As Single
 Dim BResp_Im() As Single

 GetErrorTerm_noChan BResp_Calset, "ResponseTracking(B)", BResp_freqList, BResp_Re, BResp_Im

 Dim Noise_freqList() As Double

 GetCalsetStimulus calsetName, Noise_freqList, 1, "Noise Figure Cold Source"
' Response Stimulus Range

 Dim BResp_Re_interp() As Single
 Dim BResp_Im_interp() As Single

 InterpolateData_Single BResp_freqList, BResp_Re, Noise_freqList, BResp_Re_interp
 InterpolateData_Single BResp_freqList, BResp_Im, Noise_freqList,
BResp_Im_interp

PutErrorTerm channel, calsetName, "ResponseTracking(B)", BResp_Re_interp,
BResp_Im_interp
The 4 GB FIFO data buffer is available with Option S93118A/B or Option S930900A/B on the VNA and N5264B. These commands control data in and out of FIFO data buffer. The FIFO can be emptied as it is being filled, which means that the VNA can be used to acquire an infinite amount of data.

The data placed into the FIFO is the raw data after averaging and ratioing has been applied, but prior to any calibration, formatting, or data analysis functions.

Click on a red keyword to view the command details.

See Also

- FIFO and other Antenna Features
- Fast CW command
- FIFO Example Program
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

SYSTem:FIFO:DATA? <dpoints>
**Applicable Models:** N522xB, N524xB

(Read-only) Reads the next specified number of data points from the FIFO buffer. Each data point is returned as a real/imaginary pair. Data is cleared as it is read.

### Parameters

- **<dPoints>** Number of data points to read. An error is returned if the amount of requested data is larger than the available data.

### Examples

```
SYST:FIFO:DATA? 1e6

system:fifo:data? 1e3
```

### Return Type

Use `FORMat:DATA` to change the data type (<REAL,32>, <REAL,64> or <ASCii,0>). For best results, use REAL,32.

Use `FORMat:BORDer` to change the byte order. Use “NORMal” when transferring a binary block from LabView or Vee. For other programming languages, you may need to "SWAP" the byte order.

Each data point is returned as a real/imaginary pair.

**Default** Not applicable

---

**SYSTem:FIFO:DATA:BYTe? <X>**

**Applicable Models:** N522xB, N524xB

(Read-only) Returns a specific number of bytes to read.

### Parameters

- **<X>** Number of bytes to read.

### Examples

```
SYST:FIFO:DATA:BYTe? 4096

system:fifo:byte? 4096
```

### Return Type

IEEE binary block

**Default** Not applicable

---

**SYSTem:FIFO:DATA:BYTe:COUNt?**
Applicable Models: N522xB, N524xB

(Read-only) Returns a specific number of bytes to read.

Parameters

Examples

Return Type  Integer

Default  Not applicable

SYS: FIFO: DATA: BYTe: COUNT?

SYST: FIFO: DATA: CLEar

Applicable Models: N522xB, N524xB

(Write-only) Clears the data from the FIFO buffer.

Parameters  None

Examples

Return Type  None

Default  Not applicable

SYS: FIFO: DATA: COUNT?

(Read-only) Returns the total number of data points in the FIFO buffer.

Parameters  None

Examples

Return Type  Numeric

Default  Not applicable

SYS: FIFO[:STATe] <bool>
Applicable Models: N522xB, N524xB

(Write-Read) Sets and returns the state of data storage to the FIFO buffer. Syst:Preset or an instrument state recall also ends storage to the FIFO buffer. The FIFO buffer is cleared when set to OFF.

Parameters

<bool> FIFO buffer state. Choose from:

**ON or 1** Data is stored in the FIFO buffer.

**OFF or 0** Data is NOT stored in the FIFO buffer.

Examples

```plaintext
SYST:FIFO 1
system:fifo:state off
```

Query Syntax

SYSTem:FIFO[:STATE]?

Return Type

Boolean

Default

0 OFF
Sets and reads the VNA Preferences settings.

<table>
<thead>
<tr>
<th>SYSTem:PREFerences</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFault</td>
</tr>
<tr>
<td>ITEM</td>
</tr>
<tr>
<td>ASMRamp</td>
</tr>
<tr>
<td>CORRectrion</td>
</tr>
<tr>
<td>PARallel</td>
</tr>
<tr>
<td>PROCess</td>
</tr>
<tr>
<td>CSET</td>
</tr>
<tr>
<td>COMPact</td>
</tr>
<tr>
<td>DIALog</td>
</tr>
<tr>
<td>SHOW</td>
</tr>
<tr>
<td>LICense</td>
</tr>
<tr>
<td>EEXTrapolate</td>
</tr>
<tr>
<td>EDEV: DPOLicy</td>
</tr>
<tr>
<td>FIXTure</td>
</tr>
<tr>
<td>CIRCuit</td>
</tr>
<tr>
<td>DEFAults</td>
</tr>
<tr>
<td>GDELat</td>
</tr>
<tr>
<td>LEGacy</td>
</tr>
<tr>
<td>TWOPoint</td>
</tr>
<tr>
<td>KEYS</td>
</tr>
<tr>
<td>MARKer: BANDwidth: SEARch</td>
</tr>
<tr>
<td>MARKer: SINGle</td>
</tr>
<tr>
<td>MCControl</td>
</tr>
<tr>
<td>MCMethod</td>
</tr>
<tr>
<td>MCPreset</td>
</tr>
<tr>
<td>MINTerpolate</td>
</tr>
<tr>
<td>MRU</td>
</tr>
<tr>
<td>OFFSet</td>
</tr>
<tr>
<td>RCV</td>
</tr>
</tbody>
</table>
Click on a keyword to view the command details.

See Also

- SENS:CORRection:PREFerences
SYSTem:PREFERENCEs:ITEM:ASMRamp <bool>

**Applicable Models:** N522xB, N523xB, N524xB, E5080A/B

*(Read-Write)* Set and return whether ramp sweep is used whenever possible when sweep mode is in Auto.

**Parameters**

<bool> Choose from:

- **ON (1)** Enable ramp sweep.
- **OFF (0)** Disable ramp sweep.

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:PREF:ITEM:ASMR 1</td>
</tr>
<tr>
<td>system:preferences:item:asmramp OFF</td>
</tr>
</tbody>
</table>

**Query Syntax**

SYSTem:PREFERENCEs:ITEM:ASMRamp?

**Return Type**

Boolean

**Default**

OFF

---

SYSTem:PREFERENCEs:DEFault

**Applicable Models:** All

*(Write-only)* Resets the VNA preferences to their default settings. Some default settings vary depending on the VNA Model. Learn more about VNA Preferences.

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:PREF:DEF</td>
</tr>
<tr>
<td>system:preferences:default</td>
</tr>
</tbody>
</table>

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

SYSTem:PREFERENCEs:ITEM:CORRection:PARallel:PROCess <bool>
Applicable Models: All

(Read-Write) Enable or disable parallel processing in the CPU which provides higher calculation speeds.

**Parameters**
- `<bool>` Choose from:
  - **OFF (0)** Disable parallel processing.
  - **ON (1)** Enable parallel processing.

**Examples**
- `SYST:PREF:ITEM:CORR:PAR:PROC 1`
- `system:preferences:item:correction:parallel:process OFF`

**Query Syntax**
- `SYSTem:PREFerences:ITEM:CORRection:PARallel:PROCess?`

**Return Type**
- Boolean

**Default**
- ON or 1

---

SYSTem:PREFerences:ITEM:EDEV:DPOLicy `<bool>`

Applicable Models: All

(Read-Write) Set and return whether External Devices remain activated or are de-activated when the VNA is Preset or when a Instrument State is recalled.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

**Parameters**
- `<bool>` Choose from:
  - **OFF (0)** External devices remain active when the VNA is Preset or when a Instrument State is recalled.
  - **ON (1)** External devices are de-activated (`SYST:CONF:EDEV:STAT` to OFF) when the VNA is Preset or when a Instrument State is recalled.

**Examples**
- `SYST:PREF:ITEM:EDEV:DPOL 1`
- `system:preferences:item:edev:dpolicy OFF`

**Query Syntax**
- `SYSTem:PREFerences:ITEM:EDEV:DPOLicy?`

**Return Type**
- Boolean

**Default**
- ON or 1
SYSTem:PREFerences:ITEM:CSET:COMPact <bool>

Applicable Models: All

(Read-Write) Enables/disables the creation of compact calsets.

Parameters
  <bool> Choose from:

  ON (1) Enable creation of compact calsets.

  OFF (0) Disable creation of compact calsets.

Examples
SYST:PREF:ITEM:CSET:COMP 1

system:preferences:item:cset:compact OFF

Query Syntax
SYSTem:PREFerences:ITEM:CSET:COMPact?

Return Type
Boolean

Default
OFF

SYSTem:PREFerences:ITEM:DIALog:SHOW:LICense <bool>

Applicable Models: All

(Read-Write) Enables/disables showing the licensed features dialog on startup.

Parameters
  <bool> Choose from:

  ON (1) Enable showing licensed features on startup.

  OFF (0) Disable showing licensed features on startup.

Examples
SYST:PREF:ITEM:DIAL:SHOW:LIC 1

system:preferences:item:dialog:show:license OFF

Query Syntax
SYSTem:PREFerences:ITEM:DIALog:SHOW:LICense?

Return Type
Boolean

Default
OFF

SYSTem:PREFerences:ITEM:EEXTrapolate <bool>
Applicable Models: All

(Read-Write) Sets whether a Swept IMD or IMDx calibration can exceed the stop frequency limit of an ECal module. Learn more.

Parameters

<bool> Choose from:

ON (1) Allow extrapolation.

OFF (0) Do NOT allow extrapolation.

Examples

```
SYST:PREF:ITEM:EEXT 1
```

```
system:preferences:item:eextrapolate OFF
```

Query Syntax

SYSTem:PREFerences:ITEM:EEXTrapolate?

Return Type

Boolean

Default

OFF

SYSTem:PREFerences:ITEM:FIXTure:CIRCuit:DEFAults <char>

Applicable Models: All

(Read-Write) Sets the Series-C and Shunt-L components to legacy or theoretical behavior when set to zero.

Note: This command is only applicable in Firmware versions A.14.00.00 and later.

Parameters

<char> Choose from:

THEoretical - When "R" or "L" is set to 0, the component is defined as "Short"; when "C" or "G" is set to 0, the component is defined as "Open".

LEGacy - Legacy operation provided as a user-convenience for backward compatibility. When either of the two components of Series-C, "C", and "G", is set to 0, the component is defined as "Open"; when both components are set to 0, they are defined as "Short". When either of the two components of Shunt-L, "L", and "R", is set to 0, the component is defined as "Short"; when both components are set to 0, they are defined as "Open".

Examples

```
SYST:PREF:ITEM:FIXT:CIRC:DEFA THE
```

```
system:preferences:item:fixture:circuit:defaults legacy
```

Query Syntax

SYSTem:PREFerences:ITEM:FIXTure:CIRCuit:DEFAults?
SYSTem:PREFerences:ITEM:GDELay:LEGacy <bool>

**Applicable Models:** All

*(Read-Write)* Sets the group delay aperture to use the legacy computation method. [Learn more about group delay aperture.](#)

**Parameters**

<bool> Choose from:

- **ON (1)** Use legacy computation method.
- **OFF (0)** Do not use legacy computation method.

**Examples**

```
SYST:PREF:ITEM:GDEL:LEG 1
system:preferences:item:gdelay:legacy OFF
```

**Query Syntax**

SYSTem:PREFerences:ITEM:GDELay:LEGacy?

**Return Type** Boolean

**Default** OFF

SYSTem:PREFerences:ITEM:GDELay:TWOPoint <bool>

**Applicable Models:** All

*(Read-Write)* Sets the default group delay aperture setting. [Learn more about group delay aperture.](#)

**Parameters**

<bool> Choose from:

- **ON (1)** Sets default group delay aperture to 2 points.
- **OFF (0)** Sets default group delay aperture to 11 points.

**Examples**

```
SYST:PREF:ITEM:GDEL:TWOP 1
system:preferences:item:gdelay:twopoint OFF
```

**Query Syntax**

SYSTem:PREFerences:ITEM:GDELay:TWOPoint?

**Return Type** Boolean

**Default** OFF
SYSTem:PREferences:ITEM:KEYS <bool>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Set and return whether the keys are displayed or not.

Parameters

<bool> Choose from:

**ON (1)** – Turn keys on.

**OFF (0)** – Turn keys off.

Examples

SYST:PREF:ITEM:KEYS 1

system:preferences:item:keys OFF

Query Syntax

SYSTem:PREferences:ITEM:KEYS?

Return Type

Boolean

Default

OFF (0)

---

SYSTem:PREferences:ITEM:MARKer:BANDwidth:SEARch <char>

Applicable Models: All

(Read-Write) Sets the bandwidth search preference to start a bandwidth or notch search in either peak or marker mode.

Parameters

<char> Choose from:

**MARKer** - BW/Notch marker search reference is set to current marker position after Preset.

**PEAK** - BW/Notch marker search reference is set to peak after Preset.

Examples

SYST:PREF:ITEM:MARK:BAND:SEAR MARK

system:preferences:item:marker:bandwidth:search PEAK

Query Syntax

SYSTem:PREferences:ITEM:MARKer:BANDwidth:SEARch?

Return Type

Character

Default

MARK (E5080A/B), PEAK (Others)

---

SYSTem:PREferences:ITEM:MARKer:SINGle <bool>
Applicable Models: All

(Read-Write) Set and return whether to use one marker for marker search.

Enabled behavior:

- Only one marker is used for bandwidth, notch, PNOP, and PSAT marker searches. The points of interest are marked with a notational UI element, i.e. a small triangle.
- Bandwidth, notch, PNOP, and PSAT marker searches are always tracking. Tracking cannot be disabled.
- One basic search and one advanced search may be set per marker.
- The advanced search is enabled until the user disables the search or a multi-peak or multi-target search is executed.

Disabled behavior:

- Bandwidth, notch, PSAT, and PNOP marker searches use multiple markers.
- One advanced marker search is allowed per trace.
- A marker may only perform a basic search or be part of an advanced search. Not both.
- If an advanced marker search is enabled on a trace and then the user performs a basic search, the advanced search is automatically disabled.
- Advanced searches may enable or disable tracking. Only one search may be tracked.

Parameters

<bool> Choose from:

ON (1) Enable single marker search.

OFF (0) Disable single marker search.

Examples

SYSTM:PREF:ITEM:MARK:SING 1
system:preferences:item:marker:single OFF

Query Syntax

SYSTem:PREFerences:ITEM:MARKer:SINGle?

Return Type

Boolean

Default

ON

SYSTem:PREFerences:ITEM:MCControl <bool>
Applicable Models: All

(Read-Write) Set and return whether the Coupled Markers setting controls the ON|OFF state of markers that are coupled. Learn more about Coupled Markers. Refer also to CALC:MEAS:MARK:COUP:STATE ON.

Parameters

\(<\text{bool}>\) Choose from:

ON (1) – With Coupled Markers ON, when a marker is turned on, the same-numbered marker on all coupled traces will also be turned on. Likewise, turning off a marker will turn it off on all coupled traces.

OFF (0) – Turning a marker on or off will have no effect on the markers on other traces.

Examples

```
SYST:PREF:ITEM:MCC 1
```
```
system:preferences:ITEM:mccontrol OFF
```

Query Syntax

SYSTem:PREFerences:ITEM:MCControl?

Return Type

Boolean

Default

OFF (0)

---

SYSTem:PREFerences:ITEM:MCMethod <\text{bool}>

Applicable Models: All

(Read-Write) Set and return whether Coupled Markers is set to Channel or All after Preset. Learn more about Coupled Markers. Refer also to CALC:MEAS:MARK:COUP:STATE ON and SYST:PREF:ITEM:MCPR ON.

Parameters

\(<\text{bool}>\) Choose from:

ON (1) – Marker Coupling Method is set to Channel after Preset.

OFF (0) – Marker Coupling Method is set to ALL after Preset.

Examples

```
SYST:PREF:ITEM:MCM 1
```
```
system:preferences:ITEM:mcmcontrol OFF
```

Query Syntax

SYSTem:PREFerences:ITEM:MCMethod?

Return Type

Boolean
**SYSTem:PREFerences:ITEM:MCPRest <bool>**

**Applicable Models:** All

(Read-Write) Set and return whether Coupled Markers is set to ON or OFF after Preset. Learn more about Coupled Markers.

**Parameters**

<bool> Choose from:

- **OFF (0)** – Coupled Markers is OFF after Preset.
- **ON (1)** – Coupled Markers is ON after Preset.

**Examples**

```
SYST:PRE:F:ITEM:MCPR 1
system:preferences:item:mcpreset OFF
```

**Query Syntax**

SYSTem:PREFerences:ITEM:MCPReset?

**Return Type**

Boolean

**Default**

OFF (0)

---

**SYSTem:PREFerences:ITEM:MIInterpolate <bool>**

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets and reads the state of the memory data interpolation default preference. The PNA will return to the default interpolation state after a Preset, creating a new trace, or closing the PNA application. Learn more.

**Parameters**

<bool> Choose from:

- **0 - OFF** - Set memory interpolation to OFF as the default.
- **1 - ON** - Set memory interpolation to ON as the default.

**Examples**

```
SYST:PRE:F:ITEM:MINT 1
```

**Query Syntax**

SYSTem:PREFerences:ITEM:MIInterpolate?

**Return Type**

Boolean

**Default**

0
SYSTem:PREFerences:ITEM:MRU <bool>

Applicable Models: All

(Read-Write) Set and return whether to list files for recall on softkeys by most-recently used or alphabetically.

Parameters

<bool> Choose from:

ON (1) – Recall softkeys show most recently-used files.

OFF (0) – Recall softkeys show alphabetically-ordered files.

Examples

SYST:PREF:ITEM:MRU 1
system:preferences:item:mru OFF

Query Syntax

SYSTem:PREFerences:ITEM:MRU?

Return Type

Boolean

Default

OFF (0)

SYSTem:PREFerences:ITEM:OFFSet:RCV <bool>

Applicable Models: All

(Read-Write) Set and return whether to offset the test port receivers by the amount of receiver attenuation. Learn more.

To send this command using the VNA front panel, open the GPIB Command Processor Console, then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the VNA took the command.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

Parameters

<bool> Choose from:

ON (1) Offset the test port receivers

OFF (0) Do NOT offset the test port receivers

Examples

SYST:PREF:ITEM:OFFS:RCV 1
system:preferences:item:offset:rcv OFF
**Query Syntax**  SYSTem:PREFerences:ITEM:OFFSet:RCV?

**Return Type**  Boolean

**Default**  PNA-L and E836xB: **OFF** (does NOT offset the display).

PNA-X: **ON** (offsets the display).

SYSTem:PREFerences:ITEM:OFFSet:SRC <bool>

**Applicable Models:** All

*(Read-Write)* Set and return whether to offset the reference receiver by the amount of source attenuation. Learn more.

To send this command using the VNA front panel, open the GPIB Command Processor Console, then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the VNA took the command.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

**Parameters**

<bool>  Choose from:

**ON (1)** Offset the reference receivers.

**OFF (0)** Do NOT Offset the reference receivers.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:PREF:ITEM:OFFS:SRC 1</td>
</tr>
<tr>
<td>system:preferences:item:offset:src OFF</td>
</tr>
</tbody>
</table>

**Query Syntax**  SYSTem:PREFerences:ITEM:OFFSet:SRC?

**Return Type**  Boolean

**Default**  All models: **ON** (offset the display).

SYSTem:PREFerences:ITEM:OPTimize:MEMory <bool>
**Applicable Models:** M9485A

*(Read-Write)* Set and return the memory optimization function status. This extends the maximum number of channel. However, this causes slow measurement.

**Parameters**

<bool> Choose from:

**ON (1)** Memory optimization on (more channels, but slower speed)

**OFF (0)** Memory optimization off

**Examples**

SYST:PREF:ITEM:OPT:MEM 1

system:preferences:item:optimize:memory OFF

**Query Syntax**

SYSTem:PREFerences:ITEM:OPTimize:MEMory?

**Return Type** Boolean

**Default** OFF

**SYSTem:PREFerences:ITEM:PRES:CONFirm <bool>**

**Applicable Models:** All

*(Read-Write)* Set and return preset confirmation. If preset confirmation is OFF, pressing the green PRESET key presets the instrument and opens the Preset softkey menu. If preset confirmation is ON, pressing the green PRESET causes the Preset menu to appear.

**Parameters**

<bool> Choose from:

**ON (1)** Enable preset confirmation.

**OFF (0)** Disable preset confirmation.

**Examples**

SYST:PREF:ITEM:PRES:CONF 1

system:preferences:item:preset:confirm OFF

**Query Syntax**

SYSTem:PREFerences:ITEM:PRESet:CONFirm?

**Return Type** Boolean

**Default** ON

**SYSTem:PREFerences:ITEM:PRESet:POWer[:STATe] <char>**
Applicable Models: All

(Read-Write) Set and return the Preset Power ON/OFF state. Learn more.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

Parameters

<char> Choose from:

ON - Instrument Preset always turns RF power ON.

AUTO - When the current power setting is OFF, leave power OFF after Preset. When the current power setting is ON, turn power ON after Preset.

Examples

SYST:PREF:ITEM:PRE:POW ON
system:preferences:item:preset:power:state auto

Query Syntax

SYSTem:PREFerences:ITEM:PREset:POWer[:STATe]?

Return Type

Character

Default

ON

SYSTem:PREFerences:ITEM:PSRTrace <char>

Applicable Models: All

(Read-Write) At the end of a power sweep, while waiting to trigger the next sweep, maintain source power at either the start power level or at the stop power level.

To send this command using the VNA front panel, open the GPIB Command Processor Console, then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the VNA took the command.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

Parameters

<char> Choose from:

START - Maintain source power at the start power level.

STOP - Maintain source power at the stop power level.

Examples

SYST:PREF:ITEM:PSRT STOP
system:preferences:item:psrtrace start
**Query Syntax**
SYSTem:PREFerences:ITEM:PSRTrace?

**Return Type**
Character

**Default**
STARt

**SYSTem:PREFerences:ITEM:QSTart <bool>**

**Applicable Models:** All

*(Read-Write)* This command controls the on/off state of the preference, "On PRESET show Quick Start dialog".

**Parameters**

<bool> Choose from:

**ON (1)** Display the Quick Start dialog on PRESET.

**OFF (0)** Do not display the Quick Start dialog on PRESET.

**Examples**

SYST:PREF:ITEM:QST 1
system:preferences:item:qstart OFF

**Query Syntax**
SYSTem:PREFerences:ITEM:QST?

**Return Type**
Boolean

**Default**
OFF

**SYSTem:PREFerences:ITEM:RECeivers:CERRor[:STATe] <bool>**

**Applicable Models:** All

*(Read-Write)* Set and return whether to display receiver overload warnings. Learn more.

**Parameters**

<bool> Choose from:

**ON (1)** Display overload warnings,

**OFF (0)** Do NOT display overload warnings.

**Examples**

SYST:PREF:ITEM:REC:CERR 1
system:preferences:item:receivers:cerror:state OFF

**Query Syntax**
SYSTem:PREFerences:ITEM:RECeivers:CERRor[:STATe]?

**Return Type**
Boolean
SYSTem:PREFerences:ITEM:RECeivers:OVERload:POWer[:STATe] <bool>

Applicable Models: All

(Read-Write) Set and return whether to turn source power OFF when a receiver is overloaded. Learn more.

Parameters

 BOOL Choose from:

ON (1) Turn OFF source power to ALL ports when a receiver is overloaded.

OFF (0) Power remains ON when a receiver is overloaded.

Examples

SYST:PREF:ITEM:REC:OVER:POW 1
system:preferences:item:receivers:overload:power:state OFF

Query Syntax

SYSTem:PREFerences:ITEM:RECeivers:OVERload:POWer[:STATe]?

Return Type

Boolean

Default

OFF (0)

SYSTem:PREFerences:ITEM:REDLimits <bool>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Set and return whether to draw limits lines in Red or the trace color.

Parameters

 BOOL Choose from:

ON (1) All Limit lines are drawn in Red.

OFF (0) Limit lines are drawn the same color as the trace.

Examples

SYST:PREF:ITEM:REDL 1
system:preferences:item:redlimits OFF

Query Syntax

SYSTem:PREFerences:ITEM:REDLimits?

Return Type

Boolean

Default

OFF
**SYSTem:PREFerences:ITEM:REFMarker <bool>**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-Write)* Set and return whether to treat marker 10 as a reference marker. [Learn more.]

**Parameters**

- *<bool>* Choose from:
  
  **ON (1)** Marker 10 is always a reference marker (Pre A.10.40 behavior).
  
  **OFF (0)** Marker 10 is just another marker. See Reference Marker commands

**Examples**

- `SYST:PREF:ITEM:REFM 1`
- `system:preferences:item:refmarker OFF`

**Query Syntax**

- `SYSTem:PREFerences:ITEM:REFMarker?`

**Return Type**

- Boolean

**Default**

- OFF

---

**SYSTem:PREFerences:ITEM:REMote:AUTO[:STATe] <bool>**

**Applicable Models:** All

*(Read-Write)* Enables/disables changing from local to remote status when a SCPI command is received.

**Parameters**

- *<bool>* Choose from:
  
  **OFF (0)** Turn OFF local/remote behavior.
  
  **ON (1)** Turn ON local/remote behavior.

**Examples**

- `SYST:PREF:ITEM:REM:AUTO 0`
- `system:preferences:item:remote:auto:state OFF`

**Query Syntax**

- `SYSTem:PREFerences:ITEM:REMote:AUTO[:STATe]?`

**Return Type**

- Boolean

**Default**

- ON

---

**SYSTem:PREFerences:ITEM:RETRace:POWer <char>**
Applicable Models: N522xB, N523xB, N524xB, E5080B

(Read-Write) For single-band frequency or segment sweeps ONLY, specify whether to turn RF power ON or OFF during a retrace. Learn more about RF power during sweep retrace.

To send this command using the VNA front panel, open the GPIB Command Processor Console, then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the VNA took the command.

This setting remains until changed using this command, or until the hard drive is changed or reformatted.

**Parameters**

<char> Choose from:

**AUTO**: Power is left ON during retrace of single-band frequency or segment sweeps ONLY.

**OFF**: Power is turned OFF during retrace of single-band frequency or segment sweeps ONLY.

**Examples**

```
SYST:PREF:ITEM:RETR:POW OFF
system:preferences:item:retrace:power auto
```

**Query Syntax** SYSTem:PREFerences:ITEM:RETRace:POWer?

**Return Type** Character

**Default** AUTO

```
SYSTem:PREFerences:ITEM:ROSCillator:RECall <bool>
```
**Applicable Models:** All

*(Read-Write)* Enables/disables External Reference settings being affected by Recall/Preset.

**Parameters**

- `<bool>` Choose from:
  - **OFF (0)** External Reference settings will be maintained until changed.
  - **ON (1)** External Reference settings will be affected by Recall/Preset.

**Examples**

```
SYST:PREF:ITEM:ROSC:REC 0
system:preferences:item:roscillator:recall OFF
```

*Query Syntax*  
SYSTem:PREFerences:ITEM:ROSCillator:RECall?

*Return Type*  
Boolean

*Default*  
ON

**SYSTem:PREFerences:ITEM:RTOF <bool>**

**Applicable Models:** All

*(Read-Write)* Set and return whether to display limit line failures as red trace segments or red data points (dots). Learn more.

**Parameters**

- `<bool>` Choose from:
  - **ON (1)** Display failures as red trace segments. (Red Trace On Fail).
  - **OFF (0)** Display failures as red data points (dots).

**Examples**

```
SYST:PREF:ITEM:RTOF 1
system:preferences:item:rtof OFF
```

*Query Syntax*  
SYSTem:PREFerences:ITEM:RTOF?

*Return Type*  
Boolean

*Default*  
OFF

**SYSTem:PREFerences:ITEM:SOFTkeys:NAVigation <bool>**
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) This command controls the on/off state of the preference, "Use keyboard to navigate softkeys".

Parameters

<bool> Choose from:

ON (1) Enable softkey navigation with keyboard.

OFF (0) Disable softkey navigation with keyboard.

Examples

SYST:PREF:ITEM:SOFT:NAV 1
system:preferences:item:softkeys:navigation OFF

Query Syntax

SYSTem:PREFerences:ITEM:SOFTkeys:NAVigation?

Return Type

Boolean

Default

OFF

---

SYSTem:PREFerences:ITEM:SWITch:DEF <string>, <int>

Applicable Models: All

(Read-Write) Sets the default setting for the Noise Tuner switch. This is the setting that occurs when a new channel is created. Learn more.

This command will return an error on VNA models with a built-in Noise tuner.

To send this command using the VNA front panel, open the GPIB Command Processor Console, then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the VNA took the command.

This setting remains until changed using this command, or until the hard drive is changed or reformatted.

Parameters

<string> Name of the switch to set. Choose from:

- "Port1NoiseTuner"

<int> Value to set. Choose from:

0 Sets the default (preset) to INTERNAL

1 Sets the default (preset) to EXTERNAL
### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:PREF:ITEM:SWIT:DEF &quot;Port1NoiseTuner&quot; 1 'Write</td>
<td>System:preferences:item:switch:def? &quot;Port1NoiseTuner&quot; 'Read</td>
<td>1 (External)</td>
</tr>
</tbody>
</table>

### SYStem:PREferenceS:SOURce<port>:GLOBAL:FRQuency <num>

**Applicable Models:** All

*(Read-Write)* Set and return the frequency of the specified global source. **Learn more.**

**Parameters**

- **<port>** Source port number of the VNA.
- **<num>** Global source frequency. Choose any number between the **minimum** and **maximum** frequency limits of the analyzer. Units are Hz.

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
</table>

### Query Syntax

SYSTem:PREFerences:SOURce<port>:GLOBAL:FRQuency?

**Return Type** Numeric

**Default** Not Applicable

### SYSTem:PREferences:SOURce<port>:GLOBAL:OUTPut[:STATe] <bool>
Applicable Models: All

(Read-Write) Set and return the output state of the specified global source. Learn more.

Parameters

<port> Source port number of the VNA.
<br> Choose from:

**OFF (0)** Turn global source output OFF.

**ON (1)** Turn global source output ON.

Examples

```
SYST:PREF:SOUR2:GLOB:OUTP 1
```

```
 system:preferences:source3:global:output:state OFF
```

Query Syntax

SYSTem:PREFerences:SOURce<port>:GLOBal:OUTPut[:STATe]?

Return Type

Boolean

Default OFF

---


Applicable Models: All

(Read-Write) Set and return the global sources that ignore the power off setting. Learn more.

Parameters

<port> Source port number of the VNA.
<br> Choose from:

**OFF (0)** Do not ignore power off settings for specified global sources.

**ON (1)** Ignore power off settings for specified global sources.

Examples

```
SYST:PREF:SOUR2:GLOB:POFF:IGN 1
```

```
```

Query Syntax

SYSTem:PREFerences:SOURce<port>:GLOBal:POFF:IGNore[:STATe]?

Return Type

Boolean

Default OFF

---

SYSTem:PREFerences:SOURce<port>:GLOBal:POWer <num>
Applicable Models: All

(Read-Write) Set and return the power of the specified global source. Learn more.

**Parameters**

- `<port>` Source port number of the VNA.
- `<num>` Global source frequency. Choose any number between the `minimum` and `maximum` frequency limits of the analyzer. Units are Hz.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

**Query Syntax**

SYSTem:PREferences:SOURce<port>:GLOBal[:STATe] <bool>

**Applicable Models: All**

(Read-Write) Set and return the global state of the specified global source. Learn more.

**Parameters**

- `<port>` Source port number of the VNA.
- `<bool>` Choose from:
  - **OFF (0)** Turn global state of the specified source OFF.
  - **ON (1)** Turn global state of the specified source ON.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

**Query Syntax**

SYSTem:PREferences:SOURce<port>:GLOBal[:STATe]

**Return Type**

- **Numeric**
- **Boolean**

**Default**

- OFF
SYSTem:TDR Commands

|------------|----------------------------------|----------------------------------------|

Click on a red keyword to view the command details.

See Also

- Synchronizing the Analyzer and Controller
- SCPI Command Tree

SYS Te m:TDR:CAPability:FREQuency:MAXimum <freq>

**Applicable Models:** All with TDR Options (S9x011A/B)

*(Read-Write)* Set/Get stop limit frequency in TDR app. The value change gets effective after TDR preset.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;freq&gt;</td>
<td>Stop limit frequency</td>
</tr>
</tbody>
</table>

**Examples**

```
SYST:TDR:CAP:FREQ:MAX 9E6
system:tdr:capability:frequency:maximum?
```

**Query Syntax**

```
SYSTem:TDR:CAPability:FREQuency:MAXimum ]?
```

**Return Type**

Numeric

**Default**

Same as the product default stop frequency

SYS Te m:TDR:CAPability:FREQuency:MINimum <freq>
Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) Set/Get start limit frequency in TDR app. The value change gets effective after TDR preset.

Parameters

<freq> Start limit frequency

Examples

```
SYST:TDR:CAP:FREQ:MIN 9E3
system:tdr:capability:frequency:minimum?
```

Query Syntax

```
SYSTem:TDR:CAPability:FREQuency:MINimum ]?
```

Return Type

Numeric

Default

Same as the product default start frequency

SYSTem:TDR:PRESet

Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) This command executes a TDR preset.

Examples

```
SYST:TDR:PRES
system:tdr:preset
```

Default

Not Applicable
**SYSTem:UNCertainty**

Contains the settings to create and control Dynamic Uncertainty for S-Parameters (Opt. S93015A/B).

### Setup Options

<table>
<thead>
<tr>
<th>Uncertainty Options</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calibration Options</td>
<td></td>
</tr>
<tr>
<td>Noise</td>
<td>SYST:UNC:ETER:NOIS:ENAB</td>
</tr>
<tr>
<td>Repeatability</td>
<td>SYST:UNC:ETER:CABL:REP</td>
</tr>
<tr>
<td>Standard Definition</td>
<td>SYST:UNC:ETER:SDEF</td>
</tr>
<tr>
<td>Max Uncertainty Pts</td>
<td></td>
</tr>
<tr>
<td></td>
<td>500</td>
</tr>
</tbody>
</table>

| Service Mode                                             |               |
|                                                           |               |
|                                                           |               |
|                                                           |               |
|                                                           |               |

### Noise Characterization

| Clear noise data on specified port                       | SYST:UNC:PORT<p>:NOISE:RESET |
| Clear noise data on all ports                           | SYST:UNC:PORT:NOISE:RESET    |
| Copy noise from a port to all ports                     | SYST:UNC:PORT:NOISE:ALL:COPY |

### Cables Characterization

| List cables                                              | SYST:UNC:CABLE:CATalog? |
| Assign Cable to all ports                               | SYST:UNC:PORT:CABLE:ALL |
| Assign Cable to specified port                          | SYST:UNC:PORT<p>:CABLE   |
| Reset repeatability                                     | SYST:UNC:CABL:REP:RES    |

### Uncertainty workspace

| Load workspace                                           | SYST:UNC:LOAD |
| Save workspace                                          | SYST:UNC:STORe |

### Enable a Guided Cal to include Uncertainties

| Checkbox on Guided Cal Select Ports page                 | SENS:CORR:COLL:GUID:UNC |
Trace Properties

**Uncertainty Trace Property**

<table>
<thead>
<tr>
<th>Trace Type</th>
<th>Settings</th>
<th>Add Trace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td></td>
<td>[UMax]</td>
</tr>
<tr>
<td>UMax</td>
<td>Noise</td>
<td>[UMin]</td>
</tr>
<tr>
<td>UMin</td>
<td>Freeprobability</td>
<td></td>
</tr>
<tr>
<td>UBars</td>
<td>Calibration</td>
<td></td>
</tr>
<tr>
<td>Shade</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ellipses</td>
<td>[Apply to all traces]</td>
<td></td>
</tr>
</tbody>
</table>

Apply to all traces | None
Add Trace | None
Save uncertainty data | CALC:MEAS:UNC:SAVE

**Limitations**

- Calibrations can be performed for ONLY ONE channel at a time.
- Putting Error Term data into Uncertainty Cal Sets using remote commands is NOT supported.

**See Also**

- Trace Commands for Dynamic Uncertainty
- Learn more about Dynamic Uncertainty
- Example Program
- Guided Cal commands
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

SYSTem:UNCertainty:CABLE:CATalog?
Applicable Models: N522xB, N523xB, N524xB

(Read-only) Returns a comma-delimited list of names of cables that are defined in the Uncertainty Manager application.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>SYST:UNC:CABL:CAT?</td>
</tr>
<tr>
<td>Return Type</td>
<td>Comma-delimited string</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

SYSTem:UNCertainty:CABLE:REPeat:RESet <cableName>

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Resets (clears) the characterized repeatability data associated with the specified cable.

Parameters
<\cableName> String. Name of the cable for which data is to be reset.

| Examples     | SYST:UNC:CABL:REP:RES "MyCable" |
| Query Syntax | Not Applicable |
| Default      | Not Applicable |

SYSTem:UNCertainty:PORT:CABLE:ALL[:SELect] <cableName>

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Sets the name of the cable to be associated with all the ports currently enabled on the VNA.

Parameters
<\cableName> String. Name of the cable.

| Examples     | SYST:UNC:PORT:CABL:ALL "MyCable" |
| Query Syntax | Not Applicable |
| Default      | Not Applicable |

SYSTem:UNCertainty:PORT<pNum>:CABLe[:SELect] <cableName>
Applicable Models: N522xB, N523xB, N524xB

(Write-Read) Sets and returns the name of the cable to be associated with the specified port number on the VNA

Parameters

- `<pNum>` VNA port number.
- `<cableName>` String. Name of the cable.

Examples

```
SYST:UNC:PORT3:CABL "MyCable"
```

Query Syntax

```
SYSTem:UNCertainty:PORT<pNum>:CABLe[:SELect]?
```

Return Type

String

Default

Not Applicable

---

**SYSTem:UNCertainty:PORT:NOISe:ALL:COPY <pNum>**

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Copies the characterized noise data associated with the specified port, to all the other ports

Parameters

- `<pNum>` VNA port number for which noise data will be copied.

Examples

```
SYST:UNC:PORT:NOIS:ALL:COPY 2
```

Query Syntax

Not Applicable

Default

Not Applicable

---

**SYSTem:UNCertainty:PORT:NOISe:ALL:RESet**

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Resets (clears) the characterized noise data for all currently enabled VNA ports.

Parameters

None

Examples

```
SYST:UNC:PORT:NOIS:ALL:RESet
```

Query Syntax

Not Applicable

Default

Not Applicable
SYSTem:UNCertainty:PORT<pNum>:NOISe:RESet

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Resets (clears) the characterized noise data for the specified VNA port.

Parameters
None

<pNum> VNA port number for which noise data will be reset.

Examples
SYST:UNC:PORT2:NOIS:RESet

Query Syntax Not Applicable

Default Not Applicable

SYSTem:UNCertainty:LOAD <filename>

Applicable Models: N522xB, N523xB, N524xB

(Write-only) Loads an uncertainty ‘workspace’ (*.ml4) file into the Uncertainty Manager.

Parameters
None

<filename> String. Full path, filename, and extension of the uncertainty workspace file, enclosed in quotes.

Examples
SYST:UNC:LOAD "C:\MyUncert.ml4"

Query Syntax Not Applicable

Default Not Applicable

SYSTem:UNCertainty:STORe [filename]
**Applicable Models:** N522xB, N523xB, N524xB

(Write-only) Saves the current uncertainty ‘workspace’ of the Uncertainty Manager to a (‘.ml4’) file.

**Parameters**
- None

  If filename is not specified, the current workspace is saved to the default workspace (*.ml4) file.

**Examples**

```
SYST:UNC:STORe "C:\MyUncert.ml4"
```

**Query Syntax**
- Not Applicable
- **Default**

---

**SYSTM:UNCertainty:POINts:MAXimum <num>**

**Applicable Models:** N522xB, N523xB, N524xB

(Write-Read) Sets and returns the maximum number of points (“decimation value”) for which uncertainties are to be computed for subsequent calibrations that are performed using Dynamic Uncertainty for S-Parameters.

**Parameters**
- <num> Max number of points. Specify an integer between 0 and 501.

**Examples**

```
SYST:UNC:POIN:MAX 201
```

**Query Syntax**
- SYSTM:UNCertainty:POINts:MAXimum?
- **Return Type**
- Numeric
- **Default**

---

**SYSTM:UNCertainty:ETERm:NOISe:ENABle <bool>**
**Applicable Models:** N522xB, N523xB, N524xB

*(Write-Read)* Sets and returns the ON/OFF state of allowing noise data to contribute to the uncertainty of a calibration performed using Dynamic Uncertainty for S-Parameters. Noise data must also be present for the ports at the time the calibration is performed.

**Parameters**

<bool> Enable ON/OFF state. Choose from:

- **ON** or 1 - Noise uncertainty ON.
- **OFF** or 0 - Noise uncertainty OFF.

**Examples**

```
SYST:UNC:ETER:NOIS:ENAB ON
```

**Query Syntax**

SYSTem:UNCertainty:ETERm:NOISe:ENABle?

**Return Type**

Boolean

**Default**

ON

---

**SYSTem:UNCertainty:ETERm:CABLe:REPeat[:ENABle] <bool>**

**Applicable Models:** N522xB, N523xB, N524xB

*(Write-Read)* Sets and returns the ON/OFF state of allowing cable repeatability data to contribute to the uncertainty of a calibration performed using Dynamic Uncertainty for S-Parameters. Repeatability data must also be present for the ports at the time the calibration is performed.

**Parameters**

<bool> Enable ON/OFF state. Choose from:

- **ON** or 1 - Cable repeatability uncertainty ON.
- **OFF** or 0 - Cable repeatability uncertainty OFF.

**Examples**

```
SYST:UNC:ETER:CABL:REP ON
```

**Query Syntax**

SYSTem:UNCertainty:ETERm:CABLe:REPeat[:ENABle]?

**Return Type**

Boolean

**Default**

ON

---

**SYSTem:UNCertainty:ETERm:SDEFinitions[:ENABle] <bool>**
**Applicable Models:** N522xB, N523xB, N524xB

*(Write-Read)* Sets and returns the ON/OFF state of allowing the uncertainty associated with the standard definitions in the cal kits to contribute to the uncertainty of a calibration performed using Dynamic Uncertainty for S-Parameters. The uncertainty data for the Cal standards must also be present at the time the calibration is performed.

**Parameters**

<`bool>` Enable ON/OFF state. Choose from:

- **ON** or **1** - Standard definition uncertainty ON.
- **OFF** or **0** - Standard definition uncertainty OFF.

**Examples**

```
SYST:UNC:ETER:SDEF ON
```

**Query Syntax**

SYSTem:UNCertainty:ETERm:SDEFinitions[:ENABle]?

**Return Type**

Boolean

**Default**

ON
Trigger Commands

Controls External Triggering.

```
TRIGger:  
  AUXiliary  
    | COUNt  
  CHANnel:AUXiliary  
    | DELay  
    | DURation  
    | [ENABLE]  
    | HANDshake  
    | INPut  
      | DELay  
      | HANDshake  
      | POLarity  
      | TYPE  
    | INTERVAL  
    | IPOLarity  
    | OPOLarity  
  | OUTPUT  
    | Delay  
    | DURation  
    | INTERVAL  
    | POLarity  
    | POSition  
    | POSition  
    | TYPE  
   DELay
```
Click on a keyword to view the command details.

Blue commands are superseded.

See Also

- **Example program** Triggering the VNA
- See other SCPI Triggering commands
- Learn about External / Aux Triggering
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

TRIGger:AUXiliary:COUNt?
Applicable Models: N522xB, N523xB, N524xB, E5080A, M937xA, M9485A, P937xA

(Read-only) Returns the number of AUX trigger input / output connector pairs in the instrument.

Parameters

Examples

TRIG:AUX:COUN?
trigger:auxiliary:count?

Return Type
Numeric

Default
Not Applicable

TRIGger:CHANnel<ch>:AUXiliary<n>:DURation <num> Superseded

Applicable Models: N522xB, N523xB, N524xB, M937xA

This command is replaced with TRIG:CHAN:AUX:INP:DEL.

(Read-Write) Specifies the delay that should be applied by the VNA after the Aux trigger input is received and before the acquisition is made.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> AUX Trigger connector used to send or receive signals.

Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)

If unspecified, value is set to 1.

<num> Delay value in seconds. Choose a value between 0 and 3.0 seconds.

Examples

TRIG:CHAN:AUX:DEL 0.5
trigger:channel2:aux2:delay 1.5

Query Syntax
TRIGger:CHANnel<ch>:AUXiliary<n>:DURation<num>?

Return Type
Numeric

Default
0

TRIGger:CHANnel<ch>:AUXiliary<n>:DURation <num> Superseded
**Applicable Models:** N522xB, N523xB, N524xB, E5080A, M937xA, M9485A

This command is replaced with `TRIG:CHAN:AUX:OUTP:DUR`.

(Read-Write) Specifies the width of the output pulse, which is the time that the Aux trigger output will be asserted.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1.
- `<n>`: AUX Trigger connector used to send or receive signals.

Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)

If unspecified, value is set to 1.

- `<num>`: Duration value in seconds. Choose a value between 1us (1E-6) and 1

**Examples**

```
TRIG:CHAN:AUX:DUR .1
trigger:channel2:aux2:duration .01
```

**Query Syntax**

`TRIGger:CHANnel<ch>:AUXiliary<n>:DURation?`

**Return Type**

Numeric

**Default**

1E-6

---

`TRIGger:CHANnel<ch>:AUXiliary<n>[:ENABle] <bool>`

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, E5080A, M9485A

(Read-Write) Turns ON / OFF the trigger output.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1.
- `<n>`: AUX Trigger connector used to send or receive signals.

Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)

If unspecified, value is set to 1.

- `<bool>`: ON (or 1) - turns trigger output ON.

**Examples**

```
TRIG:CHAN:AUX 1
trigger:channel2:aux2:enable off
```
TRIGger:CHANnel<ch>:AUXiliary<n>:HANDshake <bool> Superseded

**Applicable Models:** N522xB, N523xB, N524xB

This command is replaced with TRIG:CHAN:AUX:INP:HAND.

*(Read-Write)* Turns handshake ON / OFF.

To enable handshake, the main trigger enable must also be set using TRIG:CHAN:AUX:ENAB.

When ON, VNA waits indefinitely for the input line to be asserted before continuing with the acquisition. When OFF, the VNA acquires data without waiting.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` AUX Trigger connector used to send or receive signals.
  - Choose from 1 *(AUX TRIG 1)* or 2 *(AUX TRIG 2)*
  - If unspecified, value is set to 1.
- `<bool>`
  - **ON** (or 1) - turns handshaking ON.
  - **OFF** (or 0) - turns handshaking OFF.

**Examples**

```
TRIG:CHAN:AUX:HAND 1
trigger:channel2:aux2:handshake off
```

**Query Syntax**

TRIGger:CHANnel<ch>:AUXiliary<n>:HANDshake?

**Return Type** Boolean

**Default** OFF
Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Specifies the delay that should be applied by the VNA after the Aux trigger input is received and before the acquisition is made.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` AUX Trigger connector used to send or receive signals. Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)
  - If unspecified, value is set to 1
- `<num>` Delay value in seconds. Choose a value between 0 and 3.0 seconds.

Examples

```
TRIG:CHAN:AUX:INP:DEL .5
```
```
trigger:channel2:aux:input:delay 1.5
```

Query Syntax

TRIGger:CHANnel<ch>:AUXiliary<n>:INPut:DELay?

Return Type

Numeric

Default

0

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Turns handshake ON / OFF.

To enable handshake, the main trigger enable must also be set using TRIG:CHAN:AUX:ENAB.

When ON, VNA waits indefinitely for the input line to be asserted before continuing with the acquisition. When OFF, the VNA acquires data without waiting.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` AUX Trigger connector used to send or receive signals.
  - Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)
  - If unspecified, value is set to 1
- `<bool>`
  - ON (or 1) - turns handshaking ON.
  - OFF (or 0) - turns handshaking OFF.
Examples

```
TRIG:CHAN:AUX:INP:HAND 1
trigger:channel2:aux:input:handshake off
```

Query Syntax

TRIGger:CHANnel<ch>:AUXiliary<n>:INPut:HANDshake?

Return Type

Boolean

Default

OFF

TRIGger:CHANnel<ch>:AUXiliary<n>:INPut:POLarity <char>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Specifies the polarity of the trigger IN signal to which the VNA will respond.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> AUX Trigger connector used to send or receive signals.

Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)

If unspecified, value is set to 1.

<char> Choose from:

- POSitive VNA responds to leading edge or HIGH level
- NEGative VNA responds to trailing edge or LOW level.

Set Edge or Level triggering using TRIG:CHAN:AUX:TYPE

Examples

```
TRIG:CHAN:AUX:INP:POL POS
trigger:channel2:aux2:input:polarity negative
```

Query Syntax

TRIGger:CHANnel<ch>:AUXiliary<n>:INPut:POLarity?

Return Type

Character

Default

NEGative

TRIGger:CHANnel<ch>:AUXiliary<n>:INPut:TYPE <char>
**Applicable Models:** N522xB, N523xB, N524xB

*(Read-Write)* Specifies the type of Aux input detection that the VNA will employ.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` AUX Trigger connector used to send or receive signals.
  
  Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)

  If unspecified, value is set to 1.
- `<char>` Choose from:
  
  - **EDGE** VNA responds to the leading edge of a signal
  - **LEVEL** VNA responds to the level (HIGH or LOW) of a signal

**Examples**

```
TRIG:CHAN:AUX:INP:TYPE EDGE
```

```
trigger:channel2:aux:input:type level
```

**Query Syntax**

TRIGger:CHANnel<ch>:AUXiliary<n>:INPut:TYPE?

**Return Type**

Character

**Default**

EDGE

---

**TRIGger:CHANnel<ch>:AUXiliary<n>:INTerval <char> Superseded**

*Applicable Models:* N522xB, N523xB, N524xB, M937xA, E5080A/B, M9485A, M98xxA, P50xxA

This command is replaced with `TRIG:CHAN:AUX::OUTP:INT`.

*(Read-Write)* Specifies how often a trigger output signal is sent.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` AUX Trigger connector used to send or receive signals.

  Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)

  If unspecified, value is set to 1.
- `<char>` Choose from:
- **POINT** Trigger signal is sent every data point. (effectively the same as Point sweep)

- **SWEep** Trigger signal is sent once every sweep.

**Examples**

```
TRIG:CHAN:AUX:INT POI
```

```
trigger:channel2:aux2:interval sweep
```

**Query Syntax**  
TRIGger:CHANnel<ch>:AUXiliary<n>:INTerval?

**Return Type**  
Character

**Default**  
SWEep

TRIGger:CHANnel<ch>:AUXiliary<n>:IPOLarity <char> **Superseded**

**Applicable Models:** N522xB, N523xB, N524xB

This command is replaced with **TRIG:CHAN:AUX::INP:POL**.

**Parameters**

- **<ch>** Any existing channel number. If unspecified, value is set to 1.
- **<n>** AUX Trigger connector used to send or receive signals.
  - Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)
  - If unspecified, value is set to 1.
- **<char>** Choose from:
  - **POSitive** VNA responds to leading edge or HIGH level
  - **NEGative** VNA responds to trailing edge or LOW level.

**Examples**

```
TRIG:CHAN:AUX:IPOL POS
```

```
trigger:channel2:aux2:ipolarity negative
```

**Query Syntax**  
TRIGger:CHANnel<ch>:AUXiliary<n>:IPOLarity?

**Return Type**  
Character

**Default**  
NEGative
TRIGger:CHANnel<ch>:AUXiliary<n>:OPOLarity <char> Superseded

Applicable Models: N522xB, N523xB, N524xB, M937xA, E5080A/B, M9485A, M98xxA, P50xxA

This command is replaced with TRIG:CHAN:AUX::OUTP::POL.

(Read-Write) Specifies the polarity of the Aux Output signal being supplied by the VNA.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> AUX Trigger connector used to send or receive signals.

Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)

If unspecified, value is set to 1.

<char> Choose from:

- POSitive VNA sends positive going pulse.
- NEGative VNA sends negative going pulse.

Examples

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRIG:CHAN:AUX::OPOL NEG</td>
</tr>
<tr>
<td>trigger:channel2:aux2:opolarity positive</td>
</tr>
</tbody>
</table>

Query Syntax TRIGger:CHANnel<ch>:AUXiliary<n>:OPOLarity?

Return Type Character

Default NEGative

TRIGger:CHANnel<ch>:AUXiliary<n>:OUTPut:DELay <num>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, E5080A/B, M9485A, M98xxA, P50xxA

(Read-Write) Specifies the delay that should be applied by the VNA after the Aux trigger output is output and before the acquisition is made.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` AUX Trigger connector used to send or receive signals.
  
  Choose from **1** (AUX TRIG 1) or **2** (AUX TRIG 2)
  
  If unspecified, value is set to 1.

- `<num>` Delay value in seconds. Choose a value between 0 and 1 seconds.

**Examples**

```
TRIG:CHAN:AUX:OUTP:DEL .5
trigger:channel2:aux:output:delay 1.5
```

**Query Syntax**

TRIGger:CHANnel<ch>:AUXiliary<n>:OUTPut::DELay?

**Return Type** Numeric

**Default** 0

---

**TRIGger:CHANnel[ch]:AUXiliary[n]:OUTPut:DURation <num>**

(Read-Write) Specifies the width of the output pulse, which is the time that the Aux trigger output will be asserted.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` AUX Trigger connector used to send or receive signals.
  
  Choose from **1** (AUX TRIG 1) or **2** (AUX TRIG 2)
  
  If unspecified, value is set to 1.

- `<num>` Duration value in seconds. Choose a value between 1us (1E-6) and 1

**Examples**

```
TRIG:CHAN:AUX:OUTP:DUR .1
trigger:channel2:aux:output:duration .01
```

**Query Syntax**

TRIGger:CHANnel<ch>:AUXiliary<n>:OUTPut:DURation?

**Return Type** Numeric
TRIGger:CHANnel<ch>:AUXiliary<n>:OUTPut:INTerval <char>

Applicable Models: N522xB, N523xB, N524xB, M937xA, E5080A/B, M9485A, M98xxA, P50xxA

(Read-Write) Specifies how often a trigger output signal is sent.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> AUX Trigger connector used to send or receive signals.

Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)

<char> Choose from:

- **POINT** Trigger signal is sent every data point. (effectively the same as Point sweep)
- **SWEep** Trigger signal is sent once every sweep.

Examples

```
TRIG:CHAN:AUX:OUTP:INT POI
trigger:channel2:aux:output:interval sweep
```

Query Syntax TRIGger:CHANnel<ch>:AUXiliary<n>:OUTPut:INTerval?

Return Type Character

Default SWEep

---

TRIGger:CHANnel<ch>:AUXiliary<n>:OUTPut:POLarity <char>
**Applicable Models:** N522xB, N523xB, N524xB, M937xA, E5080A/B, M9485A, M98xxA, P50xxA

*(Read-Write)* Specifies the polarity of the Aux Output signal being supplied by the VNA.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` AUX Trigger connector used to send or receive signals.
  
  Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)

  If unspecified, value is set to 1.
- `<char>` Choose from:
  
  - **POSitive** VNA sends positive going pulse.
  - **NEGative** VNA sends negative going pulse.

**Examples**

```
TRIG:CHAN:AUX:OUTP:POL NEG
trigger:channel2:aux:output:polarity positive
```

**Query Syntax**

TRIGger:CHANnel<ch>:AUXiliary<n>:OUTPut:POLarity?

**Return Type**

Character

**Default** NEGative

---

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, E5080A/B, M9485A, M98xxA, P50xxA

*(Read-Write)* Specifies whether the aux trigger out signal is sent BEFore or AFTer the acquisition.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` AUX Trigger connector used to send or receive signals.
  
  Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)

  If unspecified, value is set to 1.
- `<char>` Choose from:
  
  - **BEFore** Use if the external device needs to be triggered before the data is acquired, such as a power meter.
- **AFTer** Use if the external device needs to be triggered just after data has been acquired, such as an external source. This could be more efficient since it allows the external device to get ready for the next acquisition at the same time as the VNA.

**Examples**

```
TRIG:CHAN:AUX:OUTP:POS BEF
```

```
trigger:channel2:aux:output:position after
```

**Query Syntax**

```
TRIGger:CHANnel<ch>:AUXiliary<n>:OUTPut:POSition?
```

**Return Type**

Character

**Default** AFTer

---

**TRIGger:CHANnel<ch>:AUXiliary<n>:POSition <char> Superseded**

**Applicable Models:** N522xB, N523xB, N524xB, M937xA, E5080A/B, M9485A, M98xxA, P50xxA

This command is replaced with `TRIG:CHAN:AUX::OUTP:POS`.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` AUX Trigger connector used to send or receive signals.
  
  Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)

  If unspecified, value is set to 1.
- `<char>` Choose from:
  
  - **BEF**ore Use if the external device needs to be triggered before the data is acquired, such as a power meter.
  
  - **AFT**er Use if the external device needs to be triggered just after data has been acquired, such as an external source. This could be more efficient since it allows the external device to get ready for the next acquisition at the same time as the VNA.

**Examples**

```
TRIG:CHAN:AUX:POS BEF
```

```
trigger:channel2:aux2:position after
```

**Query Syntax**

```
TRIGger:CHANnel<ch>:AUXiliary<n>:POSition?
```

**Return Type**

Character
TRIGger:CHANnel<ch>:AUXiliary<n>:TYPE  <char> Superseded

Appllicable Models: N522xB, N523xB, N524xB

This command is replaced with TRIG:CHAN:AUX::INP:TYPE.

(Read-Write) Specifies the type of Aux input detection that the VNA will employ.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> AUX Trigger connector used to send or receive signals.

Choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)

If unspecified, value is set to 1.

<char> Choose from:

- **EDGE** VNA responds to the leading edge of a signal
- **LEVel** VNA responds to the level (HIGH or LOW) of a signal

Examples

- TRIG:CHAN:AUX:TYPE EDGE
- trigger:channel2:aux2:type level

Query Syntax

TRIGger:CHANnel<ch>:AUXiliary<n>:TYPE?

Return Type

Character

Default

EDGE
Applicable Models: All

(Read-Write) Sets and reads the trigger delay for ALL channels (globally). This delay is only applied while TRIG:SOURce = EXTernal and TRIG:SCOP = ALL. After an external trigger is applied, the start of the sweep is held off for an amount of time equal to the delay setting plus any inherent latency.

To apply a trigger delay for the specified channel ONLY, use SENS:SWE:TRIG:DELay

**Parameters**

- `<num>` Delay value in seconds. Choose from 0 to 3.

**Examples**

TRIG:DEL .0003

Sets the trigger delay to 300 microseconds. The sweep will not start until approximately 300 microseconds after an external trigger is applied.

**Query Syntax** TRIGger:DELay?

**Return Type** Numeric

**Default** 0

---

TRIGger:PREFerence:ALGロbal <bool>

**Applicable Models:** N522xB, N523xB, N524xB, E5080A, M937xA, M9485A, P937xA

(Read-Write) Sets the Trigger OUT behavior to either Global or Channel. Learn more about this setting.

This command will cause the VNA to Preset.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

To send this command using the VNA GUI, open the GPIB Command Processor Console, then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the VNA took the command.

**Parameters**

- `<bool>` Choose from:

  - **ON** (or 1) - Trigger properties apply to ALL channels (Global).
    - Allows use of CONT:SIGNal command to configure the external trigger properties.
    - "Per Point" trigger property is not settable. Use the channel's Point trigger setting.
- **OFF** (or 0) - External Trigger properties apply to each channel independently.
- Must use TRIG:CHAN:AUX commands to configure the external trigger properties. CONT:SIGNal will NOT work.
- "Per Point" trigger output property is set using the channel's Point trigger setting AND TRIG:CHAN:AUX:INTerval.

### Examples

```
TRIG:PREF:AIGL 1
trigger.preference:aiglobal 0
```

### Query Syntax

TRIGger:PREFerence:AIGLobal?

### Return Type

Boolean

### Default

0

---

**TRIGger:READy:POLarity <char>**

**Applicable Models:** N522xB, N523xB, N524xB, E5080A/B, M937xA, P937xA, M98xxA, P50xxA

(Read-Write) Specifies the polarity of Ready for Trigger output.

All existing Ready for Trigger outputs are configured simultaneously with this command.

### Parameters

- **<char>**
  - LOW - Outputs a TTL low when the VNA is ready for trigger.
  - HIGH - Outputs a TTL high when the VNA is ready for trigger.

### Examples

```
TRIG:READ:POL HIGH
trigger:ready:polarity low
```

### Query Syntax

TRIGger:READy:POLarity?

### Return Type

Character

### Default

Low

---

**TRIGger:READy:SOURce:MANual:ENABle <bool>**
Applicable Models: M980xA, P50xxA, E5080B

(Read-Write)  Sets the mode of Ready for Trigger output. This allows you to have a same operation with M937xA and P937xA. This setting is not saved/recalled in the status file.

Parameters
<bool>  Choose from:

- **ON** (or 1) - Outputs ready for trigger when trigger source is external or manual. Same operation with M937xA and P937xA.
- **OFF** (or 0) - Outputs ready for trigger when trigger source is external.

Examples

```
TRIG:READ:SOUR:MAN:ENAB 1
trigger:ready:source:manual:enable 0
```

Query Syntax  TRIGger:READy:SOURce:MANual:ENABle?
Return Type  Boolean
Default  0 (Preset will not reset the setting.)

---

TRIGger[:SEQUence]:LEVEL <char> - Superseded

This command is replaced with CONTrol:SIGNal

(Read-Write)  Triggers either on a **High** or **Low** level trigger signal. This setting only has an effect when TRIG:SOURce EXTERNAL is selected.

Parameters
<char>  Choose from:

- **HIGH** - analyzer triggers on TTL High
- **LOW** - analyzer triggers on TTL Low

Examples

```
TRIG:LEV HIGH
trigger:sequence:level low
```

Query Syntax  TRIGger[:SEQUence]:LEVEL?
Return Type  Character
Default  HIGH
TRIGger[:SEQunce]:ROUTE:INPut <char>

Applicable Models: All

(Read-Write) Specifies the connector to use for the external trigger input.

Parameters

<char> Choose from:

- MAIN - Meas Trig In BNC
- MATH - handler I/O Pin 18
- PULSE3 - Internal routing of pulse 3 output to the MEAS TRIG IN on the rear pane

PXI/USB VNA

- SMB – Meas Trig In SMB

(The M9370A/71A/72A/73A/74A/75A support the following parameters for backplane trigger.)

- DSTARB – Backplane Trigger Lines (PXIe DSTARB)
- STAR – Backplane Trigger Lines (PXI STAR)
- TRIG0 – Backplane Trigger Lines (PXI TRIG0)
- TRIG1 – Backplane Trigger Lines (PXI TRIG1)
- TRIG2 – Backplane Trigger Lines (PXI TRIG2)
- TRIG3 – Backplane Trigger Lines (PXI TRIG3)
- TRIG4 – Backplane Trigger Lines (PXI TRIG4)
- TRIG5 – Backplane Trigger Lines (PXI TRIG5)
- TRIG6 – Backplane Trigger Lines (PXI TRIG6)
- TRIG7 – Backplane Trigger Lines (PXI TRIG7)

(The P50xxA support the following parameters for streamline Rear Trig
1/2 SMB port.)

NONE – Streamline Rear SMB Not connected (NONE)

REAR1 – Streamline Rear SMB (TRIG1)

REAR2 – Streamline Rear SMB (TRIG2)

Examples

TRIG:ROUTE:INP MAIN

trigger:sequence:route:input main

Query Syntax
TRIGger[:SEQuence]:ROUTE:INPut?

Return Type Character
Default MAIN

TRIGger[:SEQuence]:ROUTE:READy <char>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Specifies the connector to use for the trigger OUT ready line.

Parameters

<char> Choose from:

MAIN - Meas trig ready

MATH - handler pin 21

Examples

TRIG:ROUTE:READ MATH

trigger:sequence:route:ready math

Query Syntax
TRIGger[:SEQuence]:ROUTE:READy?

Return Type Character
Default MAIN

TRIGger[:SEQuence]:SCOpe <char>
Applicable Models: All

(Read-Write) Specifies whether a trigger signal is sent to all channels or only the current channel.

See Triggering the VNA using SCPI.

Parameters

<char> Choose from:

- **ALL** - trigger signal is sent to all channels. Also sets SENS:SWEep:TRIG:POINT OFF on ALL channels.
- **CURRENT** - trigger signal is sent to only one channel at a time. With each trigger signal, the channel is incremented to the next triggerable channel.
- **ACTIVE** - trigger signal is sent to active channel only.

Examples

```
TRIG:SCOP ALL
trigger:sequence:scope current
```

Query Syntax

TRIGger[:SEQuence]:SCOPe?

Return Type

Character

Default

ALL

---

TRIGger[:SEQuence]:SLOPe <char>

Applicable Models: All

(Read-Write) Specifies the polarity expected by the external trigger input circuitry. Also specify TRIG:TYPE (Level |Edge).

See Triggering the VNA using SCPI.

Parameters

<char> Choose from:

- **POSitive** (rising Edge) or High Level
- **NEGative** (falling Edge) or Low Level

Examples

```
TRIG:SLOP NEG
trigger:sequence:slope positive
```

Query Syntax

TRIGger[:SEQuence]:SLOPe?

Return Type

Character
TRIGger[:SEQuence]:SOURce <char>

Applicable Models: All

(Read-Write) Sets the source of the sweep trigger signal. This command is a super-set of INITiate:CONTinuous which can NOT set the source to External.

See Triggering the VNA using SCPI.

Parameters

<char> Choose from:

- **EXTERNAL** - external (rear panel) source.
- **IMMediate** - internal source sends continuous trigger signals
- **MANual** - sends one trigger signal when manually triggered from the front panel or INIT:IMM is sent.

Examples

```
TRIG:SOUR EXT
TRIGGER:SEQUENCE:SOURCE IMMEDIATE
```

Query Syntax

TRIGger[:SEQuence]:SOURce?

Return Type

Character

Default

IMMediate

---

TRIGger[:SEQuence]:TYPE <char>
Applicable Models: All

(Read-Write) Specifies the type of EXTERNAL trigger input detection used to listen for signals on the Meas Trig IN connectors. Edge triggers are most commonly used.

Parameters

<char> Choose from:

**EDGE** VNA responds to the rising and falling edge of a signal.

**LEVel** VNA responds to a level (HIGH or LOW).

Use TRIG:SLOPe to specify Rising or falling - High or Low.

Examples

```
TRIG:TYPE EDGE
trigger:sequence:type level
```

Query Syntax

TRIGger[:SEQuence]:TYPE?

Return Type

Character

Default

LEVel

---

TRIGger:STATus:READy? <char>

Applicable Models: N522xB, N523xB, N524xB, M9485A, E5080A

(Read-only) Checks if the VNA is ready for a hardware trigger.

This command is not intended to be used in a dynamic triggering situation where the ready status is constantly changing. Instead, the expected use is a more static situation where you are expecting the VNA to transition from not ready to ready, and then wait for a trigger. The VNA is polled until it becomes ready and then an operation that triggers the VNA is performed.

Note: This command is only supported on the PNA-L, PNA, and PNA-X with DSP5 installed. Any other model will return an error.

Parameters

<char> ANY - Check if the VNA is ready for any of the following hardware triggers.

**MEAS** - Check if the VNA is ready for an External trigger from the Meas Trig In BNC, Handler IO Pin 18, or Pulse 3 line.

**AUX1** - Check if the VNA is ready for a trigger from the AUX TRIG 1 IN on the rear panel. (PNA, ENA)
**AUX2** - Check if the VNA is ready for a trigger from the AUX TRIG 2 IN on the rear panel. (PNA, ENA)

**MANual** - Check if the VNA is ready for manual trigger.

**Examples**

```
TRIG:STAT:READ? MEAS
trigger:status:ready? aux1
```

**Return Type**  
Boolean

**Default**  
ANY
## SCPI Example Programs

<table>
<thead>
<tr>
<th>Setup Measurements</th>
<th>Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalog Measurements</td>
<td>Visual Basic</td>
</tr>
<tr>
<td>Create an S-parameter Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create a Balanced Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create an FOM Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Channels, Windows, and Measurements</td>
<td>VBScript</td>
</tr>
<tr>
<td>Setup Sweep Parameters</td>
<td>Visual Basic</td>
</tr>
<tr>
<td>Setup the Display</td>
<td>Visual Basic</td>
</tr>
<tr>
<td>Setup a Measurement</td>
<td>Visual C++</td>
</tr>
<tr>
<td>Triggering the Analyzer</td>
<td>VBScript</td>
</tr>
<tr>
<td>Setup RxLeveling</td>
<td>VBScript</td>
</tr>
<tr>
<td>Setup Phase Control</td>
<td>VBScript</td>
</tr>
<tr>
<td>Configure an External Source</td>
<td>VBScript</td>
</tr>
<tr>
<td>Configure a PMAR Device</td>
<td>VBScript</td>
</tr>
<tr>
<td>Setup Markers</td>
<td>VBScript</td>
</tr>
<tr>
<td>Setup PNOP and PSAT Markers</td>
<td>VBScript</td>
</tr>
</tbody>
</table>

### Calibrations

See Calibrating the VNA Using SCPI

- Calibrate All Channels  
  - VBScript
- Guided 2-Port or 4-Port Cal  
  - VBScript
- Guided 2-Port Comprehensive Cal  
  - VBScript
- Guided ECAL  
  - VBScript
- Guided Mechanical  
  - VBScript
- Guided 1-port Mechanical Cal on Port 2  
  - VBScript
- Guided TRL  
  - VBScript
- Guided Unknown Thru or TRL Cal (apply Delta Match Cal)  
  - VBScript
- Perform a Guided Cal using Multiple Power Sensors  
  - VBScript
- Perform a Guided QSOLT Cal  
  - VBScript
- Power Meter Uncertainty  
  - VBScript
- Global Delta Match Cal  
  - VBScript
- Unguided ECAL  
  - VBScript
- Unguided 2-port Mechanical Cal  
  - VBScript
- Unguided 1-port Mechanical Cal on Port 2  
  - VBScript
- Unguided 2-port Cal on a 4-Port VNA  
  - Visual Basic
- Unguided Thru Response Cals  
  - VBScript
- Perform a CalAllChannels Calibration  
  - VBScript

5142
Perform Unguided Cal on Multiple Channels
Perform an ECal User Characterization
Perform an ECAL Confidence Check
Perform a Source and Receiver Power Cal
Perform a Simple Source Power Cal
Perform a Source Power Cal with TWO Sensors
Dynamic Uncertainty
Upload a Source Power Cal
Perform a Sliding Load Cal
Load Error Terms during a Cal Sequence
Create a New Cal Kit
Modify a Calibration Kit

**Fixture Simulator**
Create fixturing function (impedance conversion and port matching) None
Create fixturing function (2-port deembed, port extension, port matching, impedance conversion) None
Create fixturing function (port impedance conversion and port extension) None

**Applications**
Active (Hot) Parameters
Create and Cal a Noise Figure Measurement VBScript
Create and Cal an NFX Measurement VBScript
Setup Noise Figure Port Mapping VBScript
Create and Cal a GCA Measurement VBScript
Create and Cal a GCX Measurement VBScript
Create an iTMSA Measurement VBScript
Create and Cal an IMD Measurement VBScript
Create a Swept IMDX Measurement VBScript
Create a Diff I/Q Measurement VBScript
Create a Spectrum Analyzer Measurement VBScript

Modulation Distortion Measurement
- Measurement Setup C#
- Measurement Setup Converters C#
- Create Modulation File VBScript
- Display Data Setup C#
- Source Modulation Calibration C#
Create a Multi-Dimensional Sweep for a Spectrum Analyzer Channel VBScript
Phase Noise Measurement
| Setting Up a Phase Noise Measurement | VBScript |
| Setting Up a Source | VBScript |
| Spurious Measurement | VBScript |
| Integrated Noise Measurement | VBScript |
| Spot Noise Measurement | VBScript |
| TDR Measurements | Excel VBA |

**FCA**

| Create and Cal a VMC Measurement | VBScript |
| Perform a VMC Mixer Characterization | VBScript |
| Create and Cal an SMC Measurement | VBScript |
| Setup an FCA Segment Sweep | VBScript |
| Create and Cal Multiple SMC Channels | VBScript |
| Create an SMC Fixed Output Measurement | VBScript |
| Create SMC Embedded LO Measurement | VBScript |
| Perform an SMC Phase Ref Cal | VBScript |
| Use an Existing Power Cal During an SMC Cal | VBScript |
| Create and Cal a VMC Measurement using Session commands | VBScript |
| Superseded | |
| Create and Cal an SMC Measurement using Session commands | VBScript |
| Superseded | |

**Pulsed App.**

| Create a Pulse Profile Measurement | VBScript |
| Create a Wideband Pulse Measurement | Visual Basic |
| Create a Narrowband Point-in-Pulse Measurement using DLL | VBScript |
| Create a Narrowband Pulse Profile Measurement using DLL | VBScript |
| Pulse Narrowband Setup | Python |

**Automatic Fixture Removal (AFR)**

| AFR Using One Differential 2X THRU | VBScript |
| AFR Using One Differential OPEN | VBScript |
| AFR Using One Single-Ended 2X THRU | VBScript |
| AFR Using One Single-Ended OPEN | VBScript |

**Source Modulation**

| Noise Power Ratio (NPR) Correction | None |

**Miscellaneous**

| *ESR? Sweep Complete | VBScript |
| Getting and Putting Data | Visual Basic |
| Getting and Putting Data (Definite Block Transfers) | RMB |
| External Test Set Control (N44xx) | VBScript |
Transfer Data using MMEM:TRANSfer
Establish a VISA Session
Status Reporting
Create a Custom Power Meter Driver
GPIB Pass-Through
VISA Pass-Through
VNA as Controller and Talker/Listener
Send SCPI commands using a Socket Client
Setup FastCW and FIFO
Upload and Download a Segment List
Perform Linear Interpolation
Set ECal States

RMB
Visual Basic
Visual Basic
Visual Basic
VBScript
VBScript
Visual Basic
C#
VBScript
VBScript
VBScript
VBScript

See more programming information and examples at: http://na.support.keysight.com/pna/programming/
AFR Using One Differential 2X THRU

This VBScript program performs AFR using one differential 2X THRU, saves the fixture data to a file, then performs deembedding.

Each VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the code into a text editor file, such as Notepad, and save it on the PNA hard drive as *.vbs.

Learn how to setup and run the macro.

See Other SCPI Example Programs

Dim app
Dim scpi
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

'The 2X THRU data file should already exist
thruFile = "S:\case_02\AA.s4p"
'Fixture A and B data file will be created or overwritten
fixaFile = "S:\case_02\demo_fix_a.s4p"
fixbFile = "S:\case_02\demo_fix_b.s4p"

'AFR extracts fixture files
scpi.Execute("AFR:INITialize")
scpi.Execute("AFR:FIXTure:INPuts DIFFerential")
scpi.Execute("AFR:FIXTure:MEASurement 4")
scpi.Execute("AFR:STANdard:USE THRU,1")
scpi.Execute("AFR:STANdard:LOAD THRU," & Q(thruFile))
scpi.Execute("AFR:SAVE:PORTs VNA")
scpi.Execute("AFR:SAVE:FILename " & Q(fixaFile) & "," & Q(fixbFile))
opc = scpi.Execute("*OPC?"")

'Deembedding
scpi.Execute("CALC:FSIM:EMB:TYPE C")
scpi.Execute("CALC:FSIM:EMB:TOP:C:PORT 1,2,3,4")
scpi.Execute("CALC:FSIM:EMBed:NETwork1:FILename " & Q(fixaFile))
scpi.Execute("CALC:FSIM:EMBed:NETwork1:PMAP 1,2,3,4")
scpi.Execute("CALC:FSIM:EMBed:NETwork1:TYPE DEEmbed")
scpi.Execute("CALC:FSIM:EMBed:NETwork2:FILename " & Q(fixbFile))
scpi.Execute("CALC:FSIM:EMBed:NETwork2:PMAP 1,2,3,4")
scpi.Execute("CALC:FSIM:EMBed:NETwork2:TYPE DEEmbed")
scpi.Execute("CALC:FSIM:EMBed:STAtE ON")
scpi.Execute("CALC:FSIM:STAtE ON")
opc = scpi.Execute("*OPC?"")

Set scpi = Nothing
Set app = Nothing
'Add double quotation marks to a string
Function Q(s)
  Q = Chr(34) & s & Chr(34)
End Function
AFR Using One Differential OPEN

This VBScript program performs AFR using one differential OPEN, saves the fixture data to a file, then performs deembedding.

Each VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the code into a text editor file, such as Notepad, and save it on the PNA hard drive as *.vbs.

Learn how to setup and run the macro.

### See Other SCPI Example Programs

```vbs
Dim app
Dim scpi
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' The AOPen data file should already exist
aopenFile = "S:\case_07\AOpen.s2p"
' Fixture data file will be created or overwritten
fixaFile = "S:\case_07\_demo_fix_a.s4p"

' AFR extracts fixture files
scpi.Execute("AFR:INITialize")
scpi.Execute("AFR:FIXTure:INPuts DIFF")
scpi.Execute("AFR:FIXTure:MEASurement 2")
scpi.Execute("AFR:STANdard:USE AOP,ON")
scpi.Execute("AFR:STANdard:LOAD AOP," & Q(aopenFile))
scpi.Execute("AFR:SAVE:PORTs VNA")
scpi.Execute("AFR:SAVE:FILename " & Q(fixaFile))
opc = scpi.Execute("*OPC?")

' deembedding
scpi.Execute("CALC:FSIM:EMB:TYPE C")
scpi.Execute("CALC:FSIM:EMB:TOP:C:PORT 1,2,3,4")
scpi.Execute("CALC:FSIM:EMBed:NETwork1:FILename " & Q(fixaFile))
scpi.Execute("CALC:FSIM:EMBed:NETwork1:PMAP 1,2,3,4")
scpi.Execute("CALC:FSIM:EMBed:NETwork1:TYPE DEEMbed")
scpi.Execute("CALC:FSIM:EMBed:STATE ON")
scpi.Execute("CALC:FSIM:STATe ON")
opc = scpi.Execute("*OPC?")

Set scpi = Nothing
Set app = Nothing

'Add double quotation marks to a string
Function Q(s)
    Q = Chr(34) & s & Chr(34)
End Function
```

5148
This VBScript program performs AFR using one single-ended 2X THRU, saves the fixture data to a file, then performs deembedding.

Each VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the code into a text editor file, such as Notepad, and save it on the PNA hard drive as *.vbs.

Learn how to setup and run the macro.

```
Dim app
Dim scpi
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

'The 2X THRU data file should already exist
thruFile = "S:\case_01\AA.s2p"
'Fixture A and B data file will be created or overwritten
fixaFile = "S:\case_01\demo_fix_a.s2p"
fixbFile = "S:\case_01\demo_fix_b.s2p"

'AFR extracts fixture files
scpi.Execute("AFR:INITialize")
scri.Execute("AFR:STANdard:USE THRU,1")
scri.Execute("AFR:STANdard:LOAD THRU," & Q(thruFile))
scri.Execute("AFR:SAVE:PORTs VNA")
scri.Execute("AFR:SAVE:FILename " & Q(fixaFile) & "," & Q(fixbFile))
opc = scri.Execute("*OPC?")

'Deembedding
scpi.Execute("CALC:FSIM:SEND:DEEM:STATe ON")
scpi.Execute("CALC:FSIM:STATe ON")
opc = scri.Execute("*OPC?")

Set scri = Nothing
Set app = Nothing

'Add double quotation marks to a string
Function Q(s)
    Q = Chr(34) & s & Chr(34)
End Function
```
This VBScript program performs AFR using one single-ended OPEN, saves the fixture data to a file, then performs deembedding.

Each VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the code into a text editor file, such as Notepad, and save it on the PNA hard drive as *.vbs.

Learn how to setup and run the macro.

---

See Other SCPI Example Programs

```vbs
Dim app
Dim scpi
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' The AOpen data file should already exist
aopenFile = "S:\case_05\AOpen.s1p"
' Fixture data file will be created or overwritten
fixaFile = "S:\case_05\_demo_fix_a.s2p"

' AFR extracts fixture files
scpi.Execute("AFR:INITialize")
scpi.Execute("AFR:FIXTure:MEASurement 1")
scpi.Execute("AFR:STANdard:USE AOPen")
scpi.Execute("AFR:STANdard:LOAD AOPen," & Q(aopenFile))
scpi.Execute("AFR:SAVE:PORTs VNA")
scpi.Execute("AFR:SAVE:FILename " & Q(fixaFile))
opc = scpi.Execute("*OPC?")

' Deembedding
scpi.Execute("CALC:FSIM:SEND:DEEM:STATE ON")
scpi.Execute("CALC:FSIM:STATE ON")
opc = scpi.Execute("*OPC?")

Set scpi = Nothing
Set app = Nothing

' Add double quotation marks to a string
Function Q(s)
    Q = Chr(34) & s & Chr(34)
End Function
```
Active (Hot) Parameters

This program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as <filename>.vbs. Learn how to setup and run the macro.

See Also

Active Parameters Commands

---

### See Other SCPI Example Programs

```vbnet
'Demonstration of Active Parameters setup.
Dim app
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

scpi.Parse "SYST:FPR"
scpi.Parse "CALC:CUST:DEF 'mytrace', 'Active Parameters', 'IPwr' "
scpi.Parse "DISP:WIND ON"
scpi.Parse "DISP:WIND:TRAC:FEED 'mytrace'"

scpi.Parse "SENS:ACT:SWE:TYPE MULT"   'Sweep type to multi sweep
scpi.Parse "SENS:ACT:SWE:POW:STOP 0"  'Stop power 0dBm
scpi.Parse "SENS:ACT:SWE:POW:STEP 4"  '4 steps with -15,-10,-5,0dBm

scpi.Parse "SENS:ACT:DISP:TRAC1:IPW -5"  'Set the display as the -5dBm input
```
This VBScript program polls for the completion of a sweep.

How to run this program:

This VBScript (*.vbs) program can be run as a macro in the analyzer. To do this, copy the following code into a text editor file such as Notepad and save it on the analyzer hard drive as NewMeas.vbs. Learn how to setup and run the macro.

See Other SCPI Example Programs

```vbs
sub SweepAndPoll

Write "SYST:PRES" ' Preset
Write "SENS:SWE:MODE HOLD" ' Put the channel into hold mode
Write "*ESE 1" ' Turn on the *ESR? bit
Write "*CLS" ' Clear any pending status
Write "SENS:SWE:TIME 2" ' Set the sweep time to 2 seconds
Write "SENS:SWE:MODE SING" ' Initiate a sweep, but don't wait for complete
Write "*OPC" ' Request notification on sweep complete
done = 0
count = 0
while (done <> 1)
    Write "*ESR?" ' Check if sweep is complete
done = Read()
count = count + 1
wscript.echo "Sweep not completed. Try#:" & count
wscript.sleep 100 ' Wait for 100 ms
wend
```
wscript.echo "Sweep Completed. Try#: " & count
end Sub

' Infrastructure to setup the Write/Read functions
dim LastReadBuffer
Sub Write(command)
    LastReadBuffer = s.Execute(command)
end sub

Function Read()
    Read = LastReadBuffer
end function

' Setup and Call SweepAndPoll
set app = CreateObject("Agilentpna835x.application")
set s = app.scpistringparser
SweepAndPoll
Perform a Cal All Channels Calibration

There are two sets of commands used to automate a Cal All Channels Calibration: `SYST:CAL:ALL <commands>` and `SENS<chan>:CORR:COLL:GUIDed <commands>`.

**SYST:CAL:ALL <commands>**

The general sequence for setting up the Cal All session is as follows:

1. Select the channels to calibrate using the `SYST:CAL:ALL:SEL` command.
2. Select the ports to calibrate using the `SYST:CAL:ALL:CHAN:PORTs` command.
3. Set the properties that are available in Cal All that are relevant to the channels you are calibrating using the `SYST:CAL:ALL:MCLass:PROP:VAL <name>,<val>` command. For example, setting `<name>` to "Include Power Calibration" and `<val>` to "true" will include a source and receiver power calibration in the Cal All calibration.
4. Query the channel number to use for the remaining cal commands. This channel is used for the sole purpose of acquiring cal data and finds the highest available channel number.

   **Note:** You must query this number – do not assume that it will always be a particular value. For example: `a.chan = SYST:CAL:ALL:GUIDed:CHAN?`

**SENS<chan>:CORR:COLL:GUIDed <commands>**

These commands are identical to the command used for a single channel calibration. However, the number used for the SENSe header is determined by the `SYST:CAL:ALL:GUIDed:CHAN?` command. The general sequence is as follows:

1. Set up the power sensor using the `SENS:CORR:COLL:GUID:PSENso` commands if you will be performing a power calibration (source and receiver power cal).
2. Set up the connector family and gender per port using the `SENS:CORR:COLL:GUID:CONN:PORT` command.
5. Query the number of steps using the `SENS:CORR:COLL:GUID:STEPs?` command.
7. Save the calset using the `SENS:CORR:COLL:GUID:SAVE` command.
Cal All Examples

1-Port, 1-Channel, no Power Cal, with ECal Module
2-Port, 1-Channel, no Power Cal, with ECal Module
2-Port, 1-Channel, with Power Cal, with ECal Module
2-Port, 2-Channel, with Power Cal, with ECal Module
Noise Figure Cal All
Noise Figure Cal All with USB noise source
SMC Cal All
IMD 2nd Order Cal All
Cal All for Mixer Channel
Independent Power Calibration
Multiple Instances of Calibrate All Channels
Cal All SMC Split Cal
Cal All Independent Calibration Channels
Cal All Multi-Channel Independent Calibration Channels

Each VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the code into a text editor file, such as Notepad, and save it on the VNA hard drive as *.vbs.

Learn how to setup and run the macro.
See CalAll SCPI commands
Learn about Cal All

See Other SCPI Example Programs
Catalog Measurements using SCPI

This Visual Basic Program does the following:

- Catalogs the currently defined measurements, windows, and traces
- Selects a measurement for further definition
- Adds a Title to the window

To run this program, you need:

- An established GPIB interface connection

See Other SCPI Example Programs

Dim Meas as String
Dim Win as String
Dim Trace as String

' Read the current measurements in Channel 1
GPIB.Write "CALCulate1:PARameter:CATalog?"
Meas = GPIB.Read
MsgBox("Ch1 Measurements: " & Meas)

' Read the current windows
GPIB.Write "DISPlay:CATalog?"
Win = GPIB.Read
MsgBox("Windows: " & Win)

' Read current traces in window 1
GPIB.Write "DISPlay:WINDow1:CATalog?"
Trace = GPIB.Read
MsgBox("Traces in Window1: " & Trace)
This VBScript program does the following:

- Presets the analyzer, deleting the default trace
- Create 2 windows
- Create 2 Measurements
- Feed the measurements to windows / traces
- Change frequency ranges for channels
- Select both measurements
- Turn marker 1 ON for each measurement

The following notes explain the basic structure of the SCPI tree on the analyzer:

- **SOURce**: and most **SENSe**: commands act on the **channel** that is specified in the command. Channel 1 is default if not specified.

- Most **DISPlay**: commands act on the **window and trace** specified in the command. Window1 and Trace1 are default if not specified.

- **CALCulate**: commands act on the **selected measurement** in the specified channel. Select the measurement for each channel using **CALCulate<channel number>:PARameter:SELect <meas name>**. You can select one measurement in each channel.

See [Traces, Channels, and Windows on the Analyzer](#)

**How to run this program:**

The SCPI commands in this example are sent over a COM interface using the **SCPIStringParser** object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the analyzer. To do this, copy the following code into a text editor file such as Notepad and save it on the analyzer hard drive as NewMeas.vbs. **Learn how to setup and run the macro.**

See [Other SCPI Example Programs](#)
Dim app
Dim scpi

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' Preset the analyzer
' This command also deletes the default trace
scpi.execute "SYSTem:FPReset"

' Create Measurements
scpi.execute "CALCulate1:PARameter:DEFine:EXT 'Meas1','S11'"
scpi.execute "CALCulate2:PARameter:DEFine:EXT 'Meas2','S21'"

' Turn on windows - creates if new
scpi.execute "DISPlay:WINDow1:STATE ON"
scpi.execute "DISPlay:WINDow2:STATE ON"

' Associate ("FEED") the measurement name('Meas1') to WINDow(1), and give the new TRACe a number(1).
scpi.execute "DISPlay:WINDow1:TRACe1:FEED 'Meas1'"
scpi.execute "DISPlay:WINDow2:TRACe2:FEED 'Meas2'"

' Change each channel's frequency range
scpi.execute "SENSe1:FREQuency:SPAN 1e9"
scpi.execute "SENSe2:FREQuency:SPAN 2e9"

' Select both measurements
scpi.execute "CALCulate1:PARameter:SELect 'Meas1'"
scpi.execute "CALCulate2:PARameter:SELect 'Meas2'"

' Turn marker 1 ON for each measurement
scpi.execute "CALCulate1:MARKer:STATE ON"
scpi.execute "CALCulate2:MARKer:STATE ON"
GPIB.Write "SYSTem:PREset"

'Create two windows
GPIB.Write "::DISPlay:SPLit 2"

'Create one trace on each window
GPIB.Write "::CALCulate1:PARameter:COUNT 1"
GPIB.Write "::CALCulate2:PARameter:COUNT 1"

'Define the parameter for each trace
GPIB.Write "::CALCulate1:MEASure1:PARameter 'S21'"
GPIB.Write "::CALCulate2:MEASure2:PARamerter 'S12'"

'Change each channel's frequency range
GPIB.Write "::SENSe1:FREQuency:SPAN 1e9"
GPIB.Write "::SENSe2:FREQuency:SPAN 2e9"

'Turn marker 1 ON for each measurement
GPIB.Write "::CALCulate1:MEASure1:MARKer:STATe ON"
GPIB.Write "::CALCulate2:MEASure2:MARKer:STATe ON"
VNA as Controller and Talker / Listener

This Visual Basic Program uses VISA to do the following:

- Control the VNA using a VISA LAN Client interface on the VNA.
- Control another instrument using the VNA as GPIB controller.
- Queries both the analyzer and other instrument to identify themselves with *IDN?

**Note:** This program can be modified to work from a remote PC to control both instruments. In that case, set up the VNA to be a talker/listener.

To run this program, you need to do the following:

- Add module **visa32.bas** to the VB project.
- Configure the VNA for VISA / SICL
- Set up the VNA to be GPIB system controller.
- Connect another instrument to the analyzer through a GPIB cable with Primary address of 13 on GPIB0 interface

**See Other SCPI Example Programs**

Sub main()

' This application run from onboard the VNA 
' can control both the VNA and another GPIB instrument.
'
' To run this program the module visa32.bas must be added
' to the project.

'VISA function status return code
Dim status As Long
'Session to Default Resource Manager
Dim defRM As Long
'Session to instrument
Dim viPNA As Long
'Session to other GPIB instrument
Dim viInstrument As Long
' String to hold results
Dim strRes As String * 200
On Error GoTo ErrorHandler

status = viOpenDefaultRM(defRM)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Open the session to the VNA
status = viOpen(defRM, "GPIB1::16::INSTR", 0, 0, viPNA)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Ask for the VNA's ID.
status = viVPrintf(viPNA, "*IDN?" + Chr$(10), 0)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Read the ID as a string.
status = viVScanf(viPNA, "%t", strRes)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler
'Display the results
MsgBox "PNA is: " + strRes

'Open the session to the other instrument
status = viOpen(defRM, "GPIB0::13::INSTR", 0, 0, viInstrument)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Ask for the instrument's ID.
status = viVPrintf(viInstrument, "*IDN?" + Chr$(10), 0)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Read the ID as a string.
status = viVScanf(viPNA, "%t", strRes)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler
'Display the results
MsgBox "Other instrument is: " + strRes
' Close the resource manager session (which closes everything)
Call viClose(defRM)
End

ErrorHandler:
'Display the error message
MsgBox "*** Error : " + Error$, MB_ICONEXCLAMATION
End

VisaErrorHandler:
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr
End
End Sub
This example program does the following:

- creates several Balanced measurements in separate windows
- generates markers
- calculates statistics
- sets limit lines and queries results
- queries a measurement to determine if we have a balanced parameter and what type it is.

**Note:** By their nature, balanced measurements are extremely sensitive to phase differences between the two RF paths that make up the balanced port, especially at higher frequencies. A good calibration (not performed in this example) is critical to achieving good balanced measurement results.

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Balanced.vbs. Learn how to setup and run the macro.

```vbs
Dim app
Dim scpi

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' A comment
scpi.Parse("SYST:FPRESET")

' This example uses DUT topology Bal-Bal -
' a DUT with a balanced input and balanced output.
'
' Port mapping for our DUT:
' logical port 1 = physical ports 1 and 4
```
' logical port 2 = physical ports 2 and 3
' The default is:
' logical port 1 = physical ports 1 and 2
' logical port 2 = physical ports 3 and 4
'
' logical 1               logical 2
' __________
' 1 ------|         |------ 2 +
'     |   DUT   |
' 4 ------|___________|------ 3 -
'
' Turn on Four windows
scpi.Parse("DISP:WIND1:STATe ON")
scpi.Parse("DISP:WIND2:STATe ON")
scpi.Parse("DISP:WIND3:STATe ON")
scpi.Parse("DISP:WIND4:STATe ON")

' Create a trace called "sdd21", and for that trace turn on the balanced
' transformation and set the balanced transformation to BBAL SDD21.
scpi.Parse("CALC:PAR:DEF:EXT ""sdd21"",S11")
scpi.Parse("CALC:PAR:SEL ""sdd21"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")

' Feed the sdd21 trace to window 1, trace 1
scpi.Parse("DISP:WIND1:TRAC1:FEED ""sdd21"")

' Similarly create 3 more balanced transmission/conversion parameters
' Create Scd21
scpi.Parse("CALC:PAR:DEF:EXT ""scd21"",S11")
scpi.Parse("CALC:PAR:SEL ""scd21"")
scpi.Parse("CALC:FSIM:PAR:STATE ON")
scpi.Parse("CALC:FSIM:PAR:BBAL:DEF SCD21")
scpi.Parse("DISP:WIND1:TRAC2:FEED ":"scd21"")

' Create Sdc21
scpi.Parse("CALC:PAR:SEL ":"sdc21"")
scpi.Parse("CALC:FSIM:PAR:STATE ON")
scpi.Parse("CALC:FSIM:PAR:BBAL:DEF SDC21")
scpi.Parse("DISP:WIND1:TRAC3:FEED ":"sdc21"")

' Create Scc21
scpi.Parse("CALC:PAR:SEL ":"scc21"")
scpi.Parse("CALC:FSIM:PAR:STATE ON")
scpi.Parse("DISP:WIND1:TRAC4:FEED ":"scc21"")

' Now create logical port 1 reflection parameters, and place them in window 2
scpi.Parse("CALC:PAR:SEL ":"sdd11"")
scpi.Parse("CALC:FSIM:PAR:STATE ON")
scpi.Parse("CALC:FSIM:PAR:BBAL:DEF SDD11")
scpi.Parse("DISP:WIND2:TRAC1:FEED ":"sdd11"")

' Feed the sdd11 trace to window 2, trace 1
scpi.Parse("DISP:WIND2:TRAC1:FEED ":"sdd11"")

' Similarly create 3 more balanced reflection/conversion parameters
scpi.Parse("CALC:PAR:SEL ":"scd11"")
scpi.Parse("CALC:FSIM:PAR:STATE ON")
scpi.Parse("CALC:FSIM:PAR:BBAL:DEF SCD11")
scpi.Parse("DISP:WIND2:TRAC2:FEED ":"scd11"")
scpi.Parse("CALC:PAR:DEF:EXT ""sdc11"",S11")
scpi.Parse("CALC:PAR:SEL ""sdc11""")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("DISP:WIND2:TRAC3:FEED ""sdc11"""")
scpi.Parse("CALC:PAR:DEF:EXT ""scc11"",S11")
scpi.Parse("CALC:PAR:SEL ""scc11""")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("DISP:WIND2:TRAC4:FEED ""scc11"""")

' Now create reverse transmission parameters, and place them in window 3
scpi.Parse("CALC:PAR:DEF:EXT ""sdd12"",S11")
scpi.Parse("CALC:PAR:SEL ""sdd12""")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")

' Feed the sdd11 trace to window 3, trace 1
scpi.Parse("DISP:WIND3:TRAC1:FEED ""sdd12"""")

' Similarly create 3 more balanced reverse transmission/conversion parameters
scpi.Parse("CALC:PAR:DEF:EXT ""scd12"",S11")
scpi.Parse("CALC:PAR:SEL ""scd12""")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SCD12")
scpi.Parse("DISP:WIND3:TRAC2:FEED ""scd12"""")
scpi.Parse("CALC:PAR:DEF:EXT ""sdc12"",S11")
scpi.Parse("CALC:PAR:SEL ""sdc12""")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse(\"DISP:WIND3:TRAC3:FEED ""sdc12""\")
scpi.Parse(\"CALC:PAR:DEF:EXT ""scc12"",S11\")
scpi.Parse(\"CALC:PAR:SEL ""scc12""\")
scpi.Parse(\"CALC:FSIM:BAL:PAR:STATe ON\")
scpi.Parse(\"DISP:WIND3:TRAC4:FEED ""scc12""\")

' Now create reverse reflection parameters, and place them in window 4
scpi.Parse(\"CALC:PAR:SEL ""sdd22""\")
scpi.Parse(\"CALC:FSIM:BAL:PAR:STATe ON\")

' Feed the sdd11 trace to window 3, trace 1
scpi.Parse(\"DISP:WIND4:TRAC1:FEED ""sdd22""\")

' Similarly create 3 more balanced reverse reflection parameters
scpi.Parse(\"CALC:PAR:DEF:EXT ""scd22"",S11\")
scpi.Parse(\"CALC:PAR:SEL ""scd22""\")
scpi.Parse(\"CALC:FSIM:BAL:PAR:STATe ON\")
scpi.Parse(\"DISP:WIND4:TRAC2:FEED ""scd22""\")
scpi.Parse(\"CALC:PAR:DEF:EXT ""sdc22"",S11\")
scpi.Parse(\"CALC:PAR:SEL ""sdc22""\")
scpi.Parse(\"CALC:FSIM:BAL:PAR:STATe ON\")
scpi.Parse(\"DISP:WIND4:TRAC3:FEED ""sdc22""\")
scpi.Parse(\"CALC:PAR:DEF:EXT ""scc22"",S11\")
scpi.Parse(\"CALC:PAR:SEL ""scc22""\")
scpi.Parse(\"CALC:FSIM:BAL:PAR:STATe ON\")
scpi.Parse("DISP:WIND4:TRAC4:FEED ""scc22"")
scpi.Parse("CALC:FSIM:BAL:DEVice BBALanced")
scpi.Parse("CALC:FSIM:BAL:TOPology:BBAL:PPORts 1,4,2,3")
'
Set up stimulus
scpi.Parse("SENS:SWE:POINts 801")
scpi.Parse("SENS:FREQ:STARt 10e6")
scpi.Parse("SENS:FREQ:STOP 1e9")
'
Here we demonstrate how to determine if we have
a balanced parameter and what type it is.
'
Read back one parameter to verify its type
scpi.Parse("CALC:PAR:SEL ""sdd21"")
'
Is this a balanced parameter?
isbal = scpi.Parse("CALC:FSIM:BAL:PAR?")
'
Which topology/device is set?
device = scpi.Parse("CALC:FSIM:BAL:DEV?")
device = Left( device, Len(device)-1 ) ' strip off newline
'
Which parameter are we measuring within that topology?
balparam = scpi.Parse("CALC:FSIM:BAL:PAR:" & device & ":DEF?")
balparam = Left( balparam, Len(balparam)-1 ) ' strip off newline
If isbal Then
WScript.Echo "Balanced Parameter: " & balparam & " in topology: " & device & "."
Else
WScript.Echo "Parameter not balanced."
End If
Create a Measurement using SCPI

This VBScript program creates a new S21 measurement and displays it on the VNA screen.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as NewMeas.vbs. Learn how to setup and run the macro.

See Other SCPI Example Programs

```vbs
Dim app
Dim scpi
    ' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
    ' A comment
    'Preset the analyzer
scpi.Execute ("SYST:PREset")
    ' Create and turn on window 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")
    'Define a measurement name, parameter
scpi.Execute ("CALCulate:PARameter:DEFine:EXT 'MyMeas',S21")
    'Associate ("FEED") the measurement name ("MyMeas") to WINDow (1), and give the new
    TRACe a number (1).
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")
```

Example

```vbs
Dim app
Dim scpi
    ' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
    ' A comment
    'Preset the analyzer
scpi.Execute ("SYST:PREset")
    'Define a measurement parameter
scpi.Execute ("CALCulate:MEASure1:PARameter 'S21'")
```
Create a Multi-Dimensional Sweep

This program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as <filename>.vbs. Learn how to setup and run the macro.

See Also

Multi-Dimensional Sweep Commands

See Other SCPI Example Programs

'Demonstration of multi-dimensional sweep setup.
set pna = CreateObject("AgilentPNA835x.Application","A-N5242A-10096")
set scpi = pna.ScpiStringParser
CreateSAMeasurement
SetupMultiSweep

Sub CreateSAMeasurement
    scpi.Parse "SYST:FPR"
    scpi.Parse "DISP:WIND ON"
    scpi.Parse "CALC:CUST:DEF 'sa_meas', 'Spectrum Analyzer', 'R1'"
    scpi.Parse "DISP:WIND:TRAC:FEED 'sa_meas'"
    scpi.Parse "CALC:PAR:SEL 'sa_meas'"
    scpi.Parse "SENS:SWE:MODE HOLD"
End Sub

Sub SetupMultiSweep
    'Turn Port 1 ON
    scpi.Parse "SOUR:POW1:MODE ON"
    'Configure Port 1 frequency start/stop range
    scpi.Parse "SOUR:FREQ1:START 2E9"
    scpi.Parse "SOUR:FREQ1:STOP 4E9"
    'Set Port 1 frequency domain's order to 3
    scpi.Parse "SOUR:FREQ1:DIM:ORDER 3"
    'Enable Port 1 frequency domain in multi-dimensional sweep
    scpi.Parse "SOUR:FREQ1:DIM:STAT ON"
    'Configure Port 1 power start/stop range
    scpi.Parse "SOUR:POW1:START -5"
    scpi.Parse "SOUR:POW1:STOP 5"

'Set Port 1 power domain's order to 5
scpi.Parse "SOUR:POW1:DIM:ORDER 5"
'Enable Port 1 power domain in multi-dimensional sweep
scpi.Parse "SOUR:POW1:DIM:STAT ON"
'Configure Port 1 phase start/stop range
scpi.Parse "SOUR:PHAS1:START 0"
scpi.Parse "SOUR:PHAS1:STOP 270"
'Set Port 1 phase domain's order to 4
scpi.Parse "SOUR:PHAS1:DIM:ORDER 4"
'Enable Port 1 phase domain in multi-dimensional sweep
scpi.Parse "SOUR:PHAS1:DIM:STAT ON"

'Turn Port 3 ON
scpi.Parse "SOUR:POW3:MODE ON"
'Configure Port 3 frequency start/stop range
scpi.Parse "SOUR:FREQ3:START 2E9"
scpi.Parse "SOUR:FREQ3:STOP 4E9"
'Set Port 3 frequency domain's order to 3
scpi.Parse "SOUR:FREQ3:DIM:ORDER 3"
'Enable Port 3 frequency domain in multi-dimensional sweep
scpi.Parse "SOUR:FREQ3:DIM:STAT ON"
'Configure Port 3 power start/stop range
scpi.Parse "SOUR:POW3:START -5"
scpi.Parse "SOUR:POW3:STOP 5"
'Set Port 3 power domain's order to 5
scpi.Parse "SOUR:POW3:DIM:ORDER 5"
'Enable Port 3 power domain in multi-dimensional sweep
scpi.Parse "SOUR:POW3:DIM:STAT ON"
'Configure Port 3 phase start/stop range
scpi.Parse "SOUR:PHAS3:START 0"
scpi.Parse "SOUR:PHAS3:STOP 270"
'Set Port 3 phase domain's order to 4
scpi.Parse "SOUR:PHAS3:DIM:ORDER 4"
'Enable Port 3 phase domain in multi-dimensional sweep
scpi.Parse "SOUR:PHAS3:DIM:STAT ON"

'Turn AO1 ON
scpi.Parse "SOUR:DC:STAT 'AO1', ON"
'Configure AO1 start/stop range
scpi.Parse "SOUR:DC:START 'AO1', 0"
scpi.Parse "SOUR:DC:STOP 'AO1', 2"
'Set AO1's order to 2
scpi.Parse "SOUR:DC:DIM:ORDER 'AO1', 2"
'Enable AO1 in multi-dimensional sweep
scpi.Parse "SOUR:DC:DIM:STAT 'AO1', ON"

'Set dimension order 3's point count to 3
scpi.Parse "SOUR:DIM3:POIN 3"
'Set dimension order 3's repeat count to 1
scpi.Parse "SOUR:DIM3:REP:COUN 1"
'Set dimension order 4's point count to 4
scpi.Parse "SOUR:DIM4:POIN 4"
'Set dimension order 4's repeat count to 1
scpi.Parse "SOUR:DIM4:REP:COUN 1"
'Set dimension order 5's point count to 5
scpi.Parse "SOUR:DIM5:POIN 5"
'Set dimension order 5's repeat count to 1
scpi.Parse "SOUR:DIM5:REP:COUN 1"

End Sub
Create a Narrowband Point-in-Pulse Measurement using the PNA-X

The following SCPI example demonstrates how to create a Narrowband Point-in-Pulse measurement using the Pulsed Application DLL on the PNA-X.

It first gets valid configuration settings and then uses those settings to configure the VNA and internal pulsed generators.

To run this program, you need:

- PNA-X
- Pulsed Application (Option H08)

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as PulseProfile.vbs. Learn how to setup and run the macro.

Note: Because of the long length of some commands in this example, word wrapping may occur when copying. These lines require modification after pasting.

See Also

- Install and register the pulsed .dll on your PC.
- ConfigEnhancedNB2 method for sending and returning parameters to the .dll.
- ConfigEnhancedNBIFAtten method for setting the receiver IF gain.
- SCPI IF Configuration commands used in the program.
- Other Pulse SCPI examples

```
//This example shows you how to perform point in pulse measurement based on
//PNA-X in narrowband mode using SCPI commands.

public partial class Form1 : Form
{

    private object pna;
    private object scpi;
```
private Type srvtype;

private AgilentPNA835x.applicationClass pulseApp;

public Form1()
{
InitializeComponent();
}

private string sendScpiCommand(string scpitext)
{
object[] parameter = new object[1];
parameter[0] = scpitext;
return (string)srvtype.InvokeMember("parse", BindingFlags.InvokeMethod, null, scpi, parameter);
}

private void ConnectToPNA()
{
srvtype = Type.GetTypeFromProgID("AgilentPNA835x.Application", true);
pna = Activator.CreateInstance(srvtype);
scpi = srvtype.InvokeMember("ScpiStringParser", BindingFlags.GetProperty, null, pna, null);
}

private void NBButton_Click(object sender, EventArgs e)
{
double dPRF = 10000, dBW = 500; //PRF=10kHz
double dPhysicalIF = 0, dNCO = 0, dClockFreq = 0;
System.Array aStage1TapArray = null, aStage2TapArray = null, aStage3TapArray = null;
bool bFixedPRF = true;
double dGateDelay = 0.000002, dGateWidth = 0.000005; //Gate width=50ns
double dSWGateDelay = 0, dSWGateWidth = 0;
int iSWGateRamp = 0;
double dModPulseWidth = 0.00001;//10 us
double dModPulseDelay = 0;//0us
short myAtten = 0;
pulseApp = new AgilentPNAPulsed.applicationClass();
ConnectToPNA();
//Preset PNA-X
sendScpiCommand("*RST");
//Measure S21
sendScpiCommand("DISP:WIND:TRAC1:DEL");
sendScpiCommand("CALCulate:PARameter:DEFine:EXT /'MyMeas/' ,S21");
sendScpiCommand("DISP:WIND:TRAC1:FEED /'MyMeas/' ");
//Set power leveling mode to Openloop
sendScpiCommand("sour:pow1:alc:mode open");
//Send desired pulsed parameters to the pulsed configuration DLL.
//The DLL will return a new set of pulse parameters to send to the PNA-X.
pulseApp.ConfigEnhancedNB2(ref dPRF, ref dBW, ref dPhysicalIF, ref dNCO, ref dClockFreq, ref aStage1TapArray,
ref aStage2TapArray, ref aStage3TapArray, bFixedPRF, dGateDelay, dGateWidth,
ref dSWGateDelay, ref dSWGateWidth, ref iSWGateRamp);
double pulsePeriod = 1 / dPRF;
//Pulse #1 as modulation source
sendScpiCommand("sens:puls:per " + pulsePeriod.ToString()); // 100us
//Set Pulse1 width
sendScpiCommand("sens:puls1:width " + dModPulseWidth.ToString()); //10us
//Set Pulse1 delay
sendScpiCommand("sens:puls1:delay " + dModPulseDelay.ToString()); //10us
//Turn on Pulse1
sendScpiCommand("SENS:PULS1:STAT 1");

//Set modulation source to Pulse1
sendScpiCommand("sens:path:conf:elem /"PulseModDrive/","Pulse1/");

//Enable pulse modulator 1
sendScpiCommand("sens:path:conf:elem /"Src1Out1PulseModEnable/","Enable/");

//Pulse #2 controls receiver gate
sendScpiCommand("sens:puls2:width " + dGateWidth.ToString()); //50ns
sendScpiCommand("sens:puls2:delay " + dGateDelay.ToString()); //0
sendScpiCommand("SENS:PULS2:STAT 1");
sendScpiCommand("sens:path:conf:elem /"IFGateA/","Pulse2/");
sendScpiCommand("sens:path:conf:elem /"IFGateB/","Pulse2/");
sendScpiCommand("sens:path:conf:elem /"IFGateR1/","Pulse2/");
sendScpiCommand("sens:path:conf:elem /"IFGateR2/","Pulse2/");

//Set IFBW
sendScpiCommand("SENS:BWID " + dBW.ToString());

//Configure IF path
sendScpiCommand("sens:path:conf:elem /"IFSigPathAll/","NBF/");
sendScpiCommand("SENS:IF:FILT:AUTO 0");
sendScpiCommand("SENS:IF:FREQ:AUTO 0");
sendScpiCommand("SENS:IF:FREQ " + dPhysicalIF.ToString());

// Set filter stages based on pulse parameters
sendScpiCommand("SENS:IF:FILT:STAGE1:FREQ " + dNCO.ToString());

//Convert StagelTapArray to string
string buf1 = new string(' ', 1000);
for (int i = 0; i < aStage1TapArray.GetLength(0); i++)
{
    buf1 = buf1 + aStage1TapArray.GetValue(i).ToString() + ",";
buf1 = buf1.Trim();
buf1 = buf1.Substring(0, buf1.Length - 1);

// Convert Stage1TapArray to string
string buf2 = new string(' ', 1000);
for (int j = 0; j < aStage2TapArray.GetLength(0); j++)
{
    buf2 = buf2 + aStage2TapArray.GetValue(j).ToString() + ",";
}
buf2 = buf2.Trim();
buf2 = buf2.Substring(0, buf2.Length - 1);

// Set IF Filter Stage1 and Stage2 Coeficent
sendScpiCommand("SENS:IF:FILT:STAG1:COEF " + buf1);
sendScpiCommand("SENS:IF:FILT:STAG2:COEF " + buf2);
if (dSWGateWidth == 0) // No valid SW gate
{
    sendScpiCommand("SENS:IF:FILT:STAG3:TYPE 'RECT'");
    sendScpiCommand("SENS:IF:FILT:STAG3:PAR 'C'," + aStage3TapArray.GetValue(0).ToString());
    sendScpiCommand("SENS:PULS0:STAT 0");
}
else
{
    sendScpiCommand("SENS:IF:FILT:STAG3:TYPE 'PWIN'");
    sendScpiCommand("SENS:IF:FILT:STAG3:PAR 'C'," + aStage3TapArray.GetValue(0).ToString());
    sendScpiCommand("SENS:IF:FILT:STAG3:PAR 'D'," + dSWGateDelay.ToString());
    sendScpiCommand("SENS:IF:FILT:STAG3:PAR 'W'," + dSWGateWidth.ToString());
sendScpiCommand("SENS:IF:FILT:STAGE3:PAR 'R'," + iSWGateRamp.ToString());

double pulse0Width = 1 / dClockFreq;

sendScpiCommand("sens:puls0:width " + pulse0Width.ToString());

sendScpiCommand("sens:puls0:delay 0");

sendScpiCommand("SENS:PULS0:STAT 1");

}

pulseApp.ConfigEnhancedNBIFAtten(dPRF, dGateWidth, ref myAtten);

sendScpiCommand("sens:path:conf:elem "/"NBFATNA/""," + myAtten.ToString() + "/");

sendScpiCommand("sens:path:conf:elem "/"NBFATNB/""," + myAtten.ToString() + "/");

sendScpiCommand("sens:path:conf:elem "/"NBFATNR1/""," + myAtten.ToString() + "/");

sendScpiCommand("sens:path:conf:elem "/"NBFATNR2/""," + myAtten.ToString() + "/");

// Set start and stop frequency

sendScpiCommand("SENS:FREQ:STAR 1000000000");

sendScpiCommand("SENS:FREQ:STOP 2000000000");

// Single Sweep

sendScpiCommand("SENS:SWE:MODE SING");

sendScpiCommand("DISP:WIND:TRAC:Y:AUTO");

}
Create a Narrowband Pulse Profile Measurement using the PNA-X

The following SCPI example demonstrates how to create a Narrowband Pulse Profile measurement using the Pulsed Application DLL on the PNA-X.

It first gets valid configuration settings and then uses those settings to configure the VNA and internal pulsed generators.

To run this program, you need:

- PNA-X
- Pulsed Application (Option H08)

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as PulseProfile.vbs. Learn how to setup and run the macro.

**Note:** Because of the long length of some commands in this example, word wrapping may occur when copying. These lines require modification after pasting.

See Also

- Install and register the pulsed .dll on your PC.
- ConfigEnhancedNB2 method for sending and returning parameters to the .dll.
- ConfigEnhancedNBIFAtten method for setting the receiver IF gain.
- SCPI IF Configuration commands used in the program.
- Other Pulse SCPI examples

```csharp
public partial class Form1 : Form
{
    private object pna;
    private object scpi;
    private Type srvtype;
    private AgilentPNAPulsed.applicationClass pulseApp;
```
public Form1()
{
    InitializeComponent();
}

private string sendScpiCommand(string scpitext)
{
    object[] parameter = new object[1];
    parameter[0] = scpitext;
    return (string)srvtpe.InvokeMember("parse", BindingFlags.InvokeMethod, null, scpi, parameter);
}

private void ConnectToPNA()
{
    srvtpe = Type.GetTypeFromProgID("AgilentPNA835x.Application", true);
    pna = Activator.CreateInstance(srvtpe);
    scpi = srvtpe.InvokeMember("ScpiStringParser", BindingFlags.GetProperty, null, pna, null);
}

private void NBGPIBbutton_Click(object sender, EventArgs e)
{
    double dPRF = 10000, dBW = 500; //PRF=10kHz
    double dPhysicalIF = 0, dNCO = 0, dClockFreq = 0;
    System.Array aStage1TapArray = null, aStage2TapArray = null, aStage3TapArray = null;
    bool bFixedPRF = true;
    double dGateDelay = 0.0, dGateWidth = 0.00000005; //Gate width=50ns
    double dSWGateDelay = 0, dSWGateWidth = 0;
    int iSWGateRamp = 0;
    double dModPulseWidth = 0.00001; //10 us
double dModPulseDelay = 0.00001;//10us
short myAtten = 0;
pulseApp = new AgilentPNAPulsed.applicationClass();
ConnectToPNA();
//Preset PNA-X
sendScpiCommand("*RST");
//Measure S21
sendScpiCommand("DISP:WIND:TRAC1:DEL");
sendScpiCommand("CALCulate:PARameter:DEFine:EXT /'MyMeas/', S21");
sendScpiCommand("DISP:WIND:TRAC1:FEED /'MyMeas/'");
//Set power leveling mode to Openloop
sendScpiCommand("sour:pow1:alc:mode open");
//Send desired pulsed parameters to the pulsed configuration DLL.
//The DLL will return a new set of pulse parameters to send to the PNA-X.
pulseApp.ConfigEnhancedNB2(ref dPRF, ref dBW, ref dPhysicalIF, ref dNCO, ref dClockFreq, ref aStage1TapArray,
ref aStage2TapArray, ref aStage3TapArray, bFixedPRF, dGateDelay, dGateWidth,
ref dSWGateDelay, ref dSWGateWidth, ref iSWGateRamp);

double pulsePeriod = 1 / dPRF;
//Pulse #1 as modulation source
sendScpiCommand("sens:puls:per " + pulsePeriod.ToString()); // 100us
//Set Pulse1 width
sendScpiCommand("sens:puls1:width "+dModPulseWidth.ToString()); //10us
//Set Pulse1 delay
sendScpiCommand("sens:puls1:delay " + dModPulseDelay.ToString()); //10us
//Turn on Pulse1
sendScpiCommand("SENS:PULS1:STAT 1");
//Set modulation source to Pulse1
sendScpiCommand("sens:path:conf:elem "/PulseModDrive","/Pulse1/"");

//Enable pulse modulator 1
sendScpiCommand("sens:path:conf:elem /Src1Out1PulseModEnable","/Enable/"");

//Pulse #2 controls receiver gate
sendScpiCommand("sens:puls2:width "+ dGateWidth.ToString(); //50ns
sendScpiCommand("sens:puls2:delay "+ dGateDelay.ToString(); //0
sendScpiCommand("SENS:PULS2:STAT 1");

sendScpiCommand("sens:path:conf:elem /IFGateA","/Pulse2/"");
sendScpiCommand("sens:path:conf:elem /IFGateB","/Pulse2/"");
sendScpiCommand("sens:path:conf:elem /IFGateR1","/Pulse2/"");
sendScpiCommand("sens:path:conf:elem /IFGateR2","/Pulse2/"");

//Set IFBW
sendScpiCommand("SENS:BWID "+dBW.ToString();

//Configure IF path
sendScpiCommand("sens:path:conf:elem /IFSigPathAll","/NBF/"");
sendScpiCommand("SENS:IF:FILT:AUTO 0");
sendScpiCommand("SENS:IF:FREQ:AUTO 0");
sendScpiCommand("SENS:IF:FREQ "+dPhysicalIF.ToString();

// Set filter stages based on pulse parameters
sendScpiCommand("SENS:IF:FILT:STAGe1:FREQ " + dNCO.ToString();

//Convert StagelTapArray to string
string buf1 = new string(' ', 1000);
for (int i = 0; i < aStage1TapArray.GetLength(0); i++)
{
    buf1 = buf1 + aStage1TapArray.GetValue(i).ToString()+",";
}
buf1 = buf1.Trim();
buf1 = buf1.Substring(0, buf1.Length - 1);

//Convert Stage1TapArray to string
string buf2 = new string(' ', 1000);
for (int j = 0; j < aStage2TapArray.GetLength(0); j++)
{
    buf2 = buf2 + aStage2TapArray.GetValue(j).ToString() + ",";
}
buf2 = buf2.Trim();
buf2 = buf2.Substring(0, buf2.Length - 1);

//Set IF Filter Stage1 and Stage2 Coeficient
sendScpiCommand("SENS:IF:FILT:STAG1:COEF " + buf1);
sendScpiCommand("SENS:IF:FILT:STAG2:COEF " + buf2);
if (dSWGateWidth == 0) // No valid SW gate
{
    sendScpiCommand("SENS:IF:FILT:STAG3:TYPE " + aStage3TapArray.GetValue(0).ToString());
    sendScpiCommand("SENS:PULS0:STAT 0");
}
else
{
    sendScpiCommand("SENS:IF:FILT:STAG3:TYPE " + aStage3TapArray.GetValue(0).ToString());
    sendScpiCommand("SENS:IF:FILT:STAG3:PAR " + dSWGateDelay.ToString());
    sendScpiCommand("SENS:IF:FILT:STAG3:PAR " + iSWGateRamp.ToString());
    double pulse0Width=1/dClockFreq;
sendScpiCommand("sens:puls0:width " + pulse0Width.ToString());
sendScpiCommand("sens:puls0:delay 0");
sendScpiCommand("SENS:PULS0:STAT 1");
}
pulseApp.ConfigEnhancedNBIFAtten(dPRF, dGateWidth, ref myAtten);
sendScpiCommand("sens:path:conf:elem /"NBFATNA/",/"+myAtten.ToString()+ "/"+");
sendScpiCommand("sens:path:conf:elem /"NBFATNB/",/" + myAtten.ToString() + "/"+");
sendScpiCommand("sens:path:conf:elem /"NBFATNR1/",/" + myAtten.ToString() + "/"+");
sendScpiCommand("sens:path:conf:elem /"NBFATNR2/",/" + myAtten.ToString() + "/"+");
//Run pulse profile using below several lines
//Set CW Mode
sendScpiCommand("SENS:SWE:TYPE CW");
double startTime, stopTime, stepTime;
int myProfilePoints;
startTime = 0.00001; //10us
stopTime = 0.00005; //50us
stepTime = 0.00000005; //50ns
myProfilePoints = (int)((stopTime - startTime) / stepTime) + 1;
sendScpiCommand("SENS:SWE:POIN " + myProfilePoints.ToString());
// Test gates
sendScpiCommand("sens:puls1:delay " + startTime.ToString());
sendScpiCommand("SENS:PULS2:DINC " + stepTime.ToString());
//Single Sweep
sendScpiCommand("SENS:SWE:MODE SING");
sendScpiCommand("DISP:WIND:TRAC:Y:AUTO");
Configure a PMAR Device

This VB Script program configures a new Power Meter as Receiver device and creates a trace using the PMAR.

Learn more about Power Meter as a Receiver

These programs can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as PMAR.vbs. Learn how to setup and run the macro.

See all External Device Configuration commands

See Other SCPI Example Programs

```vbnet
' This section gets the VNA application
' starts the scpi parser, and presets the VNA
dim app
Set app = CreateObject("AgilentPNA835x.Application")
set scpi = app.ScpiStringParser
scpi.parse "*rst"
scpi.parse "Syst:conf:edev:add 'newpmar1'"
scpi.parse "Syst:conf:edev:dtype 'newpmar1', 'Power Meter'
scpi.parse "Syst:conf:edev:driver 'pmar', 'AGPM'
scpi.parse "Syst:conf:edev:ioconfig 'newpmar1', 'gpib0::14::instr'
scpi.parse "Syst:conf:edev:pmar:sens 'newpmar1', 1"
scpi.parse "Syst:conf:edev:pmar:read:ntolerance 'newpmar1', 0.1"
scpi.parse "Syst:conf:edev:pmar:sens 'newpmar1', 1"
scpi.parse "Syst:conf:edev:pmar:fmax 'newpmar1', 10000000000"
scpi.parse "Syst:conf:edev:pmar:flim 'newpmar1', 0"
scpi.parse "Syst:conf:edev:pmar:tabl:rfac 'newpmar1', 100"
```
scpi.parse "Syst:conf:e dev:pmar:tabl:cfac:freq 'newpmar1', 1e9, 2e9, 3e9"
scpi.parse "Syst:conf:e dev:pmar:tabl:loss:freq 'newpmar1', 1e9, 2e9, 3e9"

'Activate and enable the PMAR external device
scpi.parse "Syst:conf:e dev:ioen 'newpmar1', 1"
scpi.parse "Syst:conf:e dev:stat 'newpmar1', 1"

'Create a PMAR trace with power meter connected to port 3
'Use Calc:Par:Def:Ext - NOT CALC:PAR:DEF!!
scpi.parse "CALC:PAR:DEF:EXT 'myPMARTrace', 'newpmar1,3'"
Create a Swept IMDX Measurement

This program configures several Swept IMDx parameters using power sweep. In this configuration, tone power is swept from -20 dBm to -5 dBm while the Input, LO, and Output frequencies are fixed as follows:

- Input center freq = 2.50 GHz
- Tone Delta freq = 10 MHz (f1 = 2.495 GHz and f2 = 2.505 GHz).
- LO freq = 2.00 GHz
- Output freq = 4.50 GHz

This program also allows you to optionally load a .mxr file to perform mixer setup.

To run this program without error, an external source named 'PSG' must be connected to drive the LO.

This program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as IMDX.vbs. Learn how to setup and run the macro.

See all Swept IMD commands.

See all Mixer Setup commands.

---

See Other SCPI Example Programs

```
Dim app
Dim scpi
Dim err
`

'Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
`

'Preset the system
```
scpi.parse "SYST:FPR"

scpi.parse "DISP:WIND1:STAT ON"

'Create an IMDX measurement in Channel 1 and display it as trace 1 in window 1

scpi.parse "CALC1:CUST:DEF 'ch1IMDX', 'Swept IMD Converters', 'PwrMain'"

scpi.parse "DISP:WIND1:TRAC1:FEED 'ch1IMDX'"

'Put the channel in trigger hold

scpi.parse "SENS:SWE:MODE HOLD"

scpi.parse "SENS:IMD:SWE:TYPE POW"

'Put the channel in trigger hold and setup all the mixer parameters

scpi.parse "SENS1:SWE:MODE HOLD"

scpi.parse "SENS1:IMD:SWE:TYPE POW"

scpi.parse "SENS1:MIX:INP:FREQ:MODE FIXED"

scpi.parse "SENS1:MIX:LO:FREQ:MODE FIXED"

scpi.parse "SENS1:MIX:OUTP:FREQ:MODE FIXED"

scpi.parse "SENS1:MIX:INP:FREQ:FIX 2500000000"

scpi.parse "SENS1:MIX:LO:FREQ:FIX 2000000000"

scpi.parse "SENS1:MIX:OUTP:FREQ:SID HIGH"

scpi.parse "SENS1:MIX:CALC OUTP"

scpi.Parse "SENS:MIX:APPLY"

'First apply the settings, then set LO Name

scpi.Parse "SENS:MIX:LO:NAME 'PSG'"

scpi.parse "SENS1:MIX:LO:POW 10"
scpi.Parse "SENS:MIX:APPLY"

'Optionally, put the channel in hold and load an existing .mxr file with all the mixer settings
'scp.parse "SENS1:SWE:MODE HOLD"
'scp.parse "SENS1:MIX:LOAD 'c:\users\public\network analyzer\documents/Mixer/IMD/Ch1.mxr'"
'
'Make additional IMD settings
'scp.parse "SENS1:IMD:TPOW:COUP:STAT ON"
'scp.parse "SENS1:IMD:TPOW:F1:STAR -20"
'scp.parse "SENS1:IMD:TPOW:F1:STOP -5"
'scp.parse "SENS1:IMD:FREQ:DFR:CW 10000000"
'scp.parse "SENS1:SWE:POIN 201"
'scp.parse "SENS1:imd:ifbw:main 1000"
'scp.parse "SENS1:imd:ifbw:imt 500"
'scp.parse "SOUR1:POW2:AMPL -5"
'
'Create additional measurements in the channel
'scp.parse "CALC1:CUST:DEF 'ch1IMDX2', 'Swept IMD Converters', 'IM3'"
'scp.parse "DISP:WIND1:TRAC2:FEED 'ch1IMDX2'"
'scp.parse "CALC1:CUST:DEF 'ch1IMDX3', 'Swept IMD Converters', 'OIP3'"
'scp.parse "DISP:WIND1:TRAC3:FEED 'ch1IMDX3'"
'scp.parse "CALC1:CUST:DEF 'ch1IMDX4', 'Swept IMD Converters', 'IIP3'"
'scp.parse "DISP:WIND1:TRAC4:FEED 'ch1IMDX4'"
scpi.parse "CALC1:CUST:DEF 'ch1IMDX5', 'Swept IMD Converters', 'ToneGain'"

scpi.parse "DISP:WIND1:TRAC5:FEED 'ch1IMDX5'"

'Take a single sweep to apply all stimulus changes
scpi.parse "*cls;*ese 1"
scpi.parse "sens1:swe:mode SING;*OPC?"

'Check for errors
err=scpi.parse ("SYST:ERR?")
MsgBox(err)
Create a Wideband Pulsed Measurement using the PNA-X

This Visual Basic example shows you how to configure the PNA-X internal pulse generators and modulators to make wideband pulsed measurements in **pulse profile** mode using the PNA-X.

Visit the VNA website where you can download a free Wideband Pulsed Application that performs this measurement on the PNA-X.

See all SCPI Pulsed examples

```vbscript
Private Sub Form_Load()
    Dim app
    Dim scpi
    ' Create / Get the VNA application.
    Set app = CreateObject("AgilentPNA835x.Application")
    Set scpi = app.ScpiStringParser
    'Preset the analyzer
    scpi.Execute ("*RST")
    'Set BW to 5 MHz
    scpi.Execute ("SENS:BWID 5MHZ")
    'Set sweep type to CW mode
    scpi.Execute ("SENS:SWE:TYPE CW")
    'Delete S11 trace
    scpi.Execute ("DISP:WIND:TRAC1:DEL")
    'Create S21 trace
    scpi.Execute ("CALC:PAR:DEF:EXT 'MyMeas',S21")
    scpi.Execute ("DISP:WIND:TRAC1:FEED 'MyMeas'")
    'Set modulation source to Pulse1
    scpi.Execute ("sens:path:conf:elem 'PulseModDrive','Pulse1'")
    'Set power leveling mode to Openloop
```
scpi.Execute("sour:pow1:alc:mode open")

'Enable pulse modulator 1

scpi.Execute("sens:path:conf:elem 'Src1Out1PulseModEnable','Enable'")

'Set clock of internal pulse generator to internal

scpi.Execute("sens:path:conf:elem 'PulseTrigInput','Internal'")

'Turn on Pulse0

scpi.Execute("SENS:PULS0:STAT 1")

'Turn on Pulse1

scpi.Execute("SENS:PULS1:STAT 1")

'Set pulse period to 1 ms

scpi.Execute("sens:puls:per .001")

'Set Pulsel width to 10 us

scpi.Execute("sens:puls1:width 0.00001")

'Set Pulsel delay to 8 us

scpi.Execute("sens:puls1:delay 0.000008")

'Set Pulse0 width to 1 us

scpi.Execute("sens:puls0:width 0.00001")

'Set Pulse0 delay to 400 ns

scpi.Execute("sens:puls0:delay 0.000004")

'Set trigger scope to Channel

scpi.Execute("TRIG:SCOP CURRENT")

End Sub
Create an FOM Measurement

All three VBScript examples in this topic create a FOM measurement with the following attributes:

- Sweep the Source (input) from 1 GHz to 2 GHz
- Sweep the Receivers (output) from 2 GHz to 3 GHz
- You provide an LO at 1 GHz

Learn more about Frequency Offset Mode

These programs can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as FOM.vbs. Learn how to setup and run the macro.

See Other SCPI Example Programs

The following example will run on any VNA model with FOM (opt S93080A). However, these commands have no provisions for internal second source. It uses Sens:Offset commands.

```vbs
' This section gets the VNA application
' starts the scpi parser, and presets the VNA
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Execute("SYST:FPRESET")
' Create and turn on window 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")
' Define a measurement name, parameter
scpi.Execute ("CALCulate:PARameter:DEFine:EXT 'MyMeas',S21")
' Associate ("FEED") the measurement name ('MyMeas') to WINDow (1)
' and give the new TRACe a number (1).
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")
```
The following example can be run ONLY on a VNA with FOM (opt S93080A). It uses new Sens:FOM commands.

' This section gets the VNA application
' starts the scpi parser, and presets the VNA
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Execute("SYST:FPRESET")
' Create and turn on window 1
scpi.Execute("DISPlay:WINDow1:STATE ON")
'Define a measurement name, parameter
scpi.Execute("CALCulate:PARameter:DEFine 'MyMeas',S21")
'Associate ("FEED") the measurement name ('MyMeas') to WINDow (1),
and give the new TRACe a number (1).
scpi.Execute("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")

scpi.Execute("SENS:FREQ:START 1e9")
scpi.Execute("SENS:FREQ:STOP 2e9")
' set the receivers to be 2e9 -> 3e9
scpi.Execute("SENS:OFFS:OFFS 1e9")
scpi.Execute("SENS:OFFS ON")
The following example can be run ONLY on a VNA with FOM (opt S93080A). It uses the internal 2nd source for the fixed LO frequency.

```vna
' This section gets the VNA application
' starts the scpi parser, and presets the VNA
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Execute("SYST:FPRESET")

' Create and turn on window 1
scpi.Execute("DISPlay:WINDow1:STATE ON")

'Define a measurement name, parameter
scpi.Execute("CALCulate:PARameter:DEFine 'MyMeas',S21")

'Associate ("FEED") the measurement name ("MyMeas") to WINDow (1)
'and give the new TRACe a number (1).
scpi.Execute("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")

scpi.Execute("SENS:FREQ:START 1e9")
scpi.Execute("SENS:FREQ:STOP 2e9")

'set the receivers to be 2e9 -> 3e9
scpi.Execute("SENS:FOM:RANG3:FREQ:OFFS 1e9")

'setup the 2nd source frequencies
scpi.Execute("SENS:FOM:RANG4:COUP 0")
scpi.Execute("SENS:FOM:RANG4:FREQ:START 1e9")
scpi.Execute("SENS:FOM:RANG4:FREQ:STOP 1e9")

'turn off coupling
scpi.Execute("SOUR:POW:COUP 0")

'set LO power to 10dBm
scpi.Execute("SOUR:POW3 10")
```
'turn ON port 3, our LO signal

scpi.Execute("SOUR:POW3:MODE ON")

scpi.Execute("SENS:FOM:STAT ON")
Create an iTMSA Measurement

This example program does the following:

- Create an iTMSA Balanced Sdd21 measurement
- Set sweep type = power
- Set phase offset on balanced port 1 = 180 degrees

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Balanced.vbs. Learn how to setup and run the macro.

```vbs
Dim app
Dim scpi

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' Reset the system
scpi.Parse("SYST:FPRESET")

' This example uses DUT topology Bal-Bal -
' a DUT with a True-mode balanced input and balanced output.
'
' Port mapping for our DUT:
' logical port 1 = physical ports 1 and 3
' logical port 2 = physical ports 2 and 4
'
' logical 1          logical 2
' |------------------|------------------|
' 1 -------|------ DUT ------- 2 +
```

See Other SCPI Example Programs
' Turn on a window
scpi.Parse("DISP:WIND1:STATE ON")

' Create a trace called "sdd21", and for that trace turn on the balanced
transformation and set the balanced transformation to BBAL SDD21.
scpi.Parse("CALC:PAR:DEF:EXT ""sdd21"",S11")

' Feed the sdd21 trace to window 1, trace 1
scpi.Parse("DISP:WIND1:TRAC1:FEED ""sdd21"")
scpi.Parse("CALC:PAR:SEL ""sdd21"")

' Set the topology of measurement
scpi.Parse("CALC:FSIM:BAL:DEVice BBALanced")
scpi.Parse("CALC:FSIM:BAL:TOPology:BBAL:PPORts 1,3,2,4")

' Set up stimulus
scpi.Parse("SENS:SWE:POINTs 801")
scpi.Parse("SENS:FREQ:START 10e6")
scpi.Parse("SENS:FREQ:STOP 1e9")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")

' Recall a 4-port Cal Set or perform a 4-port Cal here

' Set the sweep type to power sweep
scpi.Parse("SENS:SWE:TYPE POWER")

' Set iTMSA parameters
Create an SMC Fixed Output Measurement

This VB Script example creates a calibrated SMC fixed output measurement using an external, controlled LO. Then a single sweep is taken and data is retrieved. The external LO is NOT required when using the internal second VNA source for the LO.

Requirements:

- The external LO should be configured to match the SENS:MIX:LO:NAME command below.

Fixed output measurements require that an external LO source be swept and synchronized with the VNA source. FCA performs this synchronization using the external source configuration settings. See Configure an External Source using SCPI.

The fastest, and recommended, method of controlling the LO source is Hardware List (BNC) triggering mode. However, in this mode, FCA channels will not respond to manual triggers. Therefore, the example uses the following mechanism to trigger a sweep:

Write "SENS:SWE:MODE HOLD" 'place channel 1 in HOLD mode
Write "INIT:CONT ON" 'place VNA in internal trigger mode
Write "SENS:SWE:MODE SINGLE
Write "*OPC?" 'wait until the sweep is complete
Read

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You can run a VBScript (*.vbs) program from the VNA using Macros. To run this program, copy the following code into a text editor and save it as a *.vbs file.

```vbs
option explicit
'
' Setup infrastructure to use the SCPI over COM

dim app
set app = createobject("Agilentpna835x.application")
dim p
set p = app.scpistringparser
dim returnStr
sub Write (command)
  if len(returnStr) <> 0 then
    err.Raise 55,"Write","Query Unterminated"
end if
end sub
```
end if
returnStr = p.parse(command)
end sub

sub WriteIgnoreError(command)
returnStr = p.Execute(command)
p.Parse("SYST:ERR?") ' clear error queue
end sub

function Read
if len(returnStr) = 0 then
    err.Raise 55,"Read","Bad read"
end if
Read = returnStr
returnStr = ""
end function

Write "SYST:PRES"
' When programming in remote mode, hold mode is recommended
Write "SENS:SWE:MODE HOLD"
' Delete the standard measurement
Write "CALC:PAR:DEL:ALL"
' Create an SC21 measurement
Write "CALC:CUST:DEF 'MySC21', 'Scalar Mixer/Converter', 'SC21'
Write "DISP:WIND:TRACE:FEED 'MySC21'
Write "CALC:PAR:SEL 'MySC21'
' Set number of points to 11
Write "SENS:SWE:POIN 11"
' Setup the mixer parameters for a swept LO, fixed output measurement
Write "SENS:MIX:INP:FREQ:START 200e6"
Write "SENS:MIX:INP:FREQ:STOP 700e6"
Write "SENS:MIX:LO:FREQ:MODE Swept"
Write "SENS:MIX:OUTPUT:FREQ:FIX 3.4e9"
Write "SENS:MIX:OUTP:FREQ:SID HIGH"
Write "SENS:MIX:CALC LO_1"
Write "SENS:MIX:INP:POW -17"
Write "SENS:MIX:LO:POW 10"
Write "SENS:MIX:APPLY"

' Specify the LO name, for controlled LO.
' This name is setup in the External Source Config Dialog
Write "SENS:MIX:LO:NAME '8360'"
Write "SENS:MIX:APPLY"

' Create an S11 in the same channel
Write "CALC:CUST:DEF 'MyS11', 'Scalar Mixer/Converter', 'S11'"
Write "DISP:WIND:TRACE2:FEED 'MyS11'"
Write "CALC:PAR:SEL 'MyS11'"

' Create an IPwr in the same channel
Write "CALC:CUST:DEF 'MyIPwr', 'Scalar Mixer/Converter', 'IPwr'"
Write "DISP:WIND:TRACE3:FEED 'MyIPwr'"
Write "CALC:PAR:SEL 'MyIPwr'"

' Create an OPwr in the same channel
Write "CALC:CUST:DEF 'MyOPwr', 'Scalar Mixer/Converter', 'OPwr'"
Write "CALC:PAR:SEL 'MyOPwr'"
Write "DISP:WIND:TRACE4:FEED 'MyOPwr'"

' Perform a single sweep, synchronously. When *OPC returns, the sweep is done
Write "SENS:SWE:MODE SING1e"
Write "*OPC?"
Read

' Retrieve the SC21 data
Write "CALC:PAR:SEL 'MySC21'"

Write "CALC:DATA? SDATA"

dim data

data = Read()

wscript.echo("SC21=" & data)

'Retrieve the S11 data

Write "CALC:PAR:SEL 'MyS11'"

Write "CALC:DATA? SDATA"

data = Read()

wscript.echo("S11=" & data)
Create and Cal a GCA Measurement

This VBS program does the following:

- creates and configures GCA to perform a SMART Sweep
- performs a calibration using an ECal with 3.5 mm Female on Port A and 3.5 mm Male connectors on Port B

This program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as GCA.vbs. Learn how to setup and run the macro.

See the Gain Compression commands

```vbs
option explicit

dim CompLevel , Tolerance , StartFreq , StopFreq , NumFreqs , Scale , LinearPower

dim AcqMode , BackOff , StartPower , StopPower , NumPowers , EnableInterp , CompAlg

dim DwellTime , IFBandwidth , ShowIterations , host , app , parser

' GCA Settings/Values

'' Acquisition Mode:

'' naSmartSweep = 0
'' naSweepPowerAtEachFreq2D = 1
'' naSweepFreqAtEachPower2D = 2

'' Compression Algorithm

'' naCompressionFromLinearGain = 0
'' naCompressionFromMaximumGain = 1
'' naBackoffCompression = 2
```
CompLevel = 1  ' 1 dB compression level
Tolerance = 0.05  ' SMART Sweep tolerance
StartFreq = 1E9
StopFreq = 9E9
NumFreqs = 201
Scale = 0.1
LinearPower = -20
BackOff = 10  ' Not used for Deviation from linear gain
StartPower = -20
StopPower = 8
NumPowers = 60  ' Not used for SMART Sweep
DwellTime = 0.0005  ' Allow some time for DUT bias/thermal effects
IFBandwidth = 1000  ' Reasonable trace noise at -20 dBm
EnableInterp = False  ' Disable interpolation
AcqMode = 0  ' Smart Sweep
CompAlg = 0  ' Deviation from linear gain
ShowIterations = False  ' Configure SMART to not show iteration results
dim objargs
set objargs = wscript . Arguments
if ( objArgs . Count = 1) then host = objargs (0)
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''}
call Analysis( parser )

'' GCA Setup

sub SetupGCA ( parser , StartFreq , StopFreq , NumFreqs , EnableInterp , Scale , CompLevel , LinearPower ,_
    AcqMode , BackOff , StartPower , StopPower , NumPowers , CompAlg , DwellTime , IFBAndwidth ,_
    ShowIterations )

parser . Parse "*RST "

parser . Parse "CALC:PAR:DEL:ALL "

parser . Parse "CALC:CUST:DEF ""S21"",""Gain Compression"",""S21"""

parser . Parse "DISP:WIND:TRAC1:FEED ""S21"""

parser . Parse "CALC:PAR:SEL ""S21"""

parser . Parse "CALC:CUST:DEF ""CompIn21"",""Gain Compression"",""CompIn21"""

parser . Parse "DISP:WIND:TRAC2:FEED ""CompIn21"""

parser . Parse "CALC:CUST:DEF ""DeltaGain21"",""Gain Compression"",""DeltaGain21"""


parser . Parse "SENS:SWE:MODE HOLD"

parser . Parse "DISP:WIND1:TRAC3:Y:SCAL:PDIV " & Scale

parser . Parse "DISP:WIND1:TRAC3:Y:RLEV " & -CompLevel

select case AcqMode
    case 0 ' SMART Sweep
        parser . Parse "SENS:GCS:AMOD SMAR"
    case 1 ' 2D Power Sweeps
select case CompAlg

case 0 ' Deviation from linear gain
  parser . Parse "SENS:GCS:COMP:ALG CFLG"

case 1 ' Deviation from max gain
  parser . Parse "SENS:GCS:COMP:ALG CFMG"

case 2 ' Back Off
  parser . Parse "SENS:GCS:COMP:ALG BACK"

case 3 ' XY
  parser . Parse "SENS:GCS:COMP:ALG XYCOM"

end select

if EnableInterp then
  parser . Parse "SENS:GCS:COMP:INT ON"
else
  parser . Parse "SENS:GCS:COMP:INT OFF"
end if

if ShowIterations then
  parser . Parse "SENS:GCS:SMAR:SIT ON"
else
  parser . Parse "SENS:GCS:SMAR:SIT OFF"
end if
parser . Parse "SENS:GCS:COMP:LEV " & CompLevel
parser . Parse "SENS:GCS:COMP:BACK:LEV " & BackOff
parser . Parse "SENS:GCS:COMP:DELT:X " & BackOff
parser . Parse "SENS:GCS:SMAR:STIM " & DwellTime
parser . Parse "SENS:BAND " & IFBandwidth
parser . Parse "SENS:SWE:DWEL " & DwellTime
parser . Parse "SOUR:POW:STAR " & StartPower
parser . Parse "SOUR:POW:STOP " & StopPower
parser . Parse "SENS:FREQ:STAR " & StartFreq
parser . Parse "SENS:FREQ:STOP " & StopFreq
parser . Parse "SENS:SWE:POIN " & NumFreqs
parser . Parse "SENS:SWE:MODE SING"

dim str
str = parser .Parse ("* OPC ?")

end sub
parser . parse "SENS:CORR:COLL:GUID:CONN:PORT1  'APC 3.5 female''
parser . parse "SENS:CORR:COLL:GUID:CONN:PORT2  'APC 3.5 male''
parser . parse "SENS:CORR:COLL:GUID:CKIT:PORT1 'N4691-60004 ECal'
parser . parse "SENS:CORR:GCSetup:POW 0"
parser . parse "SENS:CORR:COLL:GUID:INIT "
CalSteps = parser . parse ( " SENS:CORR:COLL:GUID:STEP ?" )
for I = 1 to CalSteps
    msgBox parser .parse ("SENS:CORR:COLL:GUID:DESC ? " & I )
    parser . parse ( "SENS:CORR:COLL:GUID:ACQ STAN"& I )
next
parser . parse "SENS:CORR:COLL:GUID:SAVE "
msgBox "Done"
end sub

.................................
'' GCA Analysis
........................................
sub Analysis( parser )
'select measurement 1
parser.parse "CALC:PAR:MNUM 1"
parsed.parse "CALC:GCM:ANAL:ENABLE 1" 'turn on the analysis mode
parser.parse "CALC:GCM:ANAL:CWFR 1e9" 'set the analysis cw frequency
'select measurement 2
parser.parse "CALC:PAR:MNUM 2"
parser.parse "CALC:GCM:ANAL:ENABLE 1"

parser.parse "CALC:GCM:ANAL:CWFR 2e9"

parser.parse "CALC:GCM:ANAL:XAX PSO" 'set the axis to power settings
'select measurement 3

parser.parse "CALC:PAR:MNUM 3"

parser.parse "CALC:GCM:ANAL:ENABLE 1"

parser.parse "CALC:GCM:ANAL:CWFR 3e9"

parser.parse "CALC:GCM:ANAL:ISD 0" 'set the discrete frequency option to false

end sub
Create and Cal a GCX Measurement

This VBS program does the following:

- creates and configures GCX
- performs a calibration using an ECal with 3.5 mm Female on Port A and 3.5 mm Male connectors on Port B

To run this example without modification you need the following:

- An ECal module that covers the frequency range of the measurement. You may need to change the ECal model number in the program.
- A power meter must be available to the VNA. This can be accomplished either by attaching the meter to the VNA via a GPIB cable, or by using SCPI over LAN.

By removing the comments (') at the start of the BLUE code, it can also do the following:

- Use ECal characterizations
- Specify Mechanical Cal Kits
- Perform manual ECal orientation.
- Specify the thru measurement method.
- Omit the isolation part of the 2-port cal.
- Perform an LO Power Cal.
- Set LO power level on External Source (MUST be pre-configured either remotely or using the GUI. See example program.)

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example. However, some modification is necessary to make the program run on a traditional GPIB Interface. For example, during the power meter portion of this calibration, scpi.Parse will not process a command until the power meter routine has completed. Traditional GPIB would require a serial polling technique to ensure the routine has completed before proceeding.

This program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as GCX.vbs. Learn how to setup and run the macro.
option explicit

dim CompLevel, Tolerance, StartFreq, StopFreq, LOFreq, NumFreqs, Scale, LinearPower

dim AcqMode, BackOff, StartPower, StopPower, NumPowers, EnableInterp, CompAlg

dim DwellTime, IFBandwidth, ShowIterations, host, app, parser, i

CompLevel = 1 ' 1 dB compression level
Tolerance = 0.05 ' SMART Sweep tolerance
NumFreqs = 21
Scale = 0.1
LinearPower = -10
BackOff = 10 ' Not used for Deviation from linear gain
StartPower = -20
StopPower = 8
NumPowers = 60 ' Not used for SMART Sweep
DwellTime = 0.0005 ' Allow some time for DUT bias/thermal effects
IFBandwidth = 1000 ' Reasonable trace noise at -20 dBm
EnableInterp = False ' Disable interpolation
AcqMode = 0 ' Smart Sweep
CompAlg = 0 ' Deviation from linear gain
ShowIterations = False ' Configure SMART to not show iteration results

dim objargs

set objargs = wscript.Arguments

if (objArgs.Count = 1) then host = objargs(0)

'' Create and Configuration GCX Channel:
set app = CreateObject("Agilentpna835x.application")
set parser = app.ScpiStringParser

GCX Setup

parser.Parse "*RST"
parser.Parse "CALC:PAR:DEL:ALL"
parser.Parse "CALC:CUST:DEF "SC21","Gain Compression Converters","SC21""
parser.Parse "DISP:WIND:TRAC1:FEED "SC21"
parser.Parse "CALC:PAR:SEL "SC21"
parser.Parse "CALC:CUST:DEF "CompIn21","Gain Compression Converters","CompIn21"
parser.Parse "DISP:WIND:TRAC2:FEED "CompIn21"
parser.Parse "CALC:CUST:DEF "DeltaGain21","Gain Compression Converters","DeltaGain21"
parser.Parse "DISP:WIND:TRAC3:FEED "DeltaGain21"
parser.Parse "SENS:SWE:MODE HOLD"
parser.Parse "DISP:WIND1:TRAC3:Y:SCAL:PDIV " & Scale
parser.Parse "DISP:WIND1:TRAC3:Y:RLEV " & -CompLevel

select case AcqMode
    case 0 ' SMART Sweep
        parser.Parse "SENS:GCS:AMOD SMAR"
    case 1 ' 2D Power Sweeps
        parser.Parse "SENS:GCS:AMOD PFREQ"
    case 2 ' 2D Freq Sweeps
        parser.Parse "SENS:GCS:AMOD FPOW"
end select

select case CompAlg

case 0 ' Deviation from linear gain
  parser.Parse "SENS:GCS:COMP:ALG CFLG"

case 1 ' Deviation from max gain
  parser.Parse "SENS:GCS:COMP:ALG CFMG"

case 2 ' Back Off
  parser.Parse "SENS:GCS:COMP:ALG BACK"

case 3 ' XY
  parser.Parse "SENS:GCS:COMP:ALG XYCOM"
end select

if EnableInterp then
  parser.Parse "SENS:GCS:COMP:INT ON"
else
  parser.Parse "SENS:GCS:COMP:INT OFF"
end if

if ShowIterations then
  parser.Parse "SENS:GCS:SMAR:SIT ON"
else
  parser.Parse "SENS:GCS:SMAR:SIT OFF"
end if

parser.Parse "SENS:GCS:COMP:LEV " & CompLevel
parser.Parse "SENS:GCS:COMP:BACK:LEV " & BackOff
parser.Parse "SENS:GCS:COMP:DELT:X " & BackOff
parser.Parse "SENS:GCS:SMAR:STIM " & DwellTime
parser.Parse "SENS:BAND " & IFBandwidth
parser.Parse "SENS:SWE:DWEL " & DwellTime
parser.Parse "SOUR:POW:STAR " & StartPower
parser.Parse "SOUR:POW:STOP " & StopPower
parser.Parse "SOUR:POW " & LinearPower
parser.Parse "SENS:SWE:POIN " & NumFreqs

' Mixer settings
parser.Parse "SENS:MIX:INPut:FREQ:MODE SWEPt"
parser.Parse "SENS:MIX:INPut:FREQ:STAR 3.6e9"
parser.Parse "SENS:MIX:INPut:FREQ:STOP 3.9e9"
parser.Parse "SENS:MIX:LO:FREQ:MODE FIXED"
parser.Parse "SENS:MIX:LO:FREQ:FIX 1e9"
parser.Parse "SENS:MIX:LO:POW 10"
parser.Parse "SENS:MIX:OUTP:FREQ:SID LOW"
parser.Parse "SENS:MIX:CALC Output"
parser.Parse "SENS:MIX:APPLY"
'First apply the settings, then set LO Name
parser.Parse "SENS:MIX:LO:NAME 'Port 3'"
parser.Parse "SENS:MIX:APPLY"
parser.Parse "SENS:MIX:SAVE "C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Mixer\MyMixer.mrx"

' sweep single
parser.Parse "SENS:SWE:MODE SING"

dim str
str =parser.Parse("*OPC?")
'' GCX Calibration
 ''

'----------------Perform A GCX Cal using SMC commands----------------------

Specify the connector types and the cal kits for each of the ports.

```
parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT1:SEL ""APC 3.5 male"
```

' Non-factory characterizations are specified as follows:

```
' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
' Uncomment the following lines to manually orient
' the ecal port A connected to VNA port 1
parser.Parse "SENS:CORR:PREF:ECAL:ORI OFF"
```

' Specify Mechanical cal kits

```
parser.Parse "sens:corr:coll:guid:ckit:port1 '85033D/E'"
```
'Optional settings

'Specify the thru measurement method.
parser.Parse "SENS:CORR:COLL:GUID:INIT"

'Always send an INIT command before the THRU command.
parser.Parse "SENS:CORR:COLL:GUID:PATH:TMET 1,2,""UNDEFINED THRU"

'Omit the isolation part of the 2-port cal (default behavior).
parser.Parse "SENS:CORR:COLL:GUID:ISOL NONE"

'Perform LO Power Cal

'Set the LO power level for the cal on an external PSG source.

'Initialize a Guided calibration.
parser.Parse "SENS:CORR:COLL:GUID:INIT"

'Tell the wizard to generate and report the number of steps in this cal.
Dim steps
Dim desc

'Determine the number of steps required to complete the calibration.
steps =parser.Parse ("SENS:CORR:COLL:GUID:STEP?")
For i = 1 To steps

'Display the prompt for each step
MsgBox (desc)

'Perform the measurement for each step
parser.Parse "SENS:CORR:COLL:GUID:ACQ STAN" & CStr(i)
Next

'Finish the cal and save the calset
parser.Parse("SENS:CORR:COLL:GUID:SAVE ON")

Msgbox("GCX cal saved to CH1_CALREG")
Create and Cal a Noise Figure Measurement

This example program creates a Noise Figure measurement, then calibrates the measurement.

You MUST change the ECal Identification strings (in **Blue** font).

Optional: Uncomment the following lines (in **Blue** font) to change these settings:

- Noise Receiver = Noise Receiver to Std (VNA) Receiver
- Cal Method = "Vector" to "Scalar"
- Receiver Characterization Method = "NoiseSource" to "PowerMeter"

This VBScript program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as NF.vbs. **Learn how to setup and run the macro.**

See the Noise figure commands.

---

See Other SCPI Example Programs

```vbs
' This section gets the VNA application
' starts the scpi parser, and presets the VNA
windowNum = 1
channelNum = 1
set pna=CreateObject("AgilentPNA835x.Application")
set scpi = pna.ScpiStringParser
' Create noise figure measurement
scpi.Parse "SYST:FPR"
scpi.Parse "DISP:WIND ON"
scpi.Parse "CALC:CUST:DEF 'noiseFig', 'Noise Figure Cold Source', 'NF'"
scpi.Parse "DISP:WIND:TRAC:FEED 'noiseFig'"
scpi.Parse "CALC:PAR:SEL 'noiseFig'"
```
' Substitute appropriate Ecal identification strings here

tunerEcal = "N4691-60004 ECal 02821"
pullEcal = "N4691-60004 ECal 02297"

' configure channel
ConfigureChannel
ConfigureNoiseSettings

' perform calibration
SetupNoiseSource
SetupCalAttributes_Insertable
FinishCalibration

' ----- Support subroutines ------

' Configure noise channel

sub ConfigureChannel
    scpi.Parse "SENS:FREQ:START 750MHz"
    scpi.Parse "SENS:FREQ:STOP 5.0GHz"
    scpi.Parse "SENS:SWEEP:POINTS 401"
    scpi.Parse "SENS:BWID 1.0E3"
end sub

' Configure noise-specific channel settings

sub ConfigureNoiseSettings
    scpi.Parse "SENS:NOIS:REC NOIsE"  'Use noise receivers
    scpi.Parse "SENS:NOIS:REC NORM"  'Use std PNA receiver
    scpi.Parse "SENS:NOIS:AVER:STAT ON"  ' turn averaging ON
    scpi.Parse "SENS:NOIS:AVER 40"  ' noise averaging
    scpi.Parse "SENS:NOIS:BWID 8MHz"  ' noise bandwidth
    scpi.Parse "SENS:NOIS:GAIN 30"  ' gain of noise
receiver

  scpi.Parse "SENS:NOIS:TEMP:AMB 301"  ' ambient temperature, in Kelvin

  scpi.Parse "SENS:NOIS:IMP:COUN 5"  ' number of tuner impedance states


  scpi.Parse "SENS:NOIS:TUN:INP 'B'"  ' orientation of tuner input port

  scpi.Parse "SENS:NOIS:TUN:OUTP 'A'"  ' orientation of tuner output port

  scpi.Parse "SENS:CORR:TCOL:USER:VAL 300"  ' noise source cold temperature

end sub

sub  SetupCalAttributes_Insertable

  scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT1  'APC 3.5 female'

  scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT2  'APC 3.5 male'

  scpi.Parse "SENS:CORR:COLL:GUID:CKIT:PORT1 " & pullEcal & "" ' port 1 calkit


  scpi.Parse "SENS:NOIS:SOUR:CONN 'APC 3.5 male'"  ' noise source connector type

  scpi.Parse "SENS:NOIS:SOUR:CKIT " & pullEcal & ""  ' noise source calkit

  scpi.Parse "SENS:NOISE:CAL:METHOD 'Vector'

  scpi.Parse "SENS:NOISE:CAL:METHOD 'Scalar'

  scpi.Parse "SENS:NOISE:CAL:RMETHOD 'NoiseSource'

  scpi.Parse "SENS:NOISE:CAL:RMETHOD 'Receiver"
Characterization method

' scpi.Parse "SENS:NOISE:CAL:RMETHod 'PowerMeter'"

scpi.Parse "SENS:CORR:COLL:GUID:INIT"

end sub

sub SetupNoiseSource

' specify the ENR file for the noise source

enrfile = "C:\Program Files(x86)\Keysight\Network Analyzer\Noise\346C_MY44420454.enr"

scpi.Parse "SENS:NOIS:ENR:FILENAME '" & enrfile & "'

' set noise source cold temperature

scpi.Parse "SENS:CORR:TCOLd:USER:VAL 301.1"

end sub

sub FinishCalibration

' Build the connection list and acquire the calibration

steps = scpi.Parse("SENS:CORR:COLL:GUID:STEPS?"

for i = 1 to steps


msgbox str

scpi.Parse "SENS:CORR:COLL:GUID:ACQ STAN" & i

next

scpi.Parse "SENS:CORR:COLL:GUID:SAVE 0"

wscript.echo "Calibration complete"

end sub
Create and Cal a VMC Measurement

The following example program sets up a 1-stage mixer, then performs a full VMC calibration. By removing the comments (' ) at the start of the BLU3E code, it can also do the following:

- Load a mixer setup file
- Use an ECal Module
- Perform manual ECal orientation
- Load a Mixer Characterization

See Also

Setup Converter commands
VMC Cal commands
All Guided Cal commands

Example - Perform a Mixer Characterization ONLY

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as VMC.vbs. Learn how to setup and run the macro.

```vbnet
Dim app
Dim scpi

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Parse "SYSTem:PRESet"

' Create a Vector Mixer Measurement
' First, delete all measurements on the channel
```
Create a forward scalar mixer measurement and configure it in channel 1.
The first parameter is a unique identifying string to allow subsequent
commands to be directed at this specific measurement.

`scpi.Parse "CALC:CUST:DEF 'My VC21', 'Vector Mixer/Converter', 'VC21'"`

Setup the new measurement as the 2nd trace in the active window

`scpi.Parse "DISP:WIND:TRAC2:FEED 'My VC21'"`

Make the new trace the active measurement

`scpi.Parse "CALC:PAR:SEL 'My VC21'"`

The parameters of the mixer measurement can now be configured.
This can be done by either using the SENS:MIX commands
for each of the parameters or by loading a mixer setup file.

Uncomment the following line to load a mixer setup file. The path
name for the mixer file may be loaded from other mapped drives.

`scpi.Parse "SENS:MIXer:Load 'c:\users\public\network analyzer\documents/Mixer/MyMixer.mxr'"`

Setup Stimulus
Points and IFBW are channel settings

`scpi.Parse "SENS:SWEep:POINts 21"
scpi.Parse "SENS:BANDwidth 1e3"

The rest are mixer settings

`scpi.Parse "SENS:MIX:LO:FREQ:MODE SWEPt"
scpi.Parse "SENS:MIX:INPut:FREQ:STAR 3.6e9"
scpi.Parse "SENS:MIX:INPut:FREQ:STOP 3.9e9"
scpi.Parse "SENS:MIX:LO:FREQ:MODE FIXED"
scpi.Parse "SENS:MIX:LO:FREQ:FIX 1e9"
scpi.Parse "SENS:MIX:LO:POW 10"`
scpi.Parse "SENS:MIX:OUTP:FREQ:SID LOW"
scpi.Parse "SENS:MIX:CALC Output"
scpi.Parse "SENS:MIX:LO:NAME 'Port 3'"
scpi.Parse "SENS:MIX:APPLY"

' Perform Cal

' Define the DUT connectors for at ports 1 and 2 of the VNA
scpi.Parse "sens:corr:coll:guid:conn:port1 'APC 3.5 female'"
scpi.Parse "sens:corr:coll:guid:conn:port2 'APC 3.5 male'"
scpi.Parse "sens:corr:coll:guid:conn:port3 'Not used'"
scpi.Parse "sens:corr:coll:guid:conn:port4 'Not used'"

' Specify Mechanical cal kits
scpi.Parse "sens:corr:coll:guid:ckit:port1 '85033D/E'"

' Specify an ECal module the same way
'scpi.Parse "sens:corr:coll:guid:ckit:port1 'N4691-60004 ECal'"

' Non-factory characterizations are specified as follows:

' When two or more ECal modules with the same model number are connected
also specify the serial number as follows:

' When Disk Memory ECal user characterizations are used,
specify both the User char and the serial number as follows:
' By default, VMC requires the measurement of a Calibration Mixer.
' To determine the conversion loss of the calmixer, the cal wizard
' will add a step to perform a 1 port cal at the output of the mixer.
' The following commands opt to perform the mixer
' characterization using a cal kit.
' Define the DUT connectors for the output of the characterization mixer
' Use (logical) Port 3. If it is already used by the DUT,
' then specify port 4.
scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT3 'APC 3.5 male'"
' Specify the mechanical cal kit for port 3
' To avoid performing the 1-port cal, provide the cal wizard with a
' mixer characterization file. Uncomment the following line to
' specify the characterization file. This S2P file will be read.
FILE,'c:\users\public\network analyzer\documents/MyMixer.s2p'"
'
' ECal orientation
' By default, auto orientation of the ecal module is performed
' Uncomment the following lines to manually orient the ecal
'scpi.Parse "SENS:CORR:PREF:ECAL:ORI OFF"
' for 2-port portion, ecal port A connected to VNA port 1
' for mixer char, ecal port A connected to cal mixer output
' the main calibration loop
' a description for the connection instructions is read
' and then the standard is acquired
dim steps, strPrompt
scpi.Parse "sens:corr:coll:guid:init"
steps=scpi.Parse ("sens:corr:coll:guid:steps?")
wscript.echo "Number of Steps = " + cstr(steps)
if (steps > 0) then ' otherwise an error condition occurred
for i = 1 to steps
  MsgBox strPrompt, vbOKOnly, step
  scpi.Parse ("sens:corr:coll:guid:acq STAN" + CStr(i))
next
MsgBox ("Cal is done!")
end if
Create and Cal a Swept IMD Measurement

This program does the following:

- Create IMD power and IM3 measurements
- Set sweep mode to Center Frequency Sweep
- Calibrate the IMD channel

This program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as IMD.vbs.

See Also

Learn how to setup and run the macro.
See the IMD commands.
See the IMD Cal commands

See Other SCPI Example Programs

```vbs
option explicit
'declare variables
dim SweepMode, StartDeltaFreq, StopDeltaFreq, NumFreqs, TonePower, CWFreq
dim app, hostname, parser
'' Sweep type:
'' naIMDToneCWSweep = 0
'' naIMDTonePowerSweep = 1
'' naIMDToneCenterFreqSweep = 2
'' naIMDDeltaFrequencySweep = 3
'' naIMDToneSegmentSweep = 4
'init variables
SweepMode = 3 ' Sweep DeltaF
```
StartDeltaFreq  = 100e3
StopDeltaFreq   = 1e9
NumFreqs       = 201
TonePower      = -7
CWFreq         = 5e9

' get host name from commandline

dim objargs
set objargs = wscript.arguments
if(objargs.Count = 1) then hostname = objargs(0)
set app = CreateObject("Agilentpna835x.application", hostname)
set parser = app.ScpiStringParser

call SetupIMD
call CalIMD

sub SetupIMD
  parser.Parse "*RST"
  parser.Parse "CALC:PAR:DEL:ALL"
  parser.Parse "CALC:CUST:DEF 'PwrMain','Swept IMD', 'PwrMain'"  ' create PwrMain measurement
  parser.Parse "DISP:WIND:TRAC1:FEED 'PwrMain'"
  parser.Parse "CALC:PAR:SEL 'PwrMain'"
  parser.Parse "CALC:CUST:DEF 'IM3', 'Swept IMD', 'IM3'"  ' create IM3 measurement
  parser.Parse "DISP:WIND:TRAC2:FEED 'IM3'"
  parser.Parse "SENS:SWE:MODE HOLD"

' set sweep mode

select case SweepMode
    case 0 ' CW sweep
        parser.Parse "SENS:IMD:SWE:TYPE CW"
    case 1 ' Power Sweep
        parser.Parse "SENS:IMD:SWE:TYPE POW"
    case 2 ' sweep Fc
        parser.Parse "SENS:IMD:SWE:TYPE FCEN"
    case 3 ' sweep DeltaF
        parser.Parse "SENS:IMD:SWE:TYPE DFR"
    case 4 ' segment sweep
        parser.Parse "SENS:IMD:SWE:TYPE SEGM"
end select

parser.Parse "SENS:IMD:FREQ:FCEN " & CWFreq ' Frequency Center
parser.Parse "SENS:IMD:FREQ:DFR:STAR " & startDeltaFreq ' Delta Frequency Start
parser.Parse "SENS:IMD:FREQ:DFR:STOP " & stopDeltaFreq ' Delta Frequency Stop
parser.Parse "SENS:IMD:TPOW:F1 " & TonePower ' F1 power
parser.Parse "SENS:IMD:TPOW:F2 " & TonePower ' F2 power

parser.Parse "SENS:SWE:POIN " & NumFreqs
parser.Parse "SENS:SWE:MODE SING"
end sub

sub CalIMD

' Configure IMD GuidedCal for the connector types and ECal module that will be used
' Substitute appropriate connector type and ECal identification strings here
' imd custom settings
' Set the Power Level at the power sensor to be used in calibration
parser.Parse "SENS:CORR:IMD:POW 0"

' Specify the connector type of the power sensor. If there is an adapter between
' the input port and the power sensor, specify the connector type here, and set
' the appropriate cal kit type for the connector so that extra calibration can be
' performed. To skip the calibration for the adapter, set the connection type to "Ignored"
' i.e.: SENS:CORR:IMD:SENS:CONN 'Ignored'
parser.Parse "SENS:CORR:IMD:SENS:CONN 'APC 3.5 female'"  'set power sensor connector type

' Set the Max product to calibrate, valid values are 3, 5, 7, and 9
parser.Parse "SENS:CORR:IMD:MPR 3"

' Set the calibration Frequencies, can choose between calibrate only at center Frequencies (CENT)
' or calibrate at all frequencies (ALL).
parser.Parse "SENS:CORR:IMD:CAL:FREQ ALL"
'Include 2nd order product in calibration

parser.Parse "SENS:CORR:IMD:SORD:INCL 1"

parser.Parse "SENS:CORR:COLL:GUID:INIT"

dim CalSteps, I


for I = 1 to CalSteps
next

parser.Parse "SENS:CORR:COLL:GUID:SAVE"

msgBox "IMD Cal Done"

end sub
Create and Cal an NFX Measurement

This program does the following:

- Setup a Noise Figure SC21 Measurement
- Calibrate Noise Figure channel
- Optional - Configure for an Embedded LO

To run this program, make the following edits, highlighted in yellow:

- Set `hostname` to your VNA computer name
- Set `tunerECal` and `pullECal` to your ECal model and info
- Set `ENR` to correct file name and location
- Set `connector types` for ECal, power sensor, and noise source

This program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as NFX.vbs. Learn how to setup and run the macro.

**See SCPI commands**

Calc:Custom command
Noise Figure commands
Mixer commands
Embedded LO commands
Guided Cal commands
Noise Cal commands

**See Other SCPI Example Programs**

```vbs
option explicit
dim app
dim hostname
```
dim parser

hostname = "MyPNA"

set app = CreateObject("Agilentpna835x.application", hostname)
set parser = app.ScpiStringParser

' ecal and noise tuner
dim tunerEcal
dim sParamECal
tunerEcal = "N4691-60003 ECal 00591"
sParamECal = "N4693-60001 User 2 ECal 00012"

' ENR file
dim ENRFile
ENRFile = "C:\Program Files(x86)\Keysight\Network Analyzer\Noise\346C_44420601.enr"
call SetupNFX

'optional if not doing embedded LO
'call SetupEmbeddedLO
call CalNFX

sub SetupNFX

'Create NF and SC21 traces
parser.Parse "*RST"
parsed.Parse "CALC:PAR:DEL:ALL"
parsed.Parse "CALC:CUST:DEF 'NF', 'Noise Figure Converters', 'NF'"
parsed.Parse "DISP:WIND:TRAC1:FEED 'NF'"
parsed.Parse "CALC:CUST:DEF 'SC21', 'Noise Figure Converters', 'SC21'"
set channel properties
set sweep type to linear sweep
set number of points
set IF bandwidth
set nfx properties
turn averaging on
noise averaging factor
noise tuner ecal module
noise tuner input
noise tuner output
noise bandwidth
low gain of noise receiver
' sweep single
parser.Parse "SENS:SWE:MODE SING"

'Mixer settings
parser.Parse "SENS:MIX:INPut:FREQ:STAR 3.6e9"
parser.Parse "SENS:MIX:INPut:FREQ:STOP 3.9e9"
parser.Parse "SENS:MIX:LO:FREQ:MODE FIXED"
parser.Parse "SENS:MIX:LO:FREQ:FIX 1e9"
parser.Parse "SENS:MIX:LO:POW 10"
parser.Parse "SENS:MIX:OUTP:FREQ:SID LOW"
parser.Parse "SENS:MIX:CALC Output"
parser.Parse "SENS:MIX:APPLY"
'First apply the settings, then set LO Name
parser.Parse "SENS:MIX:LO:NAME 'Port 3'"

parser.Parse "SENS:MIX:APPLY"
parser.Parse "SENS:MIX:SAVE "C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Mixer\MyMixer.mrx"

'sweep single
parser.Parse "SENS:SWE:MODE SING"

end sub

sub CalNFX
'dut connector
parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 female'"
parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 female'"
'port calkits
parser.Parse "SENS:CORR:COLL:GUID:CKIT:PORT1" & sParamEcal & ":0"

parser.Parse "SENS:CORR:COLL:GUID:CKIT:PORT2" & sParamEcal & ":0"

' power sensor connector

parser.Parse "SENS:CORR:COLL:GUID:PSEN1:CONN 'APC 3.5 female'"

' power sensor adapter cal kit

parser.Parse "SENS:CORR:COLL:GUID:PSEN1:CKIT" & sParamEcal & ":0"

' disable LO power cal

parser.Parse "SENS:CORR:COLL:NOIS:LO:PCAL 0"

' power calibration power level


' noise source connector

parser.Parse "SENS:NOIS:SOUR:CONN 'APC 3.5 male'"

' noise source adapter cal cal kit

parser.Parse "SENS:NOIS:SOUR:CKIT" & sParamEcal & ":0"

' cal method

parser.Parse "SENS:NOIS:CAL:METH 'Vector'"

' ENR file

parser.Parse "SENS:NOIS:ENR:FIL" & ENRFile & ":0"

' set force both adapter cals de-embed to false

parser.Parse "SENS:CORR:COLL:NOIS:ENR:ADAP:DEEM 0"

parser.Parse "SENS:CORR:COLL:NOIS:PSEN:ADAP:DEEM 0"
' initialize guided cal
parser.Parse "SENS:CORR:COLL:GUID:INIT"

' step through calsteps
dim steps
steps = parser.Parse("SENS:CORR:COLL:GUID:STEP?")
dim i, str
for i = 1 to steps
msgbox str
parser.Parse "SENS:CORR:COLL:GUID:ACQ STAN" & i
next

parser.Parse "SENS:CORR:COLL:GUID:SAVE 0"
parser.Parse "SENS:SWE:MODE CONT"
end sub

sub SetupEmbeddedLO
' embedded LO properties
' normalize point
parser.Parse "SENS:MIX:ELO:NORM:POIN 101"
' set tuning mode to broadband and precise
parser.Parse "SENS:MIX:ELO:TUN:MODE BRO"
' tuning ifbw
parser.Parse "SENS:MIX:ELO:TUN:IFBW 3e4"
' max tuning iterations
parser.Parse "SENS:MIX:ELO:TUN:ITER 5"
' tuning tolerance
parser.Parse "SENS:MIX:ELO:TUN:TOL 1"
' tuning interval
parser.Parse "SENS:MIX:ELO:TUN:INT 1"
' turn on ELO
parser.Parse "SENS:MIX:ELO:STAT 1"
' sweep single
parser.Parse "SENS:SWE:MODE SING"
end sub
Create and Cal an SMC Measurement

This VB Script example creates and calibrates a scalar mixer measurement.

To run this example **without modification** you need the following:

- An ECal module that covers the frequency range of the measurement.
- A power meter must be available to the VNA. This can be accomplished either by attaching the meter to the VNA via a GPIB cable, or by using SCPI over LAN.

By removing the comments (' ') at the start of the **BLUE code**, it can also do the following:

- Load a mixer setup file
- Use ECal characterizations
- Specify Mechanical Cal Kits
- Perform manual ECal orientation.
- Specify the thru measurement method.
- Omit the isolation part of the 2-port cal.
- Perform an LO Power Cal.
- Set LO power level on External Source (MUST be pre-configured either remotely or using the GUI. See example program.)
- Enable and configure phase measurements

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example. However, some modification is necessary to make the program run on a traditional GPIB Interface. For example, during the power meter portion of this calibration, sepi.Parse will not process a command until the power meter routine has completed. Traditional GPIB would require a **serial polling technique** to ensure the routine has completed before proceeding.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as SMC.vbs. **Learn how to setup and run the macro.**

```
Dim app
```
Dim scpi

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' Create a Scalar Mixer Forward Measurement
' First, delete all measurements on the channel
scpi.Parse "CALC:PAR:DEL:ALL"

' Create a forward scalar mixer measurement and configure it in
' channel 1. The first parameter is a unique
' identifying string (specified by the user) to allow subsequent
' commands to be directed at this specific measurement.
scpi.Parse "CALC:CUST:DEF 'My SC21', 'Scalar Mixer/Converter', 'SC21'"

' Setup the new measurement in the active window
scpi.Parse "DISP:WIND:TRAC:FEED 'My SC21'"

' Make the new trace the active measurement
scpi.Parse "CALC:PAR:SEL 'My SC21'"

' The parameters of the mixer measurement can now be configured.
' This can be done by either using the SENS:MIX commands
' for each of the parameters or by loading a mixer setup file.
' Uncomment the following line to load a mixer setup file. The path name
' for the mixer file may be loaded from other mapped drives.
' scpi.Parse "SENS:MIXer:Load 'c:\users\public\network
analyzer\documents/Mixer/MyMixer.mxr'"

' Setup Stimulus
' Points and IFBW are channel settings
scpi.Parse "SENS:SWEep:POINTs 21"
scpi.Parse "SENS:BANDwidth 1e3"

'Mixer settings
scpi.Parse "SENS:MIX:INPut:FREQ:STAR 3.6e9"
scpi.Parse "SENS:MIX:INPut:FREQ:STOP 3.9e9"
scpi.Parse "SENS:MIX:LO:FREQ:MODE FIXED"
scpi.Parse "SENS:MIX:LO:FREQ:FIX 1e9"
scpi.Parse "SENS:MIX:LO:POW 10"
scpi.Parse "SENS:MIX:OUTP:FREQ:SID LOW"
scpi.Parse "SENS:MIX:CALC Output"
scpi.Parse "SENS:MIX:APPLY"

'First apply the settings, then set LO Name
scpi.Parse "SENS:MIX:LO:NAME 'Port 3'"
scpi.Parse "SENS:MIX:APPLY"
scpi.Parse "SENS:MIX:SAVE "C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Mixer\MyMixer.mxrx"

'--------------Perform A Scalar Mixer Calibration----------------------

'Specify the connector types and the cal kits for each of the ports.
scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT1:SEL "APC 3.5 male"

' Non-factory characterizations are specified as follows:

' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
When Disk Memory ECal user characterizations are used,
specify both the User char and the serial number as follows:

Uncomment the following lines to manually orient
the ecal port A connected to VNA port 1
'scpi.Parse "SENS:CORR:PREF:ECAL:ORI OFF"

Specify Mechanical cal kits
'scpi.Parse "sens:corr:coll:guid:ckit:port1 '85033D/E'"

Optional settings
Specify the thru measurement method.
'Always send an INIT command before the Thru command.
'scpi.Parse "SENS:CORR:COLL:GUID:INIT"
'scpi.Parse "SENS:CORR:COLL:GUID:PATH:TMET 1,2,""UNDEFINED THRU"
'Omit the isolation part of the 2-port cal (default behavior).
'scpi.Parse "SENS:CORR:COLL:GUID:ISOL NONE"

Perform LO Power Cal
'Set the LO power level for the cal on an external PSG source.

Enable and configure Phase measurements
'scpi.Parse "SENS:MIX:PHAS 1"
'scpi.Parse "SENS:MIX:NORM:POIN 1"
'Using Fixed delay

'Initialize an SMC guided calibration.
scpi.Parse "SENS:CORR:COLL:GUID:INIT"

'Tell the wizard to generate and report the number of steps in this cal.
Dim steps
Dim desc

'Determine the number of steps required to complete the calibration.
steps = scpi.Parse("SENS:CORR:COLL:GUID:STEP?")
For i = 1 To steps

'Display the prompt for each step
MsgBox(desc)

'Perform the measurement for each step
scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN" & CStr(i))
Next

'Finish the cal and save the calset
scpi.Parse("SENS:CORR:COLL:GUID:SAVE ON")
Msgbox("SMC cal saved to CH1_CALREG")
Calibrate Multiple SMC Channels

This example allows you to calibrate multiple SMC channels while connecting the power meter and required standards or ECal module only once.

In the example program:

- Modify `chans = 2` to indicate the number of channels to calibrate.
- You can also change the connector type and cal kit for each port.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as MultChanCal.vbs. Learn how to setup and run the macro.

```vbs
Dim app
Dim scpi
Dim chans
Dim i
Dim steps
Dim desc

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
app.Preset

' Set number of channels to create
chans = 2
ReDim calset(chans - 1)

For i = 1 To chans
    chanStr = CStr(i+1)  ' calibrate on channels 2 and 3
    Dim parm, measName, sens, calc
```
parm = "S" & CStr(i) & CStr(i)
measName = "My" & parm
sens = "SENS" & chanStr
calc = "CALC" & chanStr

scpi.Parse calc & ":CUST:DEF ' & measName & ' , 'Scalar Mixer/Converter', ' & parm & ''

' Setup the new measurement as the 2nd trace in the active window
scpi.Parse "DISP:WIND:TRAC" & chanStr & ":FEED ' & measName & ''

'Make the new trace the active measurement
scpi.Parse calc & ":PAR:SEL ' & measName & ''

'-------------------Perform A FCA Mixer Calibration-------------------

'Set ports and cal kits for 2 port calibration portion
scpi.Parse sens & ":CORR:COLL:GUID:CONN:PORT1:SEL "APC 3.5 male"
scpi.Parse sens & ":CORR:COLL:GUID:CONN:PORT2:SEL "APC 3.5 female"

'ECal modules are specified with the same command

'Specify the thru measurement method.
scpi.Parse sens & ":CORR:COLL:GUID:PATH:TMET 1,2,""DEFINED THRU"

'Omit the isolation part of the 2-port cal
scpi.Parse sens & ":CORR:COLL:GUID:ISOL NONE"
'Initialize an SMC guided calibration.
scpi.Parse sens & "(:CORR:COLL:GUID:INIT"

'Determine the number of steps required to complete the calibration.
steps = scpi.Parse (sens & "(:CORR:COLL:GUID:STEP?"

Next
For j = 1 To CInt(steps)

'Display the prompt for each step
MsgBox (desc)

'Measure the same standard for each channel
For i = 1 To chans

    chanStr = CStr(i+1) ' channel number as string
    scpi.Parse "SENS" & chanStr & "(:CORR:COLL:GUID:ACQ STAN" & CStr(j)
    opc_comp = scpi.Parse("*OPC?"

    Next
Next

'Finish the cal and save the calsets
For i = 1 To chans

    calset(i - 1) = scpi.Parse("SENS" & CStr(i+1) & "(:CORR:COLL:GUID:SAVE ON")

    Next

MsgBox ("SMC Cals Complete!")
Create a Mixing Plan for a Dual-Stage, Fixed-Output Converter

This VB Script example creates a dual-stage mixer with a fixed output. While all the frequency ranges are explicitly programmed, the analyzer needs the calculate command at the end to satisfy the setup.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example. However, some modification is necessary to make the program run on a traditional GPIB Interface. For example, during the power meter portion of this calibration, scpi.Parse will not process a command until the power meter routine has completed. Traditional GPIB would require a serial polling technique to ensure the routine has completed before proceeding.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as SMC.vbs. Learn how to setup and run the macro.

```vbs
Dim app
Dim scpi

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Parse "SYSTem:PRESet"

' Create a Scalar Mixer Measurement

' First, delete all measurements on the channel
scpi.Parse "CALC:PAR:DEL:ALL"

' Create a forward scalar mixer measurement and configure it in channel 1. The first parameter is a unique identifying string (specified by the user) to allow subsequent commands to be directed at this specific measurement.
scpi.Parse "CALC:CUST:DEF 'My SC21', 'Scalar Mixer/Converter', 'SC21'"

' Setup the new measurement as the 2nd trace in the active window
scpi.Parse "DISP:WIND:TRAC2:FEED 'My SC21'"
```
' Setup Stimulus

' Points and IFBW are channel settings
scpi.Parse "SENS:SWEep:POINts 21"
scpi.Parse "SENS:BANDwidth 1e3"

' Mixer settings
scpi.parse "SENS:MIX:STAG 2"
scpi.parse "SENS:MIX:INP:FREQ:MODE SWEPT"
scpi.parse "SENS:MIX:INP:FREQ:STAR 170e6"
scpi.parse "SENS:MIX:INP:FREQ:STOP 210e6"
scpi.parse "SENS:MIX:LO:FREQ:MODE SWEPT"
scpi.parse "SENS:MIX:LO:FREQ:STAR 3.89e9"
scpi.parse "SENS:MIX:LO:FREQ:STOP 3.93e9"
scpi.parse "SENS:MIX:LO:FREQ:ILTI 0"
scpi.parse "SENS:MIX:IF:FREQ:SID LOW"
scpi.parse "SENS:MIX:LO2:FREQ:MODE FIXED"
scpi.parse "SENS:MIX:LO2:FREQ:FIX 3.56e9"
scpi.parse "SENS:MIX:LO2:FREQ:ILTI 1"
scpi.Parse "SENS:MIX:OUTP:FREQ:SID LOW"
scpi.parse "SENS:MIX:OUTP:FREQ:MODE FIXED"
scpi.parse "SENS:MIX:OUTP:FREQ:FIX 160e6"
scpi.Parse "SENS:MIX:CALC Input"
Create New Cal Kit using SCPI

When creating new cal kits programmatically, the order in which cal kit commands are sent can be important. For example to create a kit with opens, shorts, loads, and thurs. Be sure to use the following sequence for each newly defined standard.

1. Programmatically select the standard number
2. Programmatically select the standard type.
3. Program the cal standard's values.
4. Repeat steps 1, 2, 3 for additional new standards being defined.

```
10 !
20 !
30 ! This example program demonstrates how to create
40 ! new PNA calibration kits.
50 !
60 ! 1) Select a kit not previously defined
70 ! 2) Define open, short, load, and thru cal standards
80 ! Note: Each of the newly defined standards is assigned
90 ! a default connector name. These default connector names
100 ! will be replaced in subsequent steps.
110 ! 3) Use the delete connector command to remove default
120 ! connector names.
130 ! 4) Add connectors. Specify:
140 ! Start and Stop Freq
150 ! Z - Impedance
160 ! sex - MALE, FEMALE, NONE
170 ! media - COAX, WAVE
180 ! cutoff - Frequency for waveguide
190 ! 5) Assign the appropriate connector to each standard
200 ! 6) Modify the class assignments for the standards defined
210 ! 7) Verify the kit values
220 !
230 ! Additional Note: After setting each new cal kit value, it is
240 ! recommended that the program periodically perform queries to
250 ! verify the new values.
260 !
270 ! This will prevent program synchronization issues that can
```
280  ! final values stored within new cal kits.
290  !
300  !------------------------------------------------------------
310  !
320  ! Set up I/O path
330  ASSIGN @Na TO 716
340  DIM Calkname$[80],Conn$[80]
350  INTEGER Calkitnum
360  !
370  CLEAR SCREEN
380  !
390  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
400  ! Designate the kit selection to be used for performing cal's
410  OUTPUT @Na;".sens:corr:ckit:count?"
420  ENTER @Na;Calkitnum
430  Calkitnum=Calkitnum+1
440  OUTPUT @Na;".sens:corr:ckit:ckit "+VAL$(Calkitnum)
450  !
460  ! Name this kit with your own name
470  OUTPUT @Na;".sens:corr:coll:ckit:name \"Special 2.4 mm Model 85056\"
480  !
490  !
500  DISP "Defining kit std 1..."
510  ! Now set up standard #1
520  OUTPUT @Na;".sens:corr:coll:ckit:stan 1"
530  OUTPUT @Na;".sens:corr:coll:ckit:stan:type SHORT"
540  Get_std
550  OUTPUT @Na;".sens:corr:coll:ckit:stan:char coax"
560  OUTPUT @Na;".sens:corr:coll:ckit:stan:label \"My Short\"
570  Get_label
580  !
590  DISP "Defining kit std 2...
600  ! Now set up standard #2
610  OUTPUT @Na;".sens:corr:coll:ckit:stan 2"
620  OUTPUT @Na;".sens:corr:coll:ckit:stan:type OPEN"
630  Get_std
640  OUTPUT @Na;".sens:corr:coll:ckit:stan:char coax"
650  OUTPUT @Na;".sens:corr:coll:ckit:stan:label \"My Open\"
660  Get_label
670  !
680  DISP "Defining kit std 3..."
690  ! Now set up standard #3
700  OUTPUT @Na;".sens:corr:coll:ckit:stan 3"
DISP "Defining kit std 4..."

! Now set up standard #4

DISP "Defining kit std 5..."

! Now set up standard #5

DISP "Defining kit std 6..."

! Now set up standard #6

DISP "Defining kit std 7..."

! Now set up standard #7

DISP "Defining kit std 8..."
1150 ! Now set up standard #8
1160 !
1170 OUTPUT @Na;""sens:corr:coll:ckit:stan 8"
1190 OUTPUT @Na;""sens:corr:coll:ckit:stan:type ARBI"
1200 Get_std
1210 OUTPUT @Na;""sens:corr:coll:ckit:stan:char coax"
1220 OUTPUT @Na;""sens:corr:coll:ckit:stan:TZR 15;"
1230 OUTPUT @Na;""sens:corr:coll:ckit:stan:TZI -9;"
1240 OUTPUT @Na;""sens:corr:coll:ckit:stan:label ""Z Load"
1250 Get_label
1260 !
1270 !
1280 !
1290 ! First remove any old connector names
1300 OUTPUT @Na;""sens:corr:coll:ckit:conn:del"
1310 ! Verify that no connectors are currently installed
1320 OUTPUT @Na;""sens:corr:coll:ckit:conn:cat?"
1330 ENTER @Na;Conn$
1340 PRINT "Verify empty list: ";Conn$
1350 !
1360 ! Define your new connectors
1370 OUTPUT @Na;""sens:corr:coll:ckit:conn:add ""PSC 2.4",0HZ,999GHZ,50.0,MALE,COAX,0.0"
1380 OUTPUT @Na;""sens:corr:coll:ckit:conn:add ""PSC 2.4",0HZ,999GHZ,50.0,FEMALE,COAX,0.0"
1390 !
1400 ! Verify that the new connectors are installed
1410 OUTPUT @Na;""sens:corr:coll:ckit:conn:cat?"
1420 ENTER @Na;Conn$
1430 PRINT "Verify new connectors: ";Conn$
1440 DISP ""
1450 !
1460 DISP "Defining conn std 1..."
1470 ! Now set up standard #1
1480 OUTPUT @Na;""sens:corr:coll:ckit:stan 1"
1490 Verify_std
1500 OUTPUT @Na;""sens:corr:coll:ckit:conn:snam ""PSC 2.4",FEMALE,1"
1510 Print_connector
1520 !
1530 DISP "Defining conn std 2..."
1540 ! Now set up standard #2
1550 OUTPUT @Na;""sens:corr:coll:ckit:stan 2"
1560 Verify_std
1570 OUTPUT @Na;"sens:corr:coll:ckit:conn:snam ""PSC 2.4"",FEMALE,1"
1580 Print_connector
1590 !
1600 DISP "Defining conn std 3..."
1610 ! Now set up standard #3
1620 OUTPUT @Na;"sens:corr:coll:ckit:stan 3"
1630 Verify_std
1640 OUTPUT @Na;"sens:corr:coll:ckit:conn:snam ""PSC 2.4"",FEMALE,1"
1650 Print_connector
1660 !
1670 DISP "Defining conn std 4..."
1680 ! Now set up standard #4
1690 OUTPUT @Na;"sens:corr:coll:ckit:stan 4"
1700 Verify_std
1710 OUTPUT @Na;"sens:corr:coll:ckit:conn:snam ""PSC 2.4"",FEMALE,1"
1720 OUTPUT @Na;"sens:corr:coll:ckit:conn:snam ""PSC 2.4"",MALE,2"
1730 Print_connector
1740 !
1750 DISP "Defining conn std 5..."
1760 ! Now set up standard #5
1770 OUTPUT @Na;"sens:corr:coll:ckit:stan 5"
1780 OUTPUT @Na;"sens:corr:coll:ckit:stan:label ""Sliding Load"
1790 Verify_std
1800 OUTPUT @Na;"sens:corr:coll:ckit:conn:snam ""PSC 2.4"",MALE,1"
1810 Print_connector
1820 !
1830 DISP "Defining conn std 6..."
1840 ! Now set up standard #6
1850 !
1860 OUTPUT @Na;"sens:corr:coll:ckit:stan 6"
1870 Verify_std
1880 OUTPUT @Na;"sens:corr:coll:ckit:conn:snam ""PSC 2.4"",MALE,1"
1890 Print_connector
1900 !
1910 DISP "Defining conn std 7..."
1920 ! Now set up standard #7
1930 OUTPUT @Na;"sens:corr:coll:ckit:stan 7"
1940 Verify_std
1950 OUTPUT @Na;"sens:corr:coll:ckit:conn:snam ""PSC 2.4"",MALE,1"
1960 Print_connector
1970 !
1980  DISP "Defining conn std 8..."
1990  ! Now set up standard #8
2000  OUTPUT @Na;"sens:corr:coll:ckit:stan 8"
2010  Verify_std
2020  OUTPUT @Na;"sens:corr:coll:ckit:conn:snam ""PSC 2.4"",MALE,1"
2030  Print_connector
2040  !
2050  DISP "Class assignments..."
2060  ! Designate the "order" associated with measuring the standards
2080  !
2090  !   Set Port 1, 1st standard measured to be standard #2
2100  OUTPUT @Na;"sens:corr:coll:ckit:order1 2"
2110  !   Set Port 1, 2nd standard measured to be standard #1
2120  OUTPUT @Na;"sens:corr:coll:ckit:order2 1,6,7"
2130  !   Set Port 1, 3rd standard measured to be standard #3 and #5
2140  OUTPUT @Na;"sens:corr:coll:ckit:order3 3,5"
2150  !   Set Port 1, 4th standard measured to be standard #4
2160  OUTPUT @Na;"sens:corr:coll:ckit:order4 4"
2170  !
2180  !   Set Port 2, 1st standard measured to be standard #2
2190  OUTPUT @Na;"sens:corr:coll:ckit:order5 2"
2200  !   Set Port 2, 2nd standard measured to be standard #1
2210  OUTPUT @Na;"sens:corr:coll:ckit:order6 1,6,7"
2220  !   Set Port 2, 3rd standard measured to be standard #3 and #6
2230  OUTPUT @Na;"sens:corr:coll:ckit:order7 3,5"
2240  !   Set Port 2, 4th standard measured to be standard #4
2250  OUTPUT @Na;"sens:corr:coll:ckit:order8 4"
2260  !
2270  !   Set Port 1, 1st standard
2280  OUTPUT @Na;"sens:corr:coll:ckit:olabel1 ""MyOpen1""
2290  !   Set Port 1, 2nd standard
2300  OUTPUT @Na;"sens:corr:coll:ckit:olabel2 ""MyShorts1""
2310  !   Set Port 1, 3rd standard
2320  OUTPUT @Na;"sens:corr:coll:ckit:olabel3 ""MyLoads1""
2330  !   Set Port 1, 4th standard measured to be standard #4
2340  OUTPUT @Na;"sens:corr:coll:ckit:olabel4 ""MyThru1"
2350  !
2360  !   Set Port 2, 1st standard
2370  OUTPUT @Na;"sens:corr:coll:ckit:olabel5 ""MyOpen2"
2380  !   Set Port 2, 2nd standard
2390  OUTPUT @Na;"sens:corr:coll:ckit:olabel6 ""MyShorts2"
2400  !   Set Port 2, 3rd standard
2410  OUTPUT @Na;"sens:corr:coll:ckit:olabel7 ""MyLoads2""
2420 !  Set Port 2, 4th standard
2430 OUTPUT @Na;"sens:corr:coll:ckit:olabel8 ""MyThrus2"
2440 !
2450 BEEP
2460 DISP "Done!"
2470 END
2480 SUB Get_label
2490  OUTPUT 716;"sens:corr:coll:ckit:stan:label?"
2500  ENTER 716;Label$
2510  PRINT Label$
2520 SUBEND
2530 !
2540 SUB Get_std
2550  OUTPUT 716;"sens:corr:coll:ckit:stan:type?"
2560  ENTER 716;Type$
2570  PRINT Type$
2580 SUBEND
2590 !
2600 SUB Print_connector
2610   DIM Nam$[40]
2620  OUTPUT 716;"sens:corr:coll:ckit:conn:sname?"
2630  ENTER 716;Nam$
2640  PRINT Nam$
2650 SUBEND
2660 !
2670 SUB Verify_std
2680  OUTPUT 716;"sens:corr:coll:ckit:stan:label?"
2690  ENTER 716;Label$
2700 SUBEND
2710 !
This topic requires that you have a working knowledge of Visual Basic.

This topic will help you create your own power meter driver for use with Source Power Calibration on the VNA. If you are using an Keysight Power Meter to perform a Source Power Calibration, you do NOT need to create your own driver.

Your Power Meter driver will be created from a template written in Visual Basic using VISA over the GPIB bus.

**Note:** This procedure applies to Visual Basic 6.0. Applicability to Visual Basic .NET has not yet been investigated.

- **Prepare Template Files**
- **Modify Template Files**
- **Compile, Copy, and Register, Your New Driver**
- **Test Your new Driver**

### Other SCPI Example Programs

#### Prepare Template Files

1. Copy all the files from the VNA hard drive C:\Program Files(x86)\Keysight\Network Analyzer\Automation\Power Meter Driver Templates folder, to a folder on your development PC.

2. In Visual Basic click *File*, then *Open Project...*, find `MyPowerMeter.vbp` (a file you copied from the VNA). Click *Open*. This is a VB ActiveX EXE template, which you will fill in to become your driver.

3. Click *Project*, then *MyPowerMeter Properties*. Click the *General* tab.

4. Overwrite the Project Name with a name of your own choosing. This will be the name of your driver’s type library (also the default name of your exe).

   **Note** If the name of your exe does not match the VB Project Name with which it was compiled, registration of the exe on the VNA will not succeed.

5. Set the Project Description. After building your driver if you wish to test it using VB, this is the string that will show up in the VB References list of your test project, and also in the lower pane of the VB Object Browser.
6. Set the Thread Pool size to 1 thread.

7. Click OK to close the project properties dialog.

8. From the VB Project menu, click References… Ensure that Keysight PNA Power Meter 1.0 Type Library and VISA Library are checked. Click OK.

**Note:** Keysight's implementation of VISA is installed as part of the Keysight I/O Libraries on the VNA. For help on VISA, go to the Windows Start button on your VNA, select Programs, Keysight IO Libraries, VISA Help.

---

### Modify Template Files

From Visual Basic View menu click Project Explorer. Expand the Modules and Class Modules folders. Ensure there is one module (WinAPI) and one class module (PowerMeter).

Let's look at the WinAPI module first.

1. In the Project Explorer window, click WinAPI.

2. From the View menu click Code.

There is only one line of code you should need to modify in this module: the value of the string constant named sIDSEARCH. The comments preceding the declaration of that string describe how to change it. The rest of this module contains functions which will use the Microsoft Windows API to insure proper registration of your driver on the VNA. If you know of other Windows API functions you feel might be helpful to call from within your PowerMeter class module (to help in formatting data, for example), this module would be the place to declare them.

Now let's look at the class module.

1. In the Project Explorer window, click PowerMeter.

2. From the View menu click Properties Window. The Instancing property must be set to MultiUse. This allows other applications to create objects from this class, such that one instance of your driver EXE can supply more than one such object at a time.

3. From the View menu click Code.

Do NOT modify the Interfaces to IPowerMeter subroutines and functions. VNA source power cal expects to find these interfaces as they are currently defined.

The only members that you need to supply code to are those containing “Your code here” comments.
In addition, comments have been provided at the beginning of each member to describe the information that member needs to be read from or written to the power meter.

To get an idea of how communicate with the power meter using the VISA functions `viWrite` and `viRead`, examine the code which has been implemented for you in `IPowerMeter_Connect`, `IPowerMeter_QueryMeter`, and `IPowerMeter_WriteMeter`.

### Compile, Copy, and Register Your New Driver

When your driver is ready to run, you will first need to compile it into an EXE.

From the File menu select **Make exe**.

After compiling, the following will instruct VB to use the same ID (GUID) every time you re-compile your project.

1. From the **Project** menu, click **PowerMeter Properties**.
2. On the **Component** tab, select **Binary Compatibility** and click **...**
3. Browse to and select your project EXE. Click **Open**.
4. Click **OK** to close **Project Properties**.
5. Save your project.
6. Copy your driver EXE file to a folder on your VNA (do NOT use C:\Program Files(x86)\Keysight\Network Analyzer\Automation\Power Meter Driver Templates folder).
7. Run the EXE file. A message box will pop up reporting whether or not registration was successful. If not successful, it will make a suggestion on what to fix.

When your driver is properly registered, VNA Source Power Cal should be able to associate it with the ID string of your power meter.

### Test Your Power Meter Driver

We have also provided a Visual Basic project to test your new Power Meter driver. This project individually calls every `IPowerMeter` method and property in your driver to verify that it performs correctly. Before running the test your PC and VNA must be configured to communicate using DCOM.

1. Connect your PC and the VNA to LAN.
2. Add your PC logon to the VNA. Both logons and password must match to communicate using DCOM. See Additional VNA users.

3. Configure your driver using DCOM Config on the VNA. This will give you permission to launch and access the driver. See Configure for COM-DCOM Programming.

Modify the Test Project

1. In Visual Basic click File, then Open Project..., find MyPowerMeterTest.vbp (a file you copied from the VNA). Click Open.

2. From the Project menu, click References... From the list, find and check your new Power Meter Driver. (It should have been registered on your PC when you successfully made your driver EXE.) Click OK.

3. From the View menu click Code.

4. Modify the CreateObject line as follows:
   Replace MyPowerMeter with the Project Name that you chose for your driver
   Replace MyPNA with the Computer Name of your VNA.
   For example:

   ```vba
   Set PowerMeterObj = CreateObject("AcmeBrand.PowerMeter", "AGILENT-PNA123")
   ```

   (This assumes that you kept PowerMeter as class module name in your driver.)

Run the Test Project

Ensure your power meter is connected to the VNA with a GPIB cable.

Put the VNA in system controller mode:

1. From the VNA System menu point to Configure then click SICL/GPIB.

2. In the GPIB box click System Controller.

Run the test project. If there are no errors, the driver is created successfully. If there are errors, try to figure out what went wrong and fix it. Then re-compile, re-copy the .exe to the VNA, and re-run the test. You should not need to re-register the driver or re-modify the test program.
ECAL Confidence Check using SCPI

This Visual Basic program performs a complete ECAL confidence check.

To run this program, you need:

- An established GPIB interface connection
- Keysight's VISA or National Instrument's VISA installed on your PC
- The module visa32.bas added to your VB project.
- A form with two buttons: cmdRun and cmdQuit
- A calibrated S11 1-port or N-port measurement active on Channel 1
- Window 1 is visible

**Note:** A confidence check can NOT be performed remotely from User Characterizations that are stored on the VNA disk.

---

**See Other SCPI Example Programs**

' *Session to VISA Default Resource Manager*
Private defRM As Long
' *Session to VNA*
Private viPNA As Long
' *VISA function status return code*
Private status As Long

Private Sub Form_Load()
  defRM = 0
End Sub

Private Sub cmdRun_Click()
' *String to receive data from the VNA*
Dim strReply As String * 200

' Open the VISA default resource manager
status = viOpenDefaultRM(defRM)
If (status < VI_SUCCESS) Then HandleVISAError

' Open a VISA session (viPNA) to the VNA at GPIB address 16.
status = viOpen(defRM, "GPIB0::16::INSTR", 0, 0, viPNA)
If (status < VI_SUCCESS) Then HandleVISAError

```
' Need to set the VISA timeout value to give all our GPIB Reads
' sufficient time to complete before a timeout error occurs.
' For this example, let’s try setting the limit to
' 10000 milliseconds (10 seconds).
status = viSetAttribute(viPNA, VI_ATTR_TMO_VALUE, 10000)
If (status < VI_SUCCESS) Then HandleVISAError

' Get the catalog of all the measurements currently on Channel 1.
status = myGPIBWrite(viPNA, "CALC1:PAR:CAT?"
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' If an S11 measurement named "MY_S11" doesn’t already exist, 
' then create it.
If InStr(strReply, "MY_S11") = 0 Then
   status = myGPIBWrite(viPNA, "CALC1:PAR:DEF:EXT MY_S11,S11")
   If (status < VI_SUCCESS) Then HandleVISAError
End If

strReply = ""

' Get the catalog of all the trace numbers currently active
' in Window 1.
status = myGPIBWrite(viPNA, "DISP:WIND1:CAT?"
If (status < VI_SUCCESS) Then HandleVISAError

status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' If a trace number 4 already exists in Window 1, then this 
' will remove it.
If InStr(strReply, "4") > 0 Then
   status = myGPIBWrite(viPNA, "DISP:WIND1:TRAC4:DEL"
   If (status < VI_SUCCESS) Then HandleVISAError
End If

' Set trace number 4 to MY_S11.
status = myGPIBWrite(viPNA, "DISP:WIND1:TRAC4:FEED MY_S11"
If (status < VI_SUCCESS) Then HandleVISAError

' Set up trace view so we are viewing only the data trace.
status = myGPIBWrite(viPNA, "DISP:WIND1:TRAC4 ON"
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBWrite(viPNA, "DISP:WIND1:TRAC4:MEM OFF"
If (status < VI_SUCCESS) Then HandleVISAError

' Select MY_S11 as the measurement to be used for the
' Confidence Check.
status = myGPIBWrite(viPNA, "SENS1:CORR:CCH:PAR MY_S11"
If (status < VI_SUCCESS) Then HandleVISAError

' Acquire the S11 confidence check data from ECal Module A
' into the memory buffer (asking for an OPC reply when it's done).
status = myGPIBWrite(viPNA, "SENS1:CORR:CCH:ACQ ECAL1:*OPC?")
If (status < VI_SUCCESS) Then HandleVISAError

' The VNA sends an OPC reply ("+1") when the confidence data
' acquisition into memory is complete, so this Read is waiting on
' the reply until it is received.
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Turn on trace math so the trace shows data divided by memory.
' You can be confident the S11 calibration is reasonably good if
' the displayed trace varies no more than a few tenths of a dB
' from 0 dB across the entire span.
status = myGPIBWrite(viPNA, "CALC1:PAR:SEL MY_S11")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBWrite(viPNA, "CALC1:MATH:FUNC DIV")
If (status < VI_SUCCESS) Then HandleVISAError
End Sub

Private Sub cmdQuit_Click()
' Turn off trace math
status = myGPIBWrite(viPNA, "CALC1:MATH:FUNC NORM")
If (status < VI_SUCCESS) Then HandleVISAError

' Conclude the confidence check to set the ECal module
' back to its idle state.
status = myGPIBWrite(viPNA, "SENS1:CORR:CCH:DONE")
If (status < VI_SUCCESS) Then HandleVISAError

' Close the resource manager session (which also closes
' the session to the VNA).
If defRM <> 0 Then Call viClose(defRM)

' End the program
End
End Sub

Private Function myGPIBWrite(ByVal viHandle As Long, ByVal strOut As String) As Long
' The "+ Chr$(10)" appends an ASCII linefeed character to the output, for
' terminating the write transaction.
myGPIBWrite = viVPrintf(viHandle, strOut + Chr$(10), 0)
End Function

Private Function myGPIBRead(ByVal viHandle As Long, strIn As String) As Long
myGPIBRead = viVScanf(viHandle, "%t", strIn)
End Function

Sub HandleVISAError()
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr, vbExclamation
End
End Sub
Establish a VISA Session

This Visual Basic program demonstrates how to send a SCPI command using VISA and the Keysight IO libraries. To run this program, you need:

- Your PC and VNA both connected to a LAN (for communicating with each other).
- The SICL and VISA components of Keysight’s I/O Libraries software installed on your PC. Both are included when you install the software, unless you already have another vendor’s VISA installed. Then specify Full SICL and VISA installation to overwrite the other vendor’s VISA.
- The module visa32.bas added to your VB project. After you install VISA, the module will be located at C:/VXIPNP/WINNT (or equivalent)/INCLUDE/Visa32.bas
- A form with two buttons: cmdRun and cmdQuit.
- Your PC configured to be a VISA LAN Client, and the SICL Server capability enabled on the analyzer. See Configure for VISA and SICL

See Other SCPI Example Programs

**Note:** This example is a piece of a larger VISA program that performs a source power calibration.

```vbnet
' Session to VISA Default Resource Manager
Private defRM As Long
' Session to VNA
Private viPNA As Long
' VISA function status return code
Private status As Long

Private Sub Form_Load()
    defRM = 0
End Sub

Private Sub cmdRun_Click()
    ' String to receive data from the VNA.
    ' Dimensioned large enough to receive scalar comma-delimited values
    ' for 21 frequency points (20 ASCII characters per point)
    Dim strReply As String * 420

    ' Open the VISA default resource manager
    status = viOpenDefaultRM(defRM)
    If (status < VI_SUCCESS) Then HandleVISAError

    ' Open a VISA session (viPNA) to the SICL LAN server
    ' at “address 16” on the VNA pointed to by the “GPIB0”
    ' VISA LAN Client on this PC.
    ' CHANGE GPIB0 TO WHATEVER YOU VNA IS SET TO
```
status = viOpen(defRM, "GPIB0::16::INSTR", 0, 0, viPNA)
If (status < VI_SUCCESS) Then HandleVISAError

' Need to set the VISA timeout value to give all our calls to
' myGPIBRead sufficient time to complete before a timeout
' error occurs.
' For this example, let's try setting the limit to
' 30000 milliseconds (30 seconds).
status = viSetAttribute(viPNA, VI_ATTR_TMO_VALUE, 30000)
If (status < VI_SUCCESS) Then HandleVISAError

' Preset the VNA
status = myGPIBWrite(viPNA, "SYST:PRES")
If (status < VI_SUCCESS) Then HandleVISAError

' Print the data using a message box
MsgBox strReply
End Sub

Private Sub cmdQuit_Click()
' Close the resource manager session (which also closes
' the session to the VNA).
If defRM <> 0 Then Call viClose(defRM)

' End the program
End End Sub

Private Function myGPIBWrite(ByVal viHandle As Long, ByVal strOut As String) As Long
' The "+ Chr$(10)" appends an ASCII linefeed character to the
' output, for terminating the write transaction.
myGPIBWrite = viVPrintf(viHandle, strOut + Chr$(10), 0)
End Function

Private Function myGPIBRead(ByVal viHandle As Long, strIn As String) As Long
myGPIBRead = viVScanf(viHandle, "%t", strIn)
End Function

Sub HandleVISAError()
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr, vbExclamation
End
End Sub
This program demonstrates the use of several External Test Set Control commands.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as ExtTS.vbs. Learn how to setup and run the macro.

```vbs
' Demonstrate some SCPI commands for external testsets.
Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")
Set scpi = pna.ScpiStringParser
' The K64 testset is only usable on a 4-port VNA
If (pna.NumberOfPorts <> 4) Then
    MsgBox("This program only runs on 4-port analyzers.")
Else
    ' If Help is active, show the measurement window and help
    scpi.Execute("DISP:ARR TILE")
    ' Return the list of supported test sets
    list=scpi.Execute("SENS:MULT:CATalog?")
    MsgBox(list)
    '************* K64 *****************
    ' The K64 is connected using the Testset I/O
    ' connector. There is no handshake information.
    ' Therefore, a testset need not be connected.
    ' Load a configuration file.
    scpi.Execute("SENS:MULT1:TYPE 'Z5623AK64'")
    scpi.Execute("SENS:MULT1:ADDR 0")
```

5272
'return stuff about the test set

' Returns number of input ports
Inports = scpi.Execute("SENS:MULT1:INCount?")
MsgBox("Input Ports: " & CStr(Inports))

' Returns number of output ports
ports = scpi.Execute("SENS:MULT1:COUNt?")
MsgBox("Output Ports: " & CStr(ports))

' Returns valid output ports for each input port
For portNum = 1 To Inports
    ports = scpi.Execute("SENS:MULT1:PORT" & CStr(portNum) & ":CAT?"")
    MsgBox("Port " & CStr(portNum) & " catalog: " & (ports))
Next

'Set different port mapping
scpi.Execute("SENS:MULT1:ALLPorts '1 ext R, 2 ext R, 3 ext R, 4 ext R'"")

'Return port mapping
portMap = scpi.Execute("SENS:MULT1:ALLPorts?"")
MsgBox("Ports will be mapped to " & CStr(portMap))

' Enable external testset control and execute port mapping. This automatically enables status bar display as well.
scpi.Execute("SENS:MULT1:STATe 1")
MsgBox("Z5623A K64 Enabled")
End If
Create fixture function (impedance conversion and port matching)

This example programs below create single-ended port impedance conversion function and add network circuit by loading/recall the file from C drive.

The first program uses Fsimulator Draft and Fsimulator Active SCPI commands while the second program uses legacy SCPI commands. Learn about Using Fixture Simulator.

See Other SCPI Example Programs

Example program with  Fsimulator Draft and Fsimulator Active SCPI commands

```
*CLS
*OPC?
SYST:PRES
calc:fsim:draft:discard
calc1:fsim:draft:zcon:send:port1:state 1
calc1:fsim:draft:zcon:send:port1:state?
calc1:fsim:draft:zcon:send:port1:SCAL 50
calc1:fsim:draft:zcon:send:port1:SCAL?
calc1:fsim:draft:zcon:send:port2:state 1
calc1:fsim:draft:zcon:send:port2:state?
calc1:fsim:draft:zcon:send:port2:SCAL 100
calc1:fsim:draft:zcon:send:port2:SCAL?
calc1:fsim:draft:zcon:send:port3:state 1
calc1:fsim:draft:zcon:send:port3:state?
calc1:fsim:draft:zcon:send:port3:SCAL 100
calc1:fsim:draft:zcon:send:port3:SCAL?
calc1:fsim:draft:zcon:send:port4:state 1
calc1:fsim:draft:zcon:send:port4:state?
```
calc1:fsim:draft:zcon:send:port4:SCAL 50
calc1:fsim:draft:zcon:send:port4:SCAL?
calc:fsim:draft:circ:reset
calc:fsim:draft:circ:next?
calc:fsim:draft:circ1:add FILE,2
calc:fsim:draft:circ1:vna:ports 1
CALC:FSIM:DRAFt:CIRCuit1:EMBED:TYPE embed
calc:fsim:draft:circ1:file
"C:\Keysight\development\bheyburn\scripts\Python_SCPI\s2pFiles\CSET_FIXTURE_plus_5.00.s2p"
calc:fsim:draft:circ1:state 1
calc:fsim:draft:circ:next?
calc:fsim:draft:circ2:add FILE,2
calc:fsim:draft:circ2:vna:ports 3
CALC:FSIM:DRAFt:CIRCuit2:EMBED:TYPE embed
calc:fsim:draft:circ2:file
"C:\Keysight\development\bheyburn\scripts\Python_SCPI\s2pFiles\CSET_FIXTURE_minus_3.00.s2p"
calc:fsim:draft:circ2:state 1
CALC:FSIM:DRAFt:SECTion:CIRCuit:ENABle on
calc:fsim:apply
calc1:fsim:stat ON
calc1:fsim:stat?

Example program with legacy SCPI commands

*CLS
*OPC?
SYST:PRES
calc1:fsim:send:oord?
calc1:fsim:send:zcon:stat on
calc1:fsim:send:zcon:stat?
calc1:fsim:send:zcon:port1:z0 50
calc1:fsim:send:zcon:port1:z0?
calc1:fsim:send:zcon:port2:z0 100
calc1:fsim:send:zcon:port2:z0?
calc1:fsim:send:zcon:port3:z0 100
calc1:fsim:send:zcon:port3:z0?
calc1:fsim:send:zcon:port4:z0 50
calc1:fsim:send:zcon:port4:z0?
calc1:fsim:send:pmc:stat on
calc1:fsim:send:pmc:stat?
calc1:fsim:send:pmc:port1:type USER
calc1:fsim:send:pmc:port1:type?
calc1:fsim:send:pmc:port1:user:fil "C:\Keysight\development\bheyburn\scripts\Python_SCPI\s2pFiles\CSET_FIXTURE_plus_5.00.s2p"
calc1:fsim:send:pmc:port1:user:fil?
calc1:fsim:send:pmc:port3:type USER
calc1:fsim:send:pmc:port3:type?
calc1:fsim:send:pmc:port3:user:fil "C:\Keysight\development\bheyburn\scripts\Python_SCPI\s2pFiles\CSET_FIXTURE_minus_3.00.s2p"
calc1:fsim:send:pmc:port3:user:fil?
calc1:fsim:stat ON
calc1:fsim:stat?
Create fixturing function (2-port deembed, port extension, port matching, impedance conversion)

This example programs below create 2-port deembed, port extension, port matching and port impedance conversion function.

The first program uses Fsimulator Draft and Fsimulator Active SCPI commands while the second program uses legacy SCPI commands. Learn about Using Fixture Simulator.

See Other SCPI Example Programs

Example program with Fsimulator Draft and Fsimulator Active SCPI commands

```plaintext
*CLS
*OPC?
SYST:PRES
calc1:fsim:send:oord?
#calc1:fsim:send:oord 1,0,2,3
#Need to create blocks in this order:
#2-port deembed, port extension, port matching, arb z
calc:fsim:draft:circ:reset
calc:fsim:draft:circ:next?
calc:fsim:draft:circ1:add FILE,2
calc:fsim:draft:circ1:vna:ports 1
calc:fsim:draft:circ1:file
"C:\Keysight\development\bheyburn\scripts\Python_SCPI\s2pFiles\CSET_FIXTURE_plus_5.00.s2p"
calc:fsim:draft:circ1:state 1
calc:fsim:draft:circ:next?
calc:fsim:draft:circ2:add FILE,2
calc:fsim:draft:circ2:vna:ports 2
calc:fsim:draft:circ2:file
```
"C:\Keysight\development\bheyburn\scripts\Python_SCPI\s2pFiles\CSET_FIXTURE_minus_3.00.s2p"
calc:fsim:draft:circ2:state 1
calc:fsim:draft:circ:next?
calc:fsim:draft:circ3:add FILE,2
calc:fsim:draft:circ3:vna:ports 3
calc:fsim:draft:circ3:file
"C:\Keysight\development\bheyburn\scripts\Python_SCPI\s2pFiles\CSET_FIXTURE_plus_4.00.s2p"
calc:fsim:draft:circ3:state 1
calc:fsim:draft:circ:next?
calc:fsim:draft:circ4:add FILE,2
calc:fsim:draft:circ4:vna:ports 4
calc:fsim:draft:circ4:file
"C:\Keysight\development\bheyburn\scripts\Python_SCPI\s2pFiles\CSET_FIXTURE_minus_2.00.s2p"
calc:fsim:draft:circ4:state 1
CALC:FSIM:DRAFt:EXTension:PORT1:DELay 5E-11
CALC:FSIM:DRAFt:EXTension:PORT1:DELay?
CALC:FSIM:DRAFt:EXTension:PORT1:STATe on
CALC:FSIM:DRAFt:EXTension:PORT1:STATe?
CALC:FSIM:DRAFt:EXTension:PORT1:END
CALC:FSIM:DRAFt:EXTension:PORT2:DELay 5E-11
CALC:FSIM:DRAFt:EXTension:PORT2:DELay?
CALC:FSIM:DRAFt:EXTension:PORT2:STATe on
CALC:FSIM:DRAFt:EXTension:PORT2:STATe?
CALC:FSIM:DRAFt:EXTension:PORT2:END
CALC:FSIM:DRAFt:EXTension:PORT3:DELay 1E-10
CALC:FSIM:DRAFt:EXTension:PORT3:DELay?
CALC:FSIM:DRAFt:EXTension:PORT3:STATe on
CALC:FSIM:DRAFt:EXTension:PORT3:STATe?
CALC:FSIM:DRAFt:EXTension:PORT3:END
CALC:FSIM:DRAFT:EXTension:PORT4:DELay 1E-10
CALC:FSIM:DRAFT:EXTension:PORT4:DELay?
CALC:FSIM:DRAFT:EXTension:PORT4:STATe on
CALC:FSIM:DRAFT:EXTension:PORT4:STATe?
CALC:FSIM:DRAFT:EXTension:PORT4:END
calc:fsim:draft:circ:next?
calc:fsim:draft:circ5:add plpc,2
calc:fsim:draft:circ5:vna:ports 1
CALC:FSIM:DRAFT:CIRCuit5:EMBED:TYPE embed
calc1:fsim:draft:circ5:stat on
calc1:fsim:draft:circ5:stat?
calc1:fsim:draft:circ5:par:L 2.2E-08
calc1:fsim:draft:circ5:par:L?
calc1:fsim:draft:circ5:par:R 5.2
calc1:fsim:draft:circ5:par:R?
calc1:fsim:draft:circ5:par:C 0
calc1:fsim:draft:circ5:par:C?
calc1:fsim:draft:circ5:par:G 0
calc1:fsim:draft:circ5:par:G?
calc:fsim:draft:circ:next?
calc:fsim:draft:circ6:add plpc,2
CALC:FSIM:DRAFT:CIRCuit6:EMBED:TYPE embed
calc1:fsim:draft:circ6:stat on
calc1:fsim:draft:circ6:stat?
calc1:fsim:draft:circ6:par:L 2.2E-08
calc1:fsim:draft:circ6:par:L?
calc1:fsim:draft:circ6:par:R 5.2
calc1:fsim:draft:circ6:par:R?
calc1:fsim:draft:circ6:par:C 0
calc1:fsim:draft:circ6:par:C?
calc1:fsim:draft:circ6:par:G 0
calc1:fsim:draft:circ6:par:G?
calc1:fsim:draft:circ:next?
calc1:fsim:draft:circ7:add plpc,2
calc1:fsim:draft:circ7:vna:ports 3
CALC:FSIM:DRAFT:CIRCuit7:EMBED:TYPE embed
calc1:fsim:draft:circ7:stat on
calc1:fsim:draft:circ7:stat?
calc1:fsim:draft:circ7:par:L 3.4E-08
calc1:fsim:draft:circ7:par:L?
calc1:fsim:draft:circ7:par:R 32
calc1:fsim:draft:circ7:par:R?
calc1:fsim:draft:circ7:par:C 3.3E-11
calc1:fsim:draft:circ7:par:C?
calc1:fsim:draft:circ7:par:G 0.00207
calc1:fsim:draft:circ7:par:G?
calc1:fsim:draft:circ:next?
calc1:fsim:draft:circ8:add plpc,2
calc1:fsim:draft:circ8:vna:ports 4
CALC:FSIM:DRAFT:CIRCuit8:EMBED:TYPE embed
calc1:fsim:draft:circ8:stat on
calc1:fsim:draft:circ8:stat?
calc1:fsim:draft:circ8:par:L 3.4E-08
calc1:fsim:draft:circ8:par:L?
calc1:fsim:draft:circ8:par:R 32
calc1:fsim:draft:circ8:par:R?
calc1:fsim:draft:circ8:par:C 3.3E-11
calc1:fsim:draft:circ8:par:C?
calc1:fsim:draft:circ8:par:G 0.00207
calc1:fsim:draft:circ8:par:G?
CALC:FSIM:DRAFT:SECTION:ZCONversion:ENABLE on
calc1:fsim:draft:zcon:send:port1:state on
calc1:fsim:draft:zcon:send:port1:state?
calc1:fsim:draft:zcon:send:port1:complex 10,1
calc1:fsim:draft:zcon:send:port1:complex?
calc1:fsim:draft:zcon:send:port2:state on
calc1:fsim:draft:zcon:send:port2:state?
calc1:fsim:draft:zcon:send:port2:complex 20,2
calc1:fsim:draft:zcon:send:port2:complex?
calc1:fsim:draft:zcon:send:port3:state on
calc1:fsim:draft:zcon:send:port3:state?
calc1:fsim:draft:zcon:send:port3:complex?
calc1:fsim:draft:zcon:send:port4:state on
calc1:fsim:draft:zcon:send:port4:state?
calc1:fsim:draft:zcon:send:port4:complex 40,4
calc1:fsim:draft:zcon:send:port4:complex?
CALC:FSIM:DRAFT:SECTION:EXTension:ENABLE ON
calc:fsim:apply
calc1:fsim:stat ON
calc1:fsim:stat?
calc1:fsim:stat OFF
calc1:fsim:stat?
Example program with legacy SCPI commands

*CLS
*OPC?
SYST:PRES
calc1:fsim:send:oord?
calc1:fsim:send:oord 1,0,2,3
calc1:fsim:send:oord?
sens1:corr:ext ON
sens1:corr:ext?
sens1:corr:ext:port1:time 5E-11
sens1:corr:ext:port1:time?
sens1:corr:ext:port2:time 5E-11
sens1:corr:ext:port2:time?
sens1:corr:ext:port3:time 1E-10
sens1:corr:ext:port3:time?
sens1:corr:ext:port4:time 1E-10
sens1:corr:ext:port4:time?
sens1:corr:ext OFF
sens1:corr:ext ON
sens1:corr:ext?
calc1:fsim:send:deem:stat ON
calc1:fsim:send:deem:stat?
calc1:fsim:send:deem:port1:type user
calc1:fsim:send:deem:port1:type?
calc1:fsim:send:deem:port1:user:fil "C:\Keysight\development\bheyburn\scripts\Python_SCPI\s2pFiles\CSET_FIXTURE_plus_5.00.s2p"
calc1:fsim:send:deem:port1:user:fil?
calc1:fsim:send:deem:port2:type user
calc1:fsim:send:deem:port2:type?
calc1:fsim:send:deem:port2:user:fil
"C:\Keysight\development\bheyburn\scripts\Python_SCPI\s2pFiles\CSET_FIXTURE_minus_3.00.s2p"
calc1:fsim:send:deem:port2:user:fil?
calc1:fsim:send:deem:port3:type user
calc1:fsim:send:deem:port3:type?
calc1:fsim:send:deem:port3:user:fil
"C:\Keysight\development\bheyburn\scripts\Python_SCPI\s2pFiles\CSET_FIXTURE_plus_4.00.s2p"
calc1:fsim:send:deem:port3:user:fil?
calc1:fsim:send:deem:port4:type user
calc1:fsim:send:deem:port4:type?
calc1:fsim:send:deem:port4:user:fil
"C:\Keysight\development\bheyburn\scripts\Python_SCPI\s2pFiles\CSET_FIXTURE_minus_2.00.s2p"
calc1:fsim:send:deem:port4:user:fil?
calc1:fsim:send:zcon:stat on
calc1:fsim:send:zcon:stat?
calc1:fsim:send:zcon:port1:real 10
calc1:fsim:send:zcon:port1:real?
calc1:fsim:send:zcon:port1:imag 1
calc1:fsim:send:zcon:port1:imag?
calc1:fsim:send:zcon:port2:real 20
calc1:fsim:send:zcon:port2:real?
calc1:fsim:send:zcon:port2:imag 2
calc1:fsim:send:zcon:port2:imag?
calc1:fsim:send:zcon:port3:real 30
calc1:fsim:send:zcon:port3:real?
calc1:fsim:send:zcon:port3:imag 3
calc1:fsim:send:zcon:port3:imag?
calc1:fsim:send:zcon:port4:real 40
calc1:fsim:send:zcon:port4:real?
calc1:fsim:send:zcon:port4:imag 4
calc1:fsim:send:zcon:port4:imag?
calc1:fsim:send:pmc:stat on
calc1:fsim:send:pmc:stat?
calc1:fsim:send:pmc:port1:type plpc
calc1:fsim:send:pmc:port1:type?
calc1:fsim:send:pmc:port1:par:L 2.2E-08
calc1:fsim:send:pmc:port1:par:L?
calc1:fsim:send:pmc:port1:par:C 0
calc1:fsim:send:pmc:port1:par:C?
calc1:fsim:send:pmc:port1:par:G 0
calc1:fsim:send:pmc:port1:par:G?
calc1:fsim:send:pmc:port2:type plpc
calc1:fsim:send:pmc:port2:type?
calc1:fsim:send:pmc:port2:par:L 2.2E-08
calc1:fsim:send:pmc:port2:par:L?
calc1:fsim:send:pmc:port2:par:C 0
calc1:fsim:send:pmc:port2:par:C?
calc1:fsim:send:pmc:port2:par:G 0
calc1:fsim:send:pmc:port2:par:G?
calc1:fsim:send:pmc:port3:type?
calc1:fsim:send:pmc:port3:par:L 3.4E-08
calc1:fsim:send:pmc:port3:par:L?
calc1:fsim:send:pmc:port3:par:R 32
(calc1:fsim:send:pmc:port3:par:R?)
calc1:fsim:send:pmc:port3:par:C 3.3E-11
calc1:fsim:send:pmc:port3:par:C?
calc1:fsim:send:pmc:port3:par:G 0.00207
(calc1:fsim:send:pmc:port3:par:G?)
calc1:fsim:send:pmc:port4:type plpc
(calc1:fsim:send:pmc:port4:type?)
calc1:fsim:send:pmc:port4:par:L 3.4E-08
(calc1:fsim:send:pmc:port4:par:L?)
calc1:fsim:send:pmc:port4:par:R 32
(calc1:fsim:send:pmc:port4:par:R?)
calc1:fsim:send:pmc:port4:par:C 3.3E-11
(calc1:fsim:send:pmc:port4:par:C?)
calc1:fsim:send:pmc:port4:par:G 0.00207
(calc1:fsim:send:pmc:port4:par:G?)
calc1:fsim:stat ON
calc1:fsim:stat?
calc1:fsim:stat OFF
calc1:fsim:stat?
Create fixturing function (port impedance conversion and port extension)

This example programs below create port extension and port impedance conversion function.

The first program uses Fsimulator Draft and Fsimulator Active SCPI commands while the second program uses legacy SCPI commands. Learn about Using Fixture Simulator.

See Other SCPI Example Programs

Example program with Fsimulator Draft and Fsimulator Active SCPI commands

```
SYST:PRES
*OPC?
#calc1:fsim:send:oord 3,2,1,0
#The command above indicates that we need to create the blocks in this order:
#Arb Impendance, Port Matching, 2-Port Deembed, Port Extensions
#Special note: It turns out that the arb impedance doesn't matter where it's at in the chain, the new fixture generator
#takes care of it appropriately, so it doesn't matter (and no need to have a command to move it) (it is in a different processing chain)
#for as an example, we'll show you how to move it:
calc1:fsim:stat ON
calc1:fsim:stat?
calc:fsim:draft:discard
CALC:FSIM:DRAFT:SECTION:ZCONversion:ENABLE on
calc1:fsim:draft:zcon:send:port1:state on
calc1:fsim:draft:zcon:send:port1:state?
calc1:fsim:draft:zcon:send:port1:scal 75
calc1:fsim:draft:zcon:send:port1:scal?
CALC:FSIM:DRAFT:EXTension:PORT1:DELay 1E-10
CALC:FSIM:DRAFT:EXTension:PORT1:DELay?
```
Example program with legacy SCPI commands

```
SYST:PRES
*OPC?
calc1:fsim:send:oord?
calc1:fsim:send:oord 3,2,1,0
calc1:fsim:stat ON
calc1:fsim:stat?
sens1:corr:ext:port1:time 1E-10
sens1:corr:ext:port1:time?
sens1:corr:ext ON
sens1:corr:ext?
sens1:corr:ext

calc1:fsim:send:zcon:stat on
calc1:fsim:send:zcon:stat?
calc1:fsim:send:zcon:port1:z0 75
```
Getting and Putting Data using SCPI

This Visual Basic Program does the following:

- Reads data from the analyzer
- Puts the data back into memory
- To see the data on the analyzer after running the program, from the front panel click:
  \[ \text{Trace - Math/Memory - Memory Trace} \]

To run this program, you need:

- An established GPIB interface connection

See Other SCPI Example Programs

**Note:** To change the read and write location of data, removing the comment from the beginning of ONE of the lines, and replace the comment in the beginning of the SDATA and SMEM lines.

```vbnet
Private Sub ReadWrite_Click()
    Dim i As Integer
    Dim t As Integer
    Dim q As Integer
    Dim dat As String
    Dim cmd As String
    Dim datum() As Double

    GPIB.Configure
    GPIB.Write "SYSTem:PRESet;*wai"

    'Select the measurement
    GPIB.Write "CALCulate:PARameter:SELect 'CH1_S11_1'"

    'Read the number of data points
    GPIB.Write "SENSe1:SWEep:POIN?"
    numpts = GPIB.Read

    'Turn continuous sweep off
    GPIB.Write "INITiate:CONTinuous OFF"

    'Take a sweep
    GPIB.Write "INITiate:IMMediate;*wai"

    'Ask for the Data

    'PICK ONE OF THESE LOCATIONS TO READ
    'GPIB.Write "CALCulate:DATA? FDATA" 'Formatted Meas
    'GPIB.Write "CALCulate:DATA? FMEM" 'Formatted Memory
    GPIB.Write "CALCulate:DATA? SDATA" 'Corrected, Complex Meas
```
This Excel VBA Program with VISA-COM does the following:

- Reads data from the analyzer
- Puts the data back into memory

**Note:** To change the read and write location of data, removing the comment from the beginning of ONE of the lines, and replace the comment in the beginning of the FDATA lines.
Sub SampleGetPutData()
    '*** The variables of the resource manager and the instrument I/O are declared.
    Dim ioMgr As VisaComLib.ResourceManager
    Dim GPIB As VisaComLib.FormattedIO488
    '*** The memory area of the resource manager and the instrument I/O are acquired.
    Set ioMgr = New VisaComLib.ResourceManager
    Set GPIB = New VisaComLib.FormattedIO488
    '*** Open the instrument.
    Set GPIB.IO = ioMgr.Open("GPIB0::16::INSTR")
    GPIB.IO.timeout = 10000

    Dim Numpts As Long
    Dim Datam As Variant

    'Select the measurement
    GPIB.WriteString "CALCulate1:MEASure1:PARameter 'S21'", True
    'Read the number of data points
    GPIB.WriteString "SENSe1:SWEep:POINts?", True
    Numpts = GPIB.ReadNumber
    'Turn continuous sweep off
    GPIB.WriteString "INITiate:CONTinuous OFF", True
    'Take a sweep
    GPIB.WriteString "INITiate1:IMMediate;*WAI", True
    'Ask for the Data
    'PICK ONE OF THESE LOCATIONS TO READ
    GPIB.WriteString "CALCulate1:MEASure1:DATA:FDATA?", True
        ' Formatted Meas
    GPIB.WriteString "CALCulate1:MEASure1:DATA:FMEM?", True
        ' Formatted Memory
    GPIB.WriteString "CALCulate1:MEASure1:DATA:SDATA?", True
        ' Corrected, Complex Meas
    GPIB.WriteString "CALCulate1:MEASure1:DATA:SMEM?", True
        ' Corrected, Complex Memory
    GPIB.WriteString "SENSe1:CORRection:CSET:ETERm:DATA 'Directivity(1,1)'", True
        ' Error-Term Directivity

    'Parse the data
    Datam = GPIB.ReadList(ASCIIType_R8, ",")

    'PUT THE DATA BACK IN
    GPIB.WriteString "CALCulate1:MEASure1:DATA:FDATA ", False
        ' Formatted Meas
    GPIB.WriteString "CALCulate1:MEASure1:DATA:FMEM ", False
        ' Formatted Memory
    GPIB.WriteString "CALCulate1:MEASure1:DATA:SDATA ", False
        ' Corrected, Complex Meas
    GPIB.WriteString "CALCulate1:MEASure1:DATA:SMEM ", False
        ' Corrected, Complex Memory
    GPIB.WriteString "SENSe1:CORRection:CSET:ETERm:DATA 'Directivity(1,1)',", False
        ' Error-Term Directivity
GPIB.WriteList Datam, ASCIIType_R8, ",", True

'*** End procedure
GPIB.IO.Close
End Sub
GPIB Pass-Through Example

The SCPI SYSTem commands used in this example allow you to send GPIB commands to another GPIB device through the VNA. The other device would typically be connected to the VNA through the System Controller GPIB port on the VNA rear-panel or alternatively be connected using a USB/GPIB interface. Uncomment the line in Blue text in the example to open a session for a USB/GPIB interface.

This VB Script example uses the COM SCPIStringParser object. However, this is not critical to the use of these commands; they can be sent using the normal syntax of your programming environment. Using the SCPIStringParser over LAN allows you to communicate with GPIB devices without requiring your remote PC to have a GPIB interface card installed.

Although this method of pass-through works for most applications, there are a couple of limitations:

- All data is transferred using ASCII format. Therefore, transferring large blocks of data is very slow.
- Only read and write functions are possible. Service Interrupts are not supported.

See Other SCPI Example Programs

```vbscript
option explicit
dim app
set app = CreateObject("AgilentPNA83x.Application")

dim p
set p = app.ScpiStringParser

' Open a new GPIB session on Bus:0 Device:14 Timeout: 100ms
p.Parse "SYST:COMM:GPIB:RDEV:OPEN 0,14,100"
' The following commented-out line shows opening the same session but
' for a USB/GPIB interface with VISA interface number GPIB4
'p.Parse "SYST:COMM:GPIB:RDEV:OPEN 4,14,100"

dim handleAsStr

' Retrieve the handle (ID number)
handleAsStr = p.Parse("SYST:COMM:GPIB:RDEV:OPEN?"

' Convert the handle to an integer
dim handleAsInt
handleAsInt = CInt(handleAsStr)

' Send the "*IDN?" query
p.Parse "SYST:COMM:GPIB:RDEV:WRITE " & handleAsInt & ","*IDN?"

' Read its results
dim idn
```
msbbox idn

' Close the GPIB session
p.Parse "SYST:COMM:GPIB:RDEV:CLOSE " & handleAsInt
This example assumes the user's PC has a National Instruments GPIB board. The example is comprised of three basic parts:

1. Initialization
2. Main Body
3. Cleanup

The Initialization portion consists of getting a handle to the VNA and then doing a GPIB clear of the VNA.

The Main Body consists of the VNA SCPI example.

The last step, Cleanup, releases the VNA for front panel control.

#include <stdio.h>
#include <stdlib.h>

#include <windows.h>
#include "decl-32.h"

#define ERRMSGSIZE 1024 // Maximum size of SCPI command string
#define ARRAYSIZE 1024 // Size of read buffer
#define BDINDEX 0 // Board Index of GPIB board
#define PRIMARY_ADDR_OF_PNA 16 // GPIB address of VNA
#define NO_SECONDARY_ADDR 0 // VNA has no Secondary address
#define TIMEOUT T10s // Timeout value = 10 seconds
#define EOTMODE 1 // Enable the END message
#define EOSMODE 0 // Disable the EOS mode

int pna;
char ValueStr[ARRAYSIZE + 1];

void GPIBWrite(char* SCPIcmd);
char *GPIBRead(void);
void GPIBCleanup(int Dev, char* ErrorMsg);

int main()
{

char *opc;
char *result;
char *value;

/*
* INITIALIZATION SECTION
* ----------------------------------------
*/

/*
* The application brings the VNA online using ibdev. A device handle,VNA, is
* returned and is used in all subsequent calls to the VNA.
*/
pna = ibdev(BDINDEX, PRIMARY_ADDR_OF_PNA, NO_SECONDARY_ADDR,
TIMEOUT, EOTMODE, EOSMODE);
if (ibsta & ERR)
{
printf("Unable to open handle to PNA/nibsta = 0x%x iberr = %d/n",
ibsta, iberr);
return 1;
}

/*
* Do a GPIB Clear of the VNA. If the error bit ERR is set in ibsta, call
* GPIBCleanup with an error message.
*/
ibclr (pna);
if (ibsta & ERR)
{
GPIBCleanup(pna, "Unable to perform GPIB clear of the PNA");
return 1;
}

/*
* MAIN BODY SECTION
* ----------------------------------------
*/

// Reset the analyzer to instrument preset
GPIBWrite("SYSTem:FPRESET");

// Create S11 measurement
GPIBWrite("CALCulate1:PARameter:DEFine:EXT 'My_S11',S11");
// Turn on Window #1
GPIBWrite("DISPlay:WINDow1:STATe ON");

// Put a trace (Trace #1) into Window #1 and 'feed' it from the measurement
GPIBWrite("DISPlay:WINDow1:TRACe1:FEED 'My_S11'");

// Setup the channel for single sweep trigger
GPIBWrite("INITiate1:CONTinuous OFF;/*OPC?");
opc = GPIBRead();
GPIBWrite("SENSe1:SWEep:TRIGger:POINt OFF");

// Set channel parameters
GPIBWrite("SENSe1:SWEep:POINts 11");
GPIBWrite("SENSe1:FREQuency:STARt 1000000000");
GPIBWrite("SENSe1:FREQuency:STOP 2000000000");

// Send a trigger to initiate a single sweep
GPIBWrite("INITiate1;*OPC?");
opc = GPIBRead();

// Must select the measurement before we can read the data
GPIBWrite("CALCulate1:PARameter:SELect 'My_S11'");

// Read the measurement data into the "result" string variable
GPIBWrite("FORMat ASCII");
GPIBWrite("CALCulate1:DATA? FDATA");
result = GPIBRead();

// Print the data to the display console window
printf("S11(dB) - Visual C++ SCPI Example for PNA/n/n");
value = strtok(result, ",");
while (value != NULL)
{
    printf("%s/n", value);
    value = strtok(NULL, ",");
}

/*
* =========================================
* CLEANUP SECTION
* =========================================
*/

/* The VNA is returned to front panel control. */
ibon1(pna, 0);
return 0;
}

/*
* Write to the VNA
void GPIBWrite(char* SCPIcmd)
{
    int length;
    char ErrorMsg[ERRMSGSIZE + 1];
    length = strlen(SCPIcmd) ;

    ibwrt (pna, SCPIcmd, length);
    if (ibsta & ERR)
    {
        strcpy(ErrorMsg, "Unable to write this command to PNA:/n");
        strcat(ErrorMsg, SCPIcmd);
        GPIBCleanup(pna, ErrorMsg);
        exit(1);
    }
}

char* GPIBRead(void)
{
    ibrd (pna, ValueStr, ARRAYSIZE); 
    if (ibsta & ERR)
    {
        GPIBCleanup(pna, "Unable to read from the PNA");
        exit(1);
    }
    else
    return ValueStr;
}

void GPIBCleanup(int Dev, char* ErrorMsg)
{
    printf("Error : %s/nibsta = 0x%x iberr = %d (%s)/n", 
    ErrorMsg, ibsta, iberr, ErrorMnemonic[iberr]);
    if (Dev != -1)
    {
        printf("Cleanup: Returning PNA to front panel control/n");
        ibonl (Dev, 0);
    }
}
Perform a Guided 2-Port or 4-Port Cal using SCPI

This example performs a Guided 2-Port or 4-port Calibration using ONE set of calibration standards or an ECAL module.

A measurement must first be set up with desired frequency range, power, and so forth, ready to be calibrated.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as *.vbs.

Learn how to setup and run the macro.

See Guided Cal SCPI commands

See Other SCPI Example Programs

```vbs
Dim app
Dim scpi
' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' To perform 2-port cal, Uncomment TwoPortGuidedCal()
' Then comment FourPortGuidedCal()

'Do 2-port Cal
'TwoPortGuidedCal()

'Do 4-port Cal
FourPortGuidedCal

Sub TwoPortGuidedCal()
' Select the connectors
scpi.Execute("sens:corr:coll:guid:conn:port1 ""APC 3.5 female""
scpi.Execute("sens:corr:coll:guid:conn:port2 ""APC 3.5 male"
scpi.Execute("sens:corr:coll:guid:conn:port3 ""Not used"
scpi.Execute("sens:corr:coll:guid:conn:port4 ""Not used"
MsgBox("Connectors defined for Ports 1 and 2")

' Select the Cal Kit for each port being calibrated.
scpi.Execute("sens:corr:coll:guid:ckit:port1 ""85052D"
```

5298

' To use an ECal module instead, comment out the above two lines
' and uncomment the appropriate lines below:
' Your ECal module must already be connected
' via USB to the VNA.
'shci.Parse "sens:corr:coll:guid:ckit:port1 'N4691-60004 ECal'"

' Non-factory characterizations are specified as follows:
' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
'
MsgBox("Cal kits defined for Ports 1 and 2")

' Initiate the calibration and query the number of steps
numSteps = GenerateSteps()
' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

Sub FourPortGuidedCal()
' Select the connectors
scpi.Execute("sens:corr:coll:guid:conn:port1 ""APC 3.5 female"" ")
scpi.Execute("sens:corr:coll:guid:conn:port2 ""APC 3.5 female"" ")
scpi.Execute("sens:corr:coll:guid:conn:port3 ""APC 3.5 female"" ")
scpi.Execute("sens:corr:coll:guid:conn:port4 ""APC 3.5 female"" ")
MsgBox("Connectors defined for Ports 1 to 4")
' Select the Cal Kit for each port being calibrated.
scpi.Execute("sens:corr:coll:guid:ckit:port1 ""85052D"" ")
scpi.Execute("sens:corr:coll:guid:ckit:port3 ""85052D"" ")
' To use an ECal module instead, comment out the above four lines
' and uncomment these four lines and use the part number printed
' on your module (which in our case was N4431-60003), followed
' by the word 'ECal'. Your ECal module must already be connected
' via USB to the VNA.

' see above for ECal options

MsgBox("Cal kits defined for Ports 1 to 4")

' Initiate the calibration and query the number of steps
numSteps = GenerateSteps()
' If your selected cal kit is not a 4-port ECal module which can
' mate to all 4 ports at once, then you may want to choose which
' thru connections to measure for the cal. You must measure at
' least 3 different thru paths for a 4-port cal (for greatest
' accuracy you can choose to measure a thru connection for all 6
' pairings of the 4 ports). If you omit this command, the default
' is to measure from port 1 to port 2, port 1 to port 3, and
' port 1 to port 4. For this example we select to measure
' from port 1 to port 2, port 2 to port 3, and port 2 to port 4.
scpi.Execute("sens:corr:coll:guid:thru:ports 1,2,2,3,2,4")
' Re-generate the connection steps to account for the thru changes
numSteps = GenerateSteps()
' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

Function GenerateSteps()
' Initiate the calibration and query the number of steps
scpi.Execute("sens:corr:coll:guid:init")
GenerateSteps =  scpi.Execute("sens:corr:coll:guid:steps?")
End Function

Sub MeasureAndComplete(numSteps)
MsgBox("Number of steps is " + CStr(numSteps))
' Measure the standards
For i = 1 To numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
MsgBox strPrompt, vbOKOnly, step
' Note: if you have set up a slow sweep speed (for example, if
' you're using a narrow IF bandwidth) or you're using ECal, and
' while a cal step is being measured you wish to have your program
' perform other operations (like checking for the click event of a
' Cancel button) and you're NOT using the COM ScpiStringParser,
' you can use the optional ASYNchronous argument with the ACQuire
' command as shown in this commented-out line below. The SCPI
' parser then will return immediately while the cal step measurement
' proceeds (i.e., the parser will NOT block-and-wait for the
' measurement step to finish, so you can send additional commands
' in the meantime). So you can do "*ESR?" or "*STB?" queries to
' monitor the status register bytes to see when the OPC bit gets set,
' which indicates the cal measurement step has finished. This OPC
' detection works for all of the VNA's SCPI parsers except the COM
' ScpiStringParser.
' "sens:corr:coll:guid:acq STAN" + CStr(i) + ",ASYN:*OPC"
scpi.Execute("sens:corr:coll:guid:acq STAN" + CStr(i))
Next
' Conclude the calibration
MsgBox ("Cal is done!")
End Sub
Load Error Terms during a Cal Sequence

This example requires that you already have a Cal Set named "foo" that contains a 1-port cal on port 1 and a 1-port cal on port 2.

This example starts a Guided Calibration specifying an Unknown Thru. It loads the 1-port Cals from the existing "foo" Cal Set, then recalculates the number of steps required to complete the cal. After loading the 1-port cals, only the Unknown Thru standard is left to acquire.

```
SENS:CORR:COLL:GUID:CONN:PORT1 "APC 3.5 female"
SENS:CORR:COLL:GUID:CONN:PORT2 "APC 3.5 female"
SENS:CORR:COLL:GUID:CKIT:PORT1 "85033D/E"
SENS:CORR:COLL:GUID:CKIT:PORT2 "85033D/E"
SENS:CORR:COLL:GUID:METH UNKN
' auto-create user calsets for SCPI
SENS:CORR:PREF:CSET:SAVU 1
SENS:CORR:COLL:GUID:INIT
' should return the number 7
SENS:CORR:COLL:GUID:STEPS?
' to port 1, from port 1 in calset
SENS:CORR:COLL:GUID:ETER:LOAD "foo",1,1
' to port 2, from port 2 in calset
SENS:CORR:COLL:GUID:ETER:LOAD "foo",2,2
' should now return the number 1
SENS:CORR:COLL:GUID:STEPS?
' measure the unknown thru
SENS:CORR:COLL:GUID:ACQ STAN1
' save the cal to new user calset
SENS:CORR:COLL:GUID:SAVE
```
Modify a Calibration Kit using SCPI

This Visual Basic program:

- Modifies Calibration kit number 3
- Completely defines standard #4 (thru)

To run this program, you need:

- An established GPIB interface connection

See Other SCPI Example Programs

```vbnet
'Modifying cal kit number 3
Calkitnum = 3

'Designate the kit selection to be used for performing cal's
GPIB.Write "SENSe:CORRection:COLLect:CKIT:SELect " & Val(Calkitnum)

'Reset to factory default values.
GPIB.Write "SENSe:CORRection:COLLect:CKIT:RESet " & Val(Calkitnum)

'Name this kit with your own name
GPIB.Write "SENSe:CORRection:COLLect:CKIT:NAME 'My Cal Kit'

'Assign standard numbers to calibration classes
'Set Port 1, class 1 (S11A) to be standard #8
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer1 8"
'Set Port 1, class 2 (S11B) to be standard #7
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer2 7"
'Set Port 1, class 3 (S11C) to be standard #3
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer3 3"
'Set Port 1, class 4 (S21T) to be standard #4
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer4 4"
'Set Port 2, class 1 (S22A) to be standard #8
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer5 8"
'Set Port 2, class 2 (S22B) to be standard #7
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer6 7"
'Set Port 2, class 3 (S22C) to be standard #3
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer7 3"
'Set Port 2, class 4 (S12T) to be standard #4
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer8 4"

'Set up Standard #4 completely
'Select Standard #4; the rest of the commands act on it
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard 4"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:FMIN 300KHz"
```
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:FMAX 9GHz"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:IMPedance 50"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:DELay 1.234 ns"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:LOSS 23e6"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:C0 0"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:C1 1"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:C2 2"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:C3 3"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:L0 10"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:L1 11"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:L2 12"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:L3 13"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:LABel 'My Special Thru'"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:TYPE THRU"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:CHARacteristic Coax"
Measurement Setup

The following C# example demonstrates how to set up a modulation distortion measurement.

See the Modulation Distortion commands.

See Other SCPI Example Programs

```csharp
using System;
using System.Collections.Generic;
using System.IO;
using System.Numerics;
namespace mod_tests
{
    partial class Vna
    {
        [TestMethod]
        public void Case01()
        {
            // Test Conditions:
            // - External source is named as "MXG";
            // - On PNA, file "D:\Symphony\FR1_100M_64QAM.mdx" is used
            // - PNA Port 1 connects to DUT Input, PNA Port 2 connects Output;
            _fmtIO.Cmd("syst:fpr");
            // add traces
            _fmtIO.Cmd("disp:wind1:stat on");
        }
    }
}
```
string[] trNames = new string[] { "PIn1", "POut2", "MDist2" };
int idx = 1;
foreach (var n in trNames)
{
    _fmtIO.Cmd(string.Format("calc:cust:deff ch1_{0}"", n));
    _fmtIO.Cmd(string.Format("disp:wind1:trac{0}:feed ch1_{1}"", idx, n));
    ++idx;
}

// hold sweep
_fmtIO.Cmd("sens:swe:mode hold");
string str;

// Setup Sweep
_fmtIO.Cmd("sens:DISTortion:SWEep:CARRier:FREQuency 1e9");
_fmtIO.Cmd("sens:freq:cent 1e9");
_fmtIO.Cmd("sens:freq:span 400e6");
_fmtIO.Cmd("sens:DISTortion:SWEep:POWer:CARRier:LEVEL:PORT DOUT2");
_fmtIO.Cmd("sens:DISTortion:SWEep:POWer:SPARam:LEVEL -25");
str = _fmtIO.CmdQry("sens:freq:cent?");
str = _fmtIO.CmdQry("sens:freq:span?");
str = _fmtIO.CmdQry("sens:DISTortion:SWEep:CARRier:FREQuency?");
str = _fmtIO.CmdQry("sens:DISTortion:SWEep:POWer:CARRier:LEVEL?");
str = _fmtIO.CmdQry("sens:DISTortion:SWEep:POWer:SPARam:LEVEL?");

// RF Path
_fmtIO.Cmd("sens:dist:path:dut:inp 1");
_fmtIO.Cmd("sens:dist:path:dut:outp 4");
str = _fmtIO.CmdQry("sens:dist:path:dut:inp?");
str = _fmtIO.CmdQry("sens:dist:path:dut:outp?");
str = _fmtIO.CmdQry("sens:dist:path:dut:nom:gain?");
str = _fmtIO.CmdQry("sens:dist:path:dut:nom:nf?");
_fmtIO.Cmd("sens:dist:path:sour:nom:ampl 10");
_fmtIO.Cmd("sens:dist:path:dut:nom:gain 10");
_fmtIO.Cmd("sens:dist:path:dut:nom:nf 3");
str = _fmtIO.CmdQry("sens:dist:path:sour:nom:ampl?");
str = _fmtIO.CmdQry("sens:dist:path:dut:nom:gain?");
str = _fmtIO.CmdQry("sens:dist:path:dut:nom:nf?");
_fmtIO.Cmd("sens:dist:path:sour:nom:ampl 0");
_fmtIO.Cmd("sens:dist:path:dut:nom:gain 0");
_fmtIO.Cmd("sens:dist:path:dut:nom:nf 0");
// Modulate
_fmtIO.Cmd("SYST:CONF:EDEV:STAT 'MXG',ON");
_fmtIO.Cmd("sens:DISTortion:MODulate:SOURce 'MXG'");
_fmtIO.Cmd("SOURce:MODulation:LOAD 'D:\Symphony\FR1_100M_\
_fmtIO.Cmd("SOURce:MODulation:STATe 1");
str = _fmtIO.CmdQry("sens:DISTortion:MODulate:SOURce?");
str = _fmtIO.CmdQry("SOURce:MODulation:FILE?");
str = _fmtIO.CmdQry("SOURce:MODulation:STATe?");
// Measure
str = _fmtIO.CmdQry("sens:DISTortion:MEASure:BAND1:NAME?");
str = _fmtIO.CmdQry("sens:DISTortion:MEASure:BAND1:TYPE?");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND1:NAME 'TEST1'");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND1:TYPE NPR");
str = _fmtIO.CmdQry("sens:DISTortion:MEASure:BAND1:NOTCh:OFFS?");
str = _fmtIO.CmdQry("sens:DISTortion:MEASure:BAND1:NOTCh:IBW?");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND1:NOTCh:OFFS 20e6");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND1:NOTCh:IBW 10e6");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND:ADD");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND2:NAME 'TEST2'");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND2:TYPE ACP");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND2:ACP:LOW:IBW 90e6");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND1:DEL");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND:INIT");
str = _fmtIO.CmdQry("sens:DISTortion:MEASure:BAND:CARR:OFFS?");
str = _fmtIO.CmdQry("sens:DISTortion:MEASure:BAND:CARR:IBW?");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND:CARR:OFFS 0");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND:CARR:IBW 90e6");
str = _fmtIO.CmdQry("sens:DISTortion:MEASure:BAND:COUNT?");
// do a full sweep (noise, s21 and measurement)
_fmtIO.Cmd("sens:swe:mode single");
_fmtIO.CmdQry("*OPC?", TimeOuts._2min);

// display scale
_fmtIO.Cmd("disp:wind1:trac:y:scal:coup on");
_fmtIO.Cmd("disp:wind1:y:auto");

// change Test Receiver Attenuation
_fmtIO.Cmd("sour:pow4:att:rec:test 10");
_fmtIO.Cmd("sens:swe:mode single");
_fmtIO.CmdQry("*OPC?", TimeOuts._2min);

// change DUT Analysis BW
str = _fmtIO.CmdQry("sens:dist:meas:corr:aper:auto?");  
str = _fmtIO.CmdQry("sens:dist:meas:corr:aper?");  
_fmtIO.Cmd("sens:dist:meas:corr:aper 20e6");  
str = _fmtIO.CmdQry("sens:dist:meas:corr:aper:auto?");  
str = _fmtIO.CmdQry("sens:dist:meas:corr:aper?");  
_fmtIO.Cmd("sens:swe:mode single");
_fmtIO.CmdQry("*OPC?", TimeOuts._2min);
_fmtIO.Cmd("sens:dist:meas:corr:aper:auto 1");

// change ADC Filter Type
str = _fmtIO.CmdQry("sens:dist:adc:filt:type?");  
_fmtIO.Cmd("sens:dist:adc:filt:type NARR");  
_fmtIO.Cmd("sens:swe:mode single");  
_fmtIO.CmdQry("*OPC?", TimeOuts._2min);
_fmtIO.Cmd("sens:dist:adc:filt:type AUTO");

// re-use linear sweeps, change power and do measurement s
str = _fmtIO.CmdQry("sens:dist:swe:lin:reus?");
_fmtIO.Cmd("sens:dist:swe:lin:reus 1");
int pwr = -30; // power start
while (pwr <= -10) // power stop
{
    _fmtIO.Cmd(string.Format("sens:DISTortion:SWEep:POWER:{0}", pwr));
    _fmtIO.Cmd("sens:swe:mode single");
    _fmtIO.CmdQry("*OPC?", TimeOuts._2min);
    pwr += 5; // power step
}
_fmtIO.Cmd("sens:dist:swe:lin:reus 1");
}
Measurement Setup Converters

The following C# example demonstrates how to set up a modulation distortion converter measurement.

See the Modulation Distortion and Mixer commands.

```csharp
def example01():
    pna = VNA(pna_addr)
    pna.write("syst:fpr")
    pna.write("disp:wind1:stat on")
    pna.write("calc:cust:def 'ch1_MGain21','Modulation Distortion Converters','MGain21'")
    pna.write("disp:wind1:trac1:feed 'ch1_MGain21'")
    pna.write("sens:swe:mode hold")

    freq_in_cent = 3.08e9
    freq_in_span = 300e6
    freq_in_star = freq_in_cent - freq_in_span/2
    freq_in_stop = freq_in_cent + freq_in_span/2
    # Mixer frequencies
    pna.write("SENS:MIX:INPut:FREQ:MODE SWEPt")  # Mixer settings
    pna.write("SENS:MIX:INPut:FREQ:STAR {}").format(freq_in_star)
    pna.write("SENS:MIX:INPut:FREQ:STOP {}").format(freq_in_stop)
    pna.write("SENS:MIX:LO:FREQ:MODE FIXED")
    pna.write("SENS:MIX:LO:FREQ:FIX 2.2e9")
    pna.write("SENS:MIX:LO:POW 9")
    pna.write("SENS:MIX:OUTP:FREQ:SID LOW")
    pna.write("SENS:MIX:CALC Output")
    pna.write("SENS:MIX:APPLY")
    pna.write("SENS:MIX:LO:NAME 'Port 3'")  # First apply the settings, then set LO Name
    pna.write("SENS:MIX:APPLY")
    # SA frequencies
    pna.write("sens:dist:swe:carr:freq {}").format(freq_in_cent)
    pna.write("sens:freq:cent {}").format(freq_in_cent)
    pna.write("sens:freq:span {}").format(freq_in_span)
    # modulate
    pna.write("syst:conf:edev:stat 'myMXG',ON")
    pna.write("sens:dist:mod:sour 'myMXG'")
    pna.write(r"sour:mod:load 'D:\mod\flat\flat_1001.mdx'")
    pna.write("sour:mod:stat 1")
```
#
pna.write("sens:swe:mode cont")
# close
pna.close()
Create Modulation File

This example program creates a source modulation file for modulation distortion measurements.

This VBScript program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as ModFile.vbs. Learn how to setup and run the macro.

See the Modulation Distortion commands.

See Other SCPI Example Programs

'Example code to create/edit modulation signals
set app = CreateObject("Agilentpna835x.application", "k-n5242b-81474")
set scpi = app.ScpiStringParser
' Preset and add a trace in Modulation Distortion channel
scpi.Parse "syst:fpr"
scpi.Parse("disp:wind1:stat on")
scpi.Parse("calc:cust:def 'ch1_PIn1','Modulation Distortion','PIn1'")
scpi.Parse "disp:wind1:trac1:feed 'ch1_PIn1'"
' Create an NPR Notch signal
scpi.Parse "sour:mod:file:type nprn"
scpi.Parse "sour:mod:file:sign:srat 200e6"
scpi.Parse "sour:mod:file:sign:span 100e6"
scpi.Parse "sour:mod:file:sign:span:pri 1"
scpi.parse "sour:mod:file:sign:tone:numb 1001"
scpi.Parse "sour:mod:file:sign:npr:notcl:span 10e6"
scpi.Parse "sour:mod:file:sign:npr:notcl:offs 0"
scpi.parse "sour:mod:file:save 'd:\symphony\scpi\npr.mdx'
' Create compact signals
' Number of tones = 101/1001/10001
' Peak-to-avg priority: on/off
' Number of files = 5
' Original signal file:
5GNR_256QAM_120kHz_SCS_100MHz_122p88MHzSR.wfm
scpi.Parse "sour:mod:file:type comp"
infile = "5GNR_256QAM_120kHz_SCS_100MHz_122p88MHzSR"
scpi.parse "sour:mod:file:sign:comp:ofile 'd:\Symphony\scpi" & infile & ".wfm"
for e = 2 to 4 '101/1001/10001
numTones = 10 ^ e + 1
for i = 1 to 5
scpi.parse "sour:mod:file:sign:comp:pavg:pri 0"
outfile = "d:\Symphony\scpi" & infile & "." & numTones & "." & i
scpi.parse "sour:mod:file:save '" & outfile & ".mdx"
scpi.parse "sour:mod:file:sign:comp:pavg:pri 1"
outfile = outfile & "p"
scpi.parse "sour:mod:file:save '" & outfile & ".mdx"
next
next

' Create a flat tone signal
scpi.parse "sour:mod:file:type flat"
scpi.parse "sour:mod:file:sign:srat 200e6"
scpi.parse "sour:mod:file:sign:span 100e6"
scpi.parse "sour:mod:file:sign:span:pri 1"
scpi.parse "sour:mod:file:sign:tone:numb 1001"
scpi.parse "sour:mod:file:save 'd:\symphony\scpi\flat.mdx'"

' Edit a signal
scpi.parse "sour:mod:file:load 'd:\symphony\scpi\npr.mdx'"
scpi.parse "sour:mod:file:sign:npr:notc1:offs 10e6"
scpi.parse "sour:mod:file:save 'd:\symphony\scpi\npr_2001tone_notch_offset_10M.mdx'"
msgbox "Done"
Display Data Setup

The following C# example demonstrates how to read display data, tone-by-tone data, distortion table data, and how to add/remove display columns in the distortion table.

See the Modulation Distortion commands.

See Other SCPI Example Programs

```csharp
using System;
using System.Collections.Generic;
using System.IO;
using System.Numerics;

namespace mod_tests
{
    partial class Vna
    {
        [TestMethod]
        public void Case02()
        {
            // Test Conditions:
            // - External source is named as "MXG";
            // - On PNA, file "D:\Symphony\FR1_100M_64QAM.mdx" is used
            // - PNA Port 1 connects to DUT Input, PNA Port 2 connects to DUT Output;
            _fmtIO.Cmd("syst:fpr");
            // add traces
```
string[] trNames = new string[] { "POut2", "PIn1", "MSig2", "MDist2", "MGain21", "MComp21", "PGain21", "S11", "S21", "LPIn1", "LPOut1", "LPOut2" };

int idx = 1;

foreach (var n in trNames)
{
    _fmtIO.Cmd(string.Format("calc:cust:def 'ch1_{0}','Modulation Distortion','{0}'", n));
    _fmtIO.Cmd(string.Format("disp:wind1:trac{0}:feed 'ch1_{1}'", idx, n));
    ++idx;
}

// hold sweep
FmtIO.Cmd("sens:swe:mode hold");

// stimulus settings
FmtIO.Cmd("SYST:CONF:EDEV:STAT 'MXG',ON");
FmtIO.Cmd("sens:DISTortion:MODulate:SOURce 'MXG'");
FmtIO.Cmd("SOURce:MODulation:LOAD 'D:\Symphony\FR1_100M_64QAM.mdx'");
FmtIO.Cmd("SOURce:MODulation:STATe 1");

// sweep
FmtIO.Cmd("sens:swe:mode single");
FmtIO.CmdQry("*OPC?", TimeOuts._2min);

// display scale
FmtIO.Cmd("disp:wind1:trac:y:scal:coup on");
FmtIO.Cmd("disp:wind1:y:auto");
_fmtIO.CmdQry("*OPC?", TimeOuts._1min);

string desktopDir =
Environment.GetFolderPath(Environment.SpecialFolder.Desktop);

// read display data
{
    _fmtIO.Cmd("format:border swapped");
    _fmtIO.Cmd("format:data real,32");
    _fmtIO.Cmd("CALC:MEAS1:DATA:SDAT?");
    Complex[] POut2S = ToComplex(_fmtIO.ReadFloats());
    _fmtIO.Cmd("CALC:MEAS1:DATA:FDAT?");
    float[] POut2F = _fmtIO.ReadFloats();
    _fmtIO.Cmd("CALC:MEAS1:MATH:MEM"); // POut2, Data to Memory
    _fmtIO.Cmd("CALC:MEAS1:DATA:SMEM?");
    Complex[] POut2Smem = ToComplex(_fmtIO.ReadFloats());
    _fmtIO.Cmd("CALC:MEAS1:DATA:FMEM?");
    float[] POut2Fmem = _fmtIO.ReadFloats();
    SaveColumnsToCSV(Path.Combine(desktopDir, "mod_data_trace.csv"),
    new string[] { "POut2 FDAT", "POut2 SDAT", "POut2 FMEM", "POut2 SMEM" },
    POut2F, POut2S, POut2Fmem, POut2Smem);
}

// read tone by tone data
{
    _fmtIO.Cmd("format:border swapped");
    _fmtIO.Cmd("format:data real,64");
double[] freqTones = _fmtIO.ReadDoubles();
_fmtIO.Cmd("format: data real, 32");

Complex[] POut2 = ToComplex(_fmtIO.ReadFloats());
_fmtIO.Cmd("CALC: MEAS: DATA: BUFFER? 'PIn1'");
Complex[] PIn1 = ToComplex(_fmtIO.ReadFloats());
_fmtIO.Cmd("CALC: MEAS: DATA: BUFFER? 'MSig2'");
Complex[] MSig2 = ToComplex(_fmtIO.ReadFloats());
_fmtIO.Cmd("CALC: MEAS: DATA: BUFFER? 'MDist2'");
Complex[] MDist2 = ToComplex(_fmtIO.ReadFloats());
_fmtIO.Cmd("CALC: MEAS: DATA: BUFFER? 'MGain21'");
Complex[] MGain21 = ToComplex(_fmtIO.ReadFloats());
_fmtIO.Cmd("CALC: MEAS: DATA: BUFFER? 'MComp21'");
Complex[] MComp21 = ToComplex(_fmtIO.ReadFloats());
_fmtIO.Cmd("CALC: MEAS: DATA: BUFFER? 'PGain21'");
Complex[] PGain21 = ToComplex(_fmtIO.ReadFloats());
_fmtIO.Cmd("CALC: MEAS: DATA: BUFFER? 'S11'");
Complex[] S11 = ToComplex(_fmtIO.ReadFloats());
_fmtIO.Cmd("CALC: MEAS: DATA: BUFFER? 'S21'");
Complex[] S21 = ToComplex(_fmtIO.ReadFloats());
_fmtIO.Cmd("CALC: MEAS: DATA: BUFFER? 'LPIn1'");
Complex[] LPIn1 = ToComplex(_fmtIO.ReadFloats());
_fmtIO.Cmd("CALC: MEAS: DATA: BUFFER? 'LPOut1'");
Complex[] LPOut1 = ToComplex(_fmtIO.ReadFloats());

FmtIO.Cmd("CALC:MEAS:DATA:BUFFer? 'LPOut2'");
Complex[] LPOut2 = ToComplex(_fmtIO.ReadFloats());

SaveColumnsToCSV(Path.Combine(desktopDir, "mod_data_tone_by_tone.csv"),
    new string[] { "Freq", "POut2", "PIn1", "MSig2", "MDist2", "MGain21", "MComp21", "PGain21", "S11", "S21", "LPIn1", "LPOut1", "LPOut2" },
    freqTones, ToLogMag(POut2), ToLogMag(PIn1), ToLogMag(MSig2), ToLogMag(MDist2), ToLogMag(MGain21), ToLogMag(MComp21), ToLogMag(PGain21), ToLogMag(S11), ToLogMag(S21), ToLogMag(LPIn1), ToLogMag(LPOut1), ToLogMag(LPOut2));

// setup distortion table display
FmtIO.Cmd("disp:wind:tabl dist");

string strColVis =
FmtIO.CmdQry("sens:dist:tabl:disp:cat?");

string[] cols = strColVis.Split(new char[] { ',' }, StringSplitOptions.RemoveEmptyEntries);

FmtIO.Cmd(string.Format("sens:dist:tabl:disp:del '{0}'", cols[0]));

FmtIO.Cmd(string.Format("sens:dist:tabl:disp:feed '{0}'", cols[0]));

// get table data

int bandCnt =
int.Parse(_fmtIO.CmdQry("sens:DISTortion:MEASure:BAND:COUNt?")));

string rowCat = _fmtIO.CmdQry("sens:dist:tabl:cat?");
{
    string colCat =
FmtIO.CmdQry("sens:dist:tabl:data:cat?");

string[] paramNames = new string[] { "EVM DistEq41"}
object[] col1 = new object[1 + bandCnt]; col1[0] = paramNames[0];
object[] col2 = new object[1 + bandCnt]; col2[0] = paramNames[0];
for (int bandId = 1; bandId <= bandCnt; ++bandId)
{
    col1[bandId] = float.Parse(_fmtIO.CmdQry(string.Format("sens:dist:tabl:data:val? {0},{1}", bandId, paramNames[0])));
    col2[bandId] = float.Parse(_fmtIO.CmdQry(string.Format("sens:dist:tabl:data:val? {0},{1}", bandId, paramNames[1])));
}
SaveColumnsToCSV(Path.Combine(desktopDir, "mod_evm.csv"), null, col1, col2);
}
// all ACPEVM parameters
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND:INIT");
_fmtIO.Cmd("sens:swe:mode single");
_fmtIO.CmdQry("*OPC?", TimeOuts._2min);
{
    string colCat = _fmtIO.CmdQry("sens:dist:tabl:data:cat?");
    string[] paramNames = colCat.Split(new char[] { ',', }, StringSplitOptions.RemoveEmptyEntries);
    float[] col1 = new float[paramNames.Length];
    for (int i = 0; i < paramNames.Length; ++i)
    {
        col1[i] =
float.Parse(_fmtIO.CmdQry(string.Format("sens:dist:tabl:data:val? {0},{1}"", 1, paramNames[i])));
}

SaveColumnsToCSV(Path.Combine(desktopDir, "mod_ACPEVM.csv"), null, paramNames, coll);

// all NPR parameters
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND:INIT");
_fmtIO.Cmd("sens:DISTortion:MEASure:BAND:TYPE NPR");
_fmtIO.Cmd("sens:swe:mode single");
_fmtIO.CmdQry("*OPC?", TimeOuts._2min);
{
    string colCat =
        _fmtIO.CmdQry("sens:dist:tabl:data:cat?");

    string[] paramNames = colCat.Split(new char[] { ',' }, StringSplitOptions.RemoveEmptyEntries);

    float[] col1 = new float[paramNames.Length];

    for (int i = 0; i < paramNames.Length; ++i)
    {
        col1[i] =
            float.Parse(_fmtIO.CmdQry(string.Format("sens:dist:tabl:data:val? {0},{1}"", 1, paramNames[i])));
    }

    SaveColumnsToCSV(Path.Combine(desktopDir, "mod_NPR.csv"), null, paramNames, col1);
}
Source Modulation Calibration

The following C# example demonstrates how to perform a source modulation calibration to compensate the IQ data to achieve a flat frequency response in gain and phase at the reference plane.

**Note:** Traditional S-parameter calibration using Cal All... must be performed before performing a Source Modulation Calibration.

See the Modulation Distortion commands.

---

**See Other SCPI Example Programs**

```csharp
using System;
using System.Collections.Generic;
using System.IO;
using System.Numerics;
namespace mod_tests
{
    partial class Vna
    {
        [TestMethod]
        public void Case04()
        {
            // Source Modulation Calibration
            _fmtIO.Cmd("syst:fpr");
            _fmtIO.Cmd("disp:wind1:stat on");
            string tname = "PIn1";
            _fmtIO.Cmd(string.Format("calc:cust:def 'ch1_{0}','Modulation Distortion','{0}'", '{0}', tname));
        }
    }
}
```
fmtIO.Cmd(string.Format("disp:wind1:tracl:feed 'ch1_{0}'", tname));

fmtIO.Cmd("SYST:CONF:EDEV:STAT 'MXG',ON");
fmtIO.Cmd("sens:DISTortion:MODulate:SOURce 'MXG'");
fmtIO.Cmd("SOURce:MODulation:LOAD 'C:\Users\Public\Documents\Network Analyzer\mod\dev.mdx'");
fmtIO.Cmd("SOUR:MOD:STAT ON");
fmtIO.Cmd("SOUR:MOD:CORR:COLL:POW:REC 'DUTIn1'");
fmtIO.Cmd("SOUR:MOD:CORR:COLL:POW:ENAB ON");
fmtIO.Cmd("SOUR:MOD:CORR:COLL:FLAT:ENAB ON");
fmtIO.Cmd("SOUR:MOD:CORR:COLL:ACQ ASYN");
fmtIO.Cmd("*OPC");

var str = _fmtIO.CmdQry("*STB?");
int safeCnt = 0;

while (safeCnt < 1000)
{
    System.Threading.Thread.Sleep(100);
    str = _fmtIO.CmdQry("*STB?");
    if (int.Parse(str) > 0)
        break;
    ++safeCnt;
}

fmtIO.Cmd("SOUR:MOD:CORR:STAT ON");
}
Perform a Simple Source Power Cal

This example performs a Source Power Cal using ONE USB Power Sensor, already connected to the VNA.

A measurement must first be set up with desired frequency range, power, and so forth, ready to be calibrated.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as spc.vbs.

Learn how to setup and run the macro.

See Source Power Cal SCPI commands

See Other SCPI Example Programs

```
'Performs a source power cal on channel 1 - port 1 using a USB power sensor
'This example assumes ONE USB power sensor is connected to the VNA

Dim app
Dim scpi
Dim sensor

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.parse "SYST:PRES"

' set power accuracy tolerance and iterations
scpi.parse "SOUR1:POW1:CORR:COLL:ITER:NTOL 0.1"
scpi.parse "SOUR1:POW1:CORR:COLL:ITER:COUN 15"

' set power sensor settling tolerance
scpi.parse "SOUR1:POW1:CORR:COLL:AVER:NTOL 0.1"
```
scpi.parse "SOUR1::POW1:CORR:COLL:AVER:COUN 15"

'set offset value for amp or attenuation

scpi.parse "SOUR1::POW1:CORR:OFFS 0 DB"

'show source power cal dialog

scpi.parse "SOUR1::POW1:CORR:COLL:DISP ON"

'read the usb power sensor ID string

sensor=scpi.parse("SYST:COMM:USB:PMET:CAT?")

'specify that sensor

scpi.parse "SYST:COMM:PSEN usb," + sensor

'do the measurement

scpi.parse "SOUR1::POW1:CORR:COLL:ACQ PMR,"ASENSOR"

'save the source cal and create an R-Channel response calset

scpi.parse "SOUR::POW:CORR:COLL:SAVE RREC"
Perform a Source and Receiver Power Cal using SCPI

Programming the VNA using COM or using SICL/VISA over LAN (as in this example) leaves the VNA free to control GPIB devices as needed.

The first example, using Visual Basic, demonstrates the following:

- Performing a source power calibration of Port 2 for Channel 1.
- Reading the calibration data.

The second example performs a Receiver Power Cal using VBScript.

Learn more about Power Calibrations.

See an example that Uploads a Source Power Cal.

Other SCPI Example Programs

To run this program, you need:

- One of the following power meters connected to the VNA through GPIB: E4416A, E4417A, E4418A/B, E4419A/B, 437B, 438A, EPM-441A, EPM-442A

**Note:** If your power meter is other than these, you can create your own Power Meter Driver using our template.

- Your PC and VNA both connected to a LAN (for communicating with each other).
- The SICL and VISA components of Agilent’s I/O Libraries software installed on your PC (both are included when you install the software, unless you already have another vendor’s VISA installed. Then specify Full SICL and VISA installation to overwrite the other vendor’s VISA.
- The module visa32.bas added to your VB project.
- A form with one button labeled cmdRun.
- A VISA interface configured on your remote PC to control the VNA. This could be GPIB interface or a VISA LAN Client.
On the VNA connect a Thru cable from port 1 to port 2.

Note: The `SOURce:POWER:CORRection:COLLect:ACQuire` command, when used with a power meter, cannot be sent over the GPIB unless the power meter is connected to a different GPIB interface. See the alternative methods described in the command details.

```vbnet
Private defRM As Long

' Session to VISA Default Resource Manager
Private viPNA As Long

' VISA function status return code
Private status As Long

Private Sub Form_Load()
    defRM = 0
    End Sub

Private Sub cmdRun_Click()
    ' String to receive data from the VNA.
    ' Dimensioned large enough to receive scalar comma-delimited values
    ' for 21 frequency points (20 ASCII characters per point)
    Dim strReply As String * 420

    Dim strStimulus, strCalValue
    Dim strResult As String

    ' Open the VISA default resource manager
    status = viOpenDefaultRM(defRM)
    If (status < VI_SUCCESS) Then HandleVISError
```
Open a session (viPNA) to the VNA at "address 16" on the VISA interface configured as "GPIB1" on this PC. This could be a VISA LAN Client pointing to the SICL LAN Server on the VNA, or an actual GPIB interface on this PC connected to the VNA GPIB (in which case the power meter would need to be connected to a different GPIB interface on the VNA, such as the Keysight 82357A USB-to-GPIB).

status = viOpen(defRM, "GPIB0::16::INSTR", 0, 0, viPNA)
If (status < VI_SUCCESS) Then HandleVISAError

Set the number of sweep points to 21 on Channel 1.
status = myGPIBWrite(viPNA, "SENS1:SWE:POIN 21")
If (status < VI_SUCCESS) Then HandleVISAError

Specify the GPIB address of the power meter that will be used in performing the calibration.
status = myGPIBWrite(viPNA, "SYST:COMM:GPIB:PMET:ADDR 13")
If (status < VI_SUCCESS) Then HandleVISAError

Turn use of the loss table OFF (this assumes there is virtually no loss in the RF path to the power sensor due to a splitter, coupler or adapter).
status = myGPIBWrite(viPNA, "SOUR:POW:CORR:COLL:TABL:LOSS OFF")
If (status < VI_SUCCESS) Then HandleVISAError

Turn frequency checking OFF (so one power sensor is used for the entire calibration acquisition sweep regardless of frequency span).
status = myGPIBWrite(viPNA, "SOUR:POW:CORR:COLL:FCH OFF")
If (status < VI_SUCCESS) Then HandleVISAError

' Specify a nominal power accuracy tolerance (NTOLerance) in dB for the calibration,
' and the maximum number (COUNt) of iterations to adjust power at each point,
' attempting to achieve within tolerance of the desired power. If at any stimulus
' point the power fails to reach within the set tolerance of the desired power
' after the maximum number of iterations, the power at that point will be set to the
' value determined by the last iteration (the Source Power Cal dialog box will
' indicate the FAIL, but we can still apply the cal if desired when it's complete).
' Each iteration is based upon a SETTLED power reading (see comments preceding the
' next two commands below).

status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:ITER:NTOL 0.1")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:ITER:COUN 3")
If (status < VI_SUCCESS) Then HandleVISAError

' The worst-case window of power uncertainty (for a calibration which meets
' tolerance) is the sum of the iteration tolerance and the power meter settling
' tolerance (which is described below).

' At each stimulus point, the VNA takes power meter readings and determine when
' they have settled by comparing the magnitude difference between consecutive
' readings versus a nominal dB tolerance limit (NTOLerance) on that magnitude
' difference. When consecutive readings are within tolerance of each other, or
' if they are not within tolerance but we've taken a maximum number of readings
' (COUNT), the VNA does a weighted average of the readings taken at that stimulus point and that is considered our settled power reading.

status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:AVER:NTOL 0.1")
If (status < VI_SUCCESS) Then HandleVISAError

status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:AVER:COUN 5")
If (status < VI_SUCCESS) Then HandleVISAError

' Specify if the cal power level is offset (positive value for a gain, negative value for a loss) from the VNA port power setting on the channel when no source power cal is active. This is to account for components between the VNA test port and cal reference plane. In this example, we will calibrate at the VNA test port, so there is no offset (it is zero).

status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:OFFS 0 DB")
If (status < VI_SUCCESS) Then HandleVISAError

' Show the source power cal dialog during the source power cal acquisition. (this is the default, so this command is only necessary if this setting may have been changed beforehand, perhaps by another program).

status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:DISP ON")
If (status < VI_SUCCESS) Then HandleVISAError

' Clear the VNA's SCPI status registers because we are going to be monitoring the event status register to detect when the OPC bit gets set indicating the cal ACQuire completed.

status = myGPIBWrite(viPNA, "*CLS")
If (status < VI_SUCCESS) Then HandleVISAError

' Setting the I/O timeout value to 6000 milliseconds (6 seconds), because the VNA may take up to that amount of time to respond to some commands/queries
' while the cal ACQuire is progressing.

status = viSetAttribute(viPNA, VI_ATTR_TMO_VALUE, 6000)

If (status < VI_SUCCESS) Then HandleVISAError

' Specify the method (type of device) that will be used to perform the cal.

' Choose from power meter (PMEter), power meter and receiver (PMReceiver)

' or just receiver (REceiver).

' PMReceiver uses the power meter for the first iteration of each point and
' the VNA's reference receiver for subsequent iterations, so is much faster
' than using power meter only. But the power meter accounts for compression
' when calibrating at the output of an active device, whereas the reference
' receiver cannot unless it is coupled to the cal reference plane (on a VNA
' which allows direct access to the receivers).

' Perform the source power cal acquisition sweep using the sensor attached to
' Channel A of the power meter (asking for an OPC reply when it’s done). This
' assumes that the power sensor is already connected to Port 2 of the VNA.

' We'll put up an hourglass cursor while waiting for the acquire to complete.

Screen.MousePointer = vbHourglass

status = myGPIBWrite(viPNA, "SOUR1:POW:CORR:COLL:ACQ PMET,'ASEN','Port 2',ASYNchronous;*OPC")

If (status < VI_SUCCESS) Then HandleVISAError

' Other valid selections would be the following:

' This mode uses Power Meter and Reference Receiver

'status = myGPIBWrite(viPNA, "SOUR1:POW:CORR:COLL:ACQ PMR,'BSEN','Port 2',ASYN;*OPC")

' This mode uses VNA receiver only (no power meter)

'status = myGPIBWrite(viPNA, "SOUR1:POW:CORR:COLL:ACQ REC,'b1','Port 2',ASYN;*OPC")
' Polling in a loop to detect when the OPC bit (bit 0, weight value 1) gets
set in the Event Status Register indicating the ACQuire finished. In this
type of loop is where you could do other operations in-between the polling
(like having your program’s user-interface still respond to user input).

If instead of Visual Basic you are programming in C or C++, as an
alternative to having a polling loop like this, you could set up an SRQ
handling function in your program (for example, see the documentation
supplied with your vendor’s implementation of VISA on how to register for
callback when an SRQ event occurs).

Do

    status = myGPIBWrite(viPNA, "*ESR?")

    If (status < VI_SUCCESS) Then HandleVISAError

    status = myGPIBRead(viPNA, strReply)

    If (status < VI_SUCCESS) Then HandleVISAError

    esrByte = CByte(strReply)

Loop While (esrByte And 1) = 0

' Change mouse cursor from hourglass back to normal

Screen.MousePointer = vbDefault

' Conclude the calibration. This applies the cal data to VNA channel memory,
' and turns the correction ON for Port 2 on Channel 1,
' but does NOT save the calibration.

    status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:SAVE")

    If (status < VI_SUCCESS) Then HandleVISAError

' At this point, if you choose to save the instrument state as a ".CST" file,
' the calibration will be saved with the instrument state in that file.
' Prepare for doing data transfer in ASCII format.
status = myGPIBWrite(viPNA, "FORM:DATA ASCII")
If (status < VI_SUCCESS) Then HandleVISAError

' Read the stimulus values from Channel 1.
status = myGPIBWrite(viPNA, "SENS1:X?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Tokenize the reply string into an array containing the values
strStimulus = Split(strReply, ",")

' Read the source power correction data.
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:DATA?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Tokenize the reply string into an array containing the values
strCalValue = Split(strReply, ",")

' Print the data using a message box (here, Chr returns the ASCII characters
' for Tab (9) and Linefeed (10)).
strResult = "Stimulus" & Chr(9) & Chr(9) & "Cal Value" & Chr(10)
For i = 0 To UBound(strStimulus)
strResult = strResult & Val(strStimulus(i)) & Chr(9) & Val(strCalValue(i)) &
Private Function myGPIBWrite(ByVal viHandle As Long, ByVal strOut As String) As Long

' The "+ Chr$(10)" appends an ASCII linefeed character to the
' output, for terminating the write transaction.
myGPIBWrite = viVPrintf(viHandle, strOut + Chr$(10), 0)
End Function

Private Function myGPIBRead(ByVal viHandle As Long, strIn As String) As Long

myGPIBRead = viVScanf(viHandle, "%t", strIn)
End Function

Sub HandleVISAError()
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr, vbExclamation

' Close the resource manager session (which also closes
' the session to the VNA).
If defRM <> 0 Then Call viClose(defRM)
End
End Sub
Public Sub Wait(ByVal mS_delay As Long)

    Dim t0 As Single
    t0 = Timer
    Do While Timer - t0 < mS_delay / 1000
        Dim dummy As Integer
        dummy = DoEvents() ' if we cross midnight, back up one day
        If Timer < t0 Then t0 = t0 - 86400
    Loop
End Sub

Perform a Receiver Power Cal

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as *.vbs.

Learn how to setup and run the macro.

Dim pna
Dim scpi
Set pna = CreateObject("AgilentPNA835x.Application")
Set scpi = pna.ScpiStringParser
    ' For simplicity, this example starts from the preset instrument state
    scpi.Execute "SYST:PRESet"
    ' Turn off continuous sweep
    scpi.Execute "INITiate:CONTinuous OFF"
    ' Select the S11 measurement that was created by the instrument preset
    scpi.Execute "CALCulate:PARameter:SELect 'CH1_S11_1'"
    ' Change the measurement parameter to measure the B receiver
    scpi.Execute "CALCulate:PARameter:MODify B,1"
    ' Specify the Calibration Type, then Prompt
    ' to ensure the receiver is connected to port 1.
    scpi.Execute "SENSe:CORRection:COLLect:METHod RPOWer"
MsgBox "Connect port 1 to port 2 so power is supplied to the B receiver, then press enter"
    ' Acquire the power measurement; returning reply to *OPC? when finished.
response = scpi.Execute( "SENSe:CORRection:COLLect:ACQuire POWER;*OPC?" )
    ' Compute the error term, store to calset and turn on the calibration.
response = scpi.Execute( "SENSe:CORRection:COLLect:SAVE" )
MsgBox "Done with calibration."
Perform a Source Power Cal with TWO Sensors

This example performs a Source Power Cal using TWO power sensors on a Dual-Channel Power Meter connected to the VNA via GPIB. Use of two power sensors is necessary when the frequency range of the measurement is greater than can be attained with a single sensor.

A measurement must first be set up with desired frequency range, power, and so forth, ready to be calibrated.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do not need to control the VNA via GPIB to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as spc.vbs.

Learn how to setup and run the macro.

See Source Power Cal SCPI commands

```
'Performs a source power cal on channel 1 - port 1 using TWO USB power sensors.

Dim app
Dim scpi
Dim sensor

' Create / Get the VNA application.
set app = CreateObject("AgilentPNA835x.Application")
set scpi = app.ScpiStringParser

scpi.parse "SYST:PRES"

' set power accuracy tolerance and iterations
scpi.parse "SOUR1:POW1:CORR:COLL:ITER:NTOL 0.1"
scpi.parse "SOUR1:POW1:CORR:COLL:ITER:COUN 15"

' set power sensor settling tolerance
scpi.parse "SOUR1:POW1:CORR:COLL:AVER:NTOL 0.1"
scpi.parse "SOUR1::POW1:CORR:COLL:AVER:COUN 15"
```
'set offset value for amp or attenuation
scpi.parse "SOUR1:POW1:CORR:OFFS 0 DB"

'show source power cal dialog
scpi.parse "SOUR1:POW1:CORR:COLL:DISP ON"

'set frequency ranges for two power sensors
scpi.parse "SOUR:POW:CORR:COLL:ASEN 100E3, 3E9"
scpi.parse "SOUR:POW:CORR:COLL:BSEN 3E9,6E9"

'enable frequency check
scpi.parse "SOUR1:POW:CORR:COLL:FCHeck ON"

'specify the address of the power meter
scpi.parse "SYST:COMM:PSEN GPIB,'14'"

'do the measurements
MsgBox "Connect Sensor A to Port 1"
scpi.parse "SOUR1:POW1:CORR:COLL:ACQ PMR,'ASENSOR'"

MsgBox "Disconnect Sensor A from Port 1 and connect Sensor B to Port 1"
scpi.parse "SOUR1:POW1:CORR:COLL:ACQ PMR,'BSENSOR'"

'save the source cal and create an R-Channel response calset
scpi.parse "SOUR:POW:CORR:COLL:SAVE RREC"
Perform an ECaI User Characterization

This example performs a user-characterization and stores it to both the ECaI module memory and VNA disk memory. It also demonstrates the use of the EXPort, CLEar, IMPort and ‘KNAIe:INF?’ commands.

It then performs two 2-port cals: the first using the characterization from module memory, then using the characterization from disk memory.

Note: This example requires that channel 1 be already calibrated.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as ECaI.vbs.

Learn how to setup and run the macro.

See all ECaI User Characterization SCPI commands

See Other SCPI Example Programs

```vbs
Option Explicit

Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")

Dim scpi
Set scpi = pna.ScpiStringParser

' Substitute here the model number and serial number of your own ECaI.

' Note that this example corresponds to a 4-port ECaI module with
' serial number 00001. If you have a 2-port ECaI module, their model
' numbers are '5x5' numbers -- for example, 'N4691-60001'.

Dim ecalModelNum

ecalModelNum = "N4433A"

Dim ecalSerialNum
```
ecalSerialNum = "00001"


MsgBox "ECal module to be characterized is: " & 

' Set which user characterization number (1-12) the new characterization
' will be stored to in the ECal module when it is done. If you intend to
' store your user characterization just to VNA Disk Memory and NOT the
' ECal module's memory, then omit this command.

Dim characterizationNumber

characterizationNumber = 1


' The following commented-out lines of code show how you can access
' the list of connector type names you can set for the ports of an
' ECal when you user-characterize it. However, please note that if
' you are writing the user characterization to the ECal module's memory,
' as of yet only the Factory Defined set of connector choices will work
' properly (see SENS:CORR:CKIT:ECAL:CHAR:CONN:CAT?). If you will be saving
' your characterization to just VNA Disk Memory only, then all connector
' names returned by this query will work,
' user-defined connector names as well as factory-defined.

'Dim connTypeList


'MsgBox connTypeList

' For each port of the ECal module, specify which connector type
' is at the end of the adapter (or cable or fixture) that is
' connected to that port of the ECal for the characterization
(must be one of the connector types that is included in the list that "SENS:CORR:CKIT:ECAL:CHAR:CONN:CAT?" returns). The default is "No adapter", which assumes you are characterizing that port of the ECal "as is" (nothing attached to it). So in this example, Ports C and D of the ECal are being characterized to just the ECal's connectors.

scpi.Parse "SENS1:CORR:CKIT:ECAL:CHAR:CONN:PORT1 'APC 3.5 male'" ' ECal Port A

As with the connector types, the information set in these next few properties also gets stored within the characterization.

Set the name of the person and/or company that is producing this characterization.


Set user-specified description of the VNA being used.


Set descriptions of what you have connected to the ECal module's ports for the characterization.

Port A of the ECal

scpi.Parse "SENS1:CORR:CKIT:ECAL:CHAR:DESC:PORT1 '3.5 mm adapter, SN 00001'"

Port B of the ECal

scpi.Parse "SENS1:CORR:CKIT:ECAL:CHAR:DESC:PORT2 '3.5 mm adapter, SN 00002'"

Note that the "SENS:CORR:CKIT:ECAL:CHAR:" INITiate, ACQuire and SAVE ('CHAR:SAVE" but not "CHAR:DMEMory:SAVE") commands can all each take a significant amount of time to execute/complete. If you are looking at this example to leverage this functionality into a SCPI via GPIB or SCPI via SICL-LAN (VXI-11.2/11.3) application, then you could issue the "*CLS" and "*ESE 1" commands as shown in the commented-out lines below, and use your I/O libraries' Serial
Poll

' function to repeatedly read the status byte until you detect bit 5 (weight of 32)

' in that byte is set. That will happen when the command you are pairing with

' ":*OPC" has completed its operation. But that technique only works for the

GPLB

' and SICL-LAN interfaces. If you need to use the TCPIP Socket or COM

' ScpiStringParser (as is used in this example) SCPI interfaces where there's

' no "built-in" Serial Poll type of function, to ensure your program operates in

' a

' synchronized manner it will need to wait on the ":*OPC?" reply (and not time out)

' before proceeding to the next line of your program. In that event, we

' recommend

' you execute these commands on a thread of execution separate from your

' program's

' user interface thread.

' Of the "SENS:CORR:CKIT:ECAL:CHAR:" INITiate, ACQuire and SAVE commands, the

' SAVE

' command takes the longest amount of time to complete (unless you've set up your

' measurement channel to have a very slow sweep time, in which case the ACQuire

' command could take longer). For an ECal that is a N469x, N4432A or N4433A, or an

' 8509x or N4431x produced by Keysight in 2005 or later, the SAVE command can

' take a

' maximum of approximately 4 to 5 minutes to complete (that corresponds to a

' characterization that will result in the ECal's memory becoming completely

' filled).

' For an 8509x or N4431x ECal that was produced in 2004 or earlier, the SAVE

' command

' can take a maximum of 9 to 10 minutes to complete (again that corresponds to a

' characterization that will result in the ECal's memory becoming completely

' full).
' Begin a user characterization on Channel 1.

' If you will be storing this characterization to the ECal module's memory, then
' the boolean argument to this command is optional (but if you choose to include
' for that case then you must specify it as 1 or ON). If you will be storing
' characterization to VNA disk memory ONLY, then you should specify 0 or OFF for
' that argument. In this example we will be storing the characterization to both
' module memory and VNA disk memory, so we can just omit the argument and let it
' default to 1.


Dim numSteps


Dim opcReply

' Dim statusByte

' Measure the steps.

' Note: prior to measuring the steps you must already have a calibration of the
' necessary number of ports applied to the channel (which in this example is
' Channel 1).

' Otherwise an error will be reported to the SCPI error queue.

Dim i

For i = 1 To numSteps

' Display the step's description.


' Clear the instrument's Status Byte.

' scpi.Execute "*CLS"

' Enable for the OPC bit (bit 0, which has weight 1) in the instrument's
' Event Status Register, so that when that bit's value transitions from 0 to 1
' then the Event Status Register bit in the Status Byte (bit 5 of that byte,
' weight 32) will become set.
'   scpi.Execute "*ESE 1"
'   Issue the ACQuire command
opcReply = scpi.Parse("SENS1:CORR:CKIT:ECAL:CHAR:ACQ STAN" & CStr(i) & ";*OPC?"
)
'   scpi.Execute "SENS1:CORR:CKIT:ECAL:CHAR:ACQ STAN" & CStr(i) & ";*OPC"
'   Do
   ' here is where if you leverage this example into an environment where
   ' you are using SCPI via GPIB or SICL-LAN, that in this loop you could do a
   ' Serial Poll via that interface to read the status byte into this
   ' statusByte variable. Then this If statement would detect when bit 5 is set.
   If ( (statusByte/32) Mod 2) Then Exit Do
   ' And note that normally you would want to have your program do some other
   ' processing (for example, check for user input from keyboard/mouse, for
   ' a cancellation request) here in this loop.
   Loop
   MsgBox "ACQuire is complete"
Next
MsgBox "Now the user characterization will be saved to the ECal module and to PNA disk memory"
' scpi.Execute "*CLS;*ESE 1"
' Save the user characterization to the ECal module's memory.
)
' Do
   ' again here you could do a Serial Poll to get statusByte if using GPIB or SICL-LAN
   If ( (statusByte/32) Mod 2) Then Exit Do
' Loop
Save the user characterization to VNA Disk Memory.

Dim characterizationName
characterizationName = "test"


Dim pnaDiskMemCalKitName
pnaDiskMemCalKitName = GetCalKitName(characterizationName)

Exporting the characterization from VNA disk memory into a file.

The file can be used for loading the characterization into VNA disk memory on another VNA.


Demonstrating that the characterization can be cleared from VNA disk memory and then re-loaded (IMPORTed) from the file that was created by the "SENS:CORR:CKIT:ECAL:EXP".


Dim moduleMemCalKitName
moduleMemCalKitName = GetCalKitName("User " & CStr(characterizationNumber))


MsgBox "User characterization is complete. Now we will calibrate using it. First we will use it from ECal module memory."

DoTwoPortCal moduleMemCalKitName

MsgBox "Now we will calibrate using the characterization from PNA Disk Memory."

DoTwoPortCal pnaDiskMemCalKitName

MsgBox "Example has completed"

Function GetCalKitName(characterizationName)
Dim calKitName

calKitName = ecalModelNum

If Len(characterizationName) > 0 Then calKitName = calKitName & " " & characterizationName

calKitName = calKitName & " ECal " & ecalSerialNum

GetCalKitName = calKitName

End Function

Sub DoTwoPortCal(calKitName)

' Specify the DUT connector for each VNA port to be calibrated (DUT connector = ECal characterization's connector)
scpi.Parse "SENS1:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 male'"
scki.Parse "SENS1:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 male'"

' Specify the "cal kit" for each of those ports
scpi.Parse "SENS1:CORR:COLL:GUID:CKIT:PORT1 '" & calKitName & '""
scki.Parse "SENS1:CORR:COLL:GUID:CKIT:PORT2 '" & calKitName & '""

' This results in a calibration sequence of a single "connection step"
scpi.Parse "SENS1:CORR:COLL:GUID:INIT"

' Acquire the cal connection step
opcReply = scpi.Parse("SENS1:CORR:COLL:GUID:ACQ STAN1;*OPC?")

' Again here instead of waiting for opcReply you could do a Serial Poll to get statusByte if using GPIB or SICL-LAN
'scki.Execute "SENS1:CORR:COLL:GUID:ACQ STAN1:*OPC"

'Do

' If ( (statusByte/32) Mod 2) Then Exit Do

'Loop

' Conclude the cal and turn it on
scpi.Parse "SENS1:CORR:COLL:GUID:SAVE"

End Sub
Perform an Unguided Cal on a 4-Port VNA

This topic describes how to perform an unguided calibration on a multiport network analyzer using SCPI. The objective here is to make clear the relationship between the physical port on which a standard is being measured, the actual device in the cal kit, and the SCPI command used to acquire the device.

There are two sets of SCPI commands that acquire calibrations. One set is used for guided cal, the other for unguided. The SCPI commands that provide remote access to unguided cal are in the SENS:CORR:COLL block:

- SENS:CORR:COLL:METHod
- SENS:CORR:COLL:ACQuire
- SENS:CORR:COLL:SAVE

On a four port network analyzer, the remote programmer needs to be aware of the relationship between the physical port and the calibration kit class assignments. The example program (below) illustrates the usage by performing three unique 2 port cals, taking care to acquire the appropriate standards.

Calibration standards classes are ‘categories’ of standard types. To perform a 2 port calibration, the cal wizard requires the user to measure:

**3 reflection standards on the forward port:**

- Class S11A typically an open
- Class S11B typically a short
- Class S11C typically a load

Likewise, **3 reflection standards are required for the reverse port:**

- Class S22A typically an open
- Class S22B typically a short
- Class S22C typically a load

**There is also a transmission standard that is measured in both directions:**
Class S21T typically a thru

The following illustrates the relationship between cal kit physical standards and calibration classes.

Here is a list of the physical devices in my calibration kit.

Standard #1 = "3.5 mm male short"
Standard #2 = "3.5 mm male open"
Standard #3 = "3.5 mm male broadband load"
Standard #4 = "Insertable thru standard"
Standard #5 = "3.5 mm male sliding load"
Standard #6 = "3.5 mm male lowband load"
Standard #7 = "3.5 mm female short"
Standard #8 = "Female to female characterized thru adapter"
Standard #9 = "0-2 Load"
Standard #10 = "Open"
Standard #11 = "Non-insertable thru"
Standard #12 = "3.5 mm female lowband load"
Standard #13 = "3.5 mm female sliding load"
Standard #14 = "3.5 mm female broadband load"
Standard #15 = "3.5 mm female open"

When you perform a calibration remotely using SCPI, you don’t specify the device number directly. Rather, you specify the class you want to measure. Each device in the calibration kit is assigned to a class. And since more than one device can be assigned to the same class, each class contains an ordered list of devices. The class assignments are user-settable using the Advanced Modify Cal Kit dialog or the SCPI command:

```
SENS:CORR:COLL:CKIT:ORDer<class>, <std>, <std>, <std>, <std>,<std>,<std>,<std>
```

The 85052B kit used in the example program had the following standard list for each class: The list was obtained by issuing the corresponding SCPI query:

```
SENS:CORR:COLL:CKIT:OLIST1?  S11A = +2,+15,+0,+0,+0,+0,+0
```
When you perform the calibration, you acquire data by issuing the ACQuire command:

```plaintext
SENS:CORR:COLL:ACQ <class>[, <subst> ]
```

For example:

```plaintext
SENS:CORR:COLL:SFOR 1
SENS:CORR:COLL:ACQ STANA, SST2
```

The SFOR command tells the wizard to make the next acquisition in the forward direction. The ACQuire command specifies that we are measuring the 2nd device in the list for STANA. And since we are measuring SFORward, STANA refers to class #1 or S11A. The list of devices for this class are specified in the OLIST1 query above. The associations are shown in red.

Alternately, you could modify the device order for the S11A class to move device #15 into the first position (SENS:CORR:COLL:CKIT:ORDER1). When the desired device is in the first position, you needn’t specify the order number in the ACQuire command. The default is the first device in the OLIST. This worked well for two port network analyzers where the order for S11A,B,C classes were setup for port 1 and the order for S22A,B,C was set up for port 2. With the kit setup in the proper order, you could eliminate the specification of the substandard number (SST<\text{n}>).

When performing 2 port calibrations on 4 Port Network Analyzers, the wizard applies S11A,B,C standards to the lower numbered port, S22A,B,C standards to the higher numbered port. Since the two classes (S11A,B,C and S22A,B,C) are applied to multiple ports, the programmer must take into account the ports being measured and take greater care when specifying the ACQuire command to ensure that the correct device is being measured.

**Port to class relationship**
The following example program shows one method of handling two port cals on a multiport network analyzer. The connectors at the measurement plane are assumed to be (1) male, (2) female, (3) male, and (4) male. In the example, three cals are performed: 1-2 (insertable male to female), 2-3 (insertable female to male), and 3-4 (noninsertable using an characterized adapter).

```vbscript
option explicit
public scpi
public pna
' assume a 4 port VNA with the following connectors:
' the standard measured on these ports will be the opposite gender
' PORT 1 = 3.5 male
' PORT 2 = 3.5 female
' PORT 3 = 3.5 male
' PORT 4 = 3.5 male
' To perform 2 port calibrations between 1-2, 2-3, and 3-4 you need to do the following

call main

sub main
set pna = CreateObject("AgilentPnA835x.Application")
set scpi = pna.ScpiStringParser
pna.Preset
' select a kit to use for this demonstration
' kit #1 for the N5230A is the 85052B 3.5mm kit with sliding load
scpi.execute("SENS:CORR:COLL:CKIT:SELECT 1")
PrintKitStandardInfo 1
PrintKitOlist 1

' --------------------------------------------
' CALIBRATE PORTS 1 and 2, insertable cal
' --------------------------------------------
wscript.echo
wscript.echo "Calibrating ports 1 and 2"
scpi.execute("SYST:PRES;")
scpi.execute("calc:par:sel CH1_S11_1")
scpi.execute("SENS:CORR:TST:STATE 0")
```

<table>
<thead>
<tr>
<th>Ports</th>
<th>S11A Port</th>
<th>S22A Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1,3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>1,4</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>2,3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>2,4</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3,4</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>
scpi.execute("SENS:CORR:COLL:METHod SPARSOLT")
scpi.execute("SENS:CORR:SFOR 1")
MeasureFemaleStandards 1
scpi.execute("SENS:CORR:SFOR 0")
MeasureMaleStandards 2
MeasureTransmissionStandards 1,2
scpi.execute("SENS:CORR:COLL:SAVE")

' ------------------------------------------------------
'   CALIBRATE PORTS 2 and 3, insertable cal
' ------------------------------------------------------

wscript.echo
wscript.echo "Calibrating ports 2 and 3"
scpi.execute("SYST:PRES;")
scpi.execute("calc:par:sel CH1_S11_1")
scpi.execute("calc:par:mod S23")
scpi.execute("SENS:CORR:TST:STATE 0")
scpi.execute("SENS:CORR:COLL:METHod SPARSOLT")
scpi.execute("SENS:CORR:SFOR 1")
MeasureMaleStandards 2
scpi.execute("SENS:CORR:SFOR 0")
MeasureFemaleStandards 3
MeasureTransmissionStandards 2,3
scpi.execute("SENS:CORR:COLL:SAVE")

' ------------------------------------------------------
'   CALIBRATE PORTS 3 and 4, non-insertable cal
' ------------------------------------------------------

wscript.echo
wscript.echo "Calibrating ports 3 and 4"
scpi.execute("SYST:PRES;")
scpi.execute("calc:par:sel CH1_S11_1")
scpi.execute("calc:par:mod S43")
scpi.execute("SENS:CORR:COLL:METHod SPARSOLT")
scpi.execute("SENS:CORR:SFOR 1")
MeasureFemaleStandards 3
scpi.execute("SENS:CORR:SFOR 0")
MeasureFemaleStandards 4
MeasureAdapter 3, 4
scpi.execute("SENS:CORR:COLL:SAVE")
end sub

sub MeasureMaleStandards ( portNumber )
dim portstr
portstr = formatnumber(portNumber,0)
Promptconnect1 1, 1, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN1;*OPC?")

Promptconnect1 2, 1, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN2;*OPC?")
Promptconnect1 3, 3, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN3,SST3;*OPC?")
sub MeasureFemaleStandards (portNumber)
    dim portstr
    portstr = formatnumber(portNumber, 0)
    Promptconnect1 1, 2, portNumber
    scpi.execute("SENS:CORR:COLL:ACQ STAN1,SST2;*OPC?"
    Promptconnect1 2, 2, portNumber
    scpi.execute("SENS:CORR:COLL:ACQ STAN2,SST2;*OPC?"
    Promptconnect1 3, 6, portNumber
    scpi.execute("SENS:CORR:COLL:ACQ STAN3,SST6;*OPC?"
end sub

sub MeasureTransmissionStandards( port1, port2)
    dim p1str
    dim p2str
    p1str = formatnumber( port1, 0)
    p2str = formatnumber( port2, 0)
    Promptconnect2 4, 1, port1, port2
    scpi.execute("SENS:CORR:COLL:ACQ STAN4;*OPC?"
end sub

sub MeasureAdapter( port1, port2)
    dim p1str
    dim p2str
    p1str = formatnumber( port1, 0)
    p2str = formatnumber( port2, 0)
    Promptconnect2 4, 2, port1, port2
    scpi.execute("SENS:CORR:COLL:ACQ STAN4,SST2;*OPC?"
end sub

' return the nth item in the comma separated list
Function GetItemNumber( list, n)
    dim strVector
    strVector = split(list,",",-1,1)
    GetItemNumber = strVector(n-1)
end function

' remove the trailing newline from str
function chop( str )
    dim tmp
    tmp = str
    ' remove the appended newline
    dim pos
    pos = InStrRev(tmp,vblf)
    if (pos >0) then
        tmp = mid(tmp,1,pos-1)
    end if
    chop = tmp
end function
'return the label for the nth standard assigned to the class described by class_index.'
'if class_index = 1, class is S11A (STAN1)
'if class_index = 2, class is S11B (STAN2), etc

function GetStandardLabel( class_index, nth)
dim olist
dim stdnum
dim resp
olist = scpi.execute("SENS:CORR:COLL:CKIT:OLIST" + formatnumber(class_index,0)+"?")
stdnum = GetItemNumber( olist, nth)
scpi.execute("SENS:CORR:COLL:CKIT:STAN " + formatnumber(stdnum,0))
resp = scpi.execute("SENS:CORR:COLL:CKIT:STAN:LABel?")
GetStandardLabel = chop(resp)
end function

sub PromptConnect1( class_index, nth, port)
    wscript.echo "CONNECT " + GetStandardLabel( class_index, nth) + " to port " + formatnumber(port,0)
end sub

sub PromptConnect2( class_index, nth, port1, port2)
    wscript.echo "CONNECT " + GetStandardLabel( class_index, nth) + " between ports " + formatnumber(port1,0) + " and " + formatnumber(port2,0)
end sub

' Print the order of standards per class for this kit
sub PrintKitOlist( kit )
dim i
dim cmd
dim resp
wscript.echo
dim olistcmd
olistcmd = "SENS:CORR:COLL:CKIT:OLIST"
' list the sub standards for each of the following classes
' S11A, S11B, S11C, FWD TRANS, FWD ISOL, S22A, S22B, S22C, REV TRANS, REV ISOL
for i = 1 to 8
    cmd = olistcmd + formatNumber(i,0) + "?"
    resp = scpi.execute(cmd)
    wscript.echo cmd + "= " + chop(resp)
next
end sub

sub PrintKitStandardInfo( kit )
wscript.echo scpi.execute("SENS:CORR:COLL:CKIT:NAME?")
dim i
for i = 1 to 30
    dim slabel
dim snum
    snum = formatNumber(i,0)
    scpi.execute("SENS:CORR:COLL:CKIT:STAN " + snum)
    slabel = scpi.execute("SENS:CORR:COLL:CKIT:STAN:LABel?")
wscript.echo "Standard #"+snum+ " = " + chop(slabel)
The output from this program is as follows:

Microsoft (R) Windows Script Host Version 5.6

Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.

"85052B 3.5 mm with sliding load"
Standard #1 = "3.5 mm male short"
Standard #2 = "3.5 mm male open"
Standard #3 = "3.5 mm male broadband load"
Standard #4 = "Insertable thru standard"
Standard #5 = "3.5 mm male sliding load"
Standard #6 = "3.5 mm male lowband load"
Standard #7 = "3.5 mm female short"
Standard #8 = "female to female characterized thru adapter"
Standard #9 = "0-2 Load"
Standard #10 = "Open"
Standard #11 = "Non-insertable thru"
Standard #12 = "3.5 mm female lowband load"
Standard #13 = "3.5 mm female sliding load"
Standard #14 = "3.5 mm female broadband load"
Standard #15 = "3.5 mm female open"
Standard #16 = "Open"
Standard #17 = "Open"
Standard #18 = "Open"
Standard #19 = "Open"
Standard #20 = "Open"
Standard #21 = "Open"
Standard #22 = "Open"
Standard #23 = "Open"
Standard #24 = "Open"
Standard #25 = "Open"
Standard #26 = "Open"
Standard #27 = "Open"
Standard #28 = "Open"
Standard #29 = "Open"
Standard #30 = "Open"

SENS:CORR:COLL:CKIT:OLIST1? = +2,+15,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST2? = +1,+7,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST3? = +6,+5,+3,+12,+13,+14,+0
SENS:CORR:COLL:CKIT:OLIST4? = +4,+8,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST5? = +2,+15,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST6? = +1,+7,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST7? = +6,+5,+3,+12,+13,+14,+0
SENS:CORR:COLL:CKIT:OLIST8? = +4,+8,+0,+0,+0,+0,+0

Calibrating ports 1 and 2

CONNECT "3.5 mm female open" to port 1
CONNECT "3.5 mm female short" to port 1
CONNECT "3.5 mm female broadband load" to port 1
CONNECT "3.5 mm male open" to port 2
CONNECT "3.5 mm male short" to port 2
CONNECT "3.5 mm male broadband load" to port 2
CONNECT "Insertable thru standard" between ports 1 and 2
Calibrating ports 2 and 3
CONNECT "3.5 mm male open" to port 2
CONNECT "3.5 mm male short" to port 2
CONNECT "3.5 mm male broadband load" to port 2
CONNECT "3.5 mm female open" to port 3
CONNECT "3.5 mm female short" to port 3
CONNECT "3.5 mm female broadband load" to port 3
CONNECT "Insertable thru standard" between ports 2 and 3
Calibrating ports 3 and 4
CONNECT "3.5 mm female open" to port 3
CONNECT "3.5 mm female short" to port 3
CONNECT "3.5 mm female broadband load" to port 3
CONNECT "3.5 mm female open" to port 4
CONNECT "3.5 mm female short" to port 4
CONNECT "3.5 mm female broadband load" to port 4
CONNECT "female to female characterized thru adapter" between ports 3 and 4
Perform Global Delta Match Cal

The following program performs a Global Delta Match Calibration. This may be required when performing an Unknown Thru Cal or TRL Cal on PNA-L models. See example of Unknown Thru or TRL Cal.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Delta.vbs. Learn how to setup and run the macro.

```vb
Sub PerformGlobalDeltaMatchCal()
    Set pna = CreateObject("AgilentPNA835x.Application")
    Set scpi = pna.ScpiStringParser

    ' Initiate a Global Delta Match calibration, choosing connector and cal kit
    scpi.Parse "SENS:CORR:COLL:GUID:DMAT 'APC 3.5 female', '85033D/E'"

    ' Query the number of calibration steps
    retStr = scpi.Parse("SENS:CORR:COLL:GUID:STEP?")
    numSteps = CInt(retStr)

    ' Measure the cal standards
    For i = 1 To numSteps
        retVal = MsgBox(prompt, vbOKCancel)
        If retVal = vbCancel Then Exit Sub
        retStr = scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN" & CStr(i) & " ;*OPC?"")
    Next

    ' Compute the error coefficients and save the cal to Global Delta Match CalSet
    scpi.Parse "SENS:CORR:COLL:GUID:SAVE"
    MsgBox "Cal is done!"
End Sub
```
Perform a Guided 1-Port Cal on Port 2

This VBScript program does the following:

1. Clear measurements from the VNA
2. Create a new S22 measurement
3. Set an instrument state
4. Select the connector types
5. Select a cal kit
6. Initiate a Guided calibration
7. Display a prompt to connect each standard
8. Save the calibration to a newly created cal set

**Note:** This example illustrates an important step when calibrating a reflection measurement in the reverse direction. You MUST create a reverse (S22) measurement and have it be the active (selected) measurement on the channel that is being calibrated. This is not necessary for any calibrating any other measurement parameter.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Guided.vbs. Learn how to setup and run the macro.

```vbnet
Dim App
Set App = CreateObject("AgilentPNA835x.Application")
App.Preset

Dim step
Dim Parser
Dim prompt
Dim txtDat
Dim Chan

Rm Clear old measurements
App.Reset

Rm Create a new Measurement
```
Set Parser = App.SCPIStringParser
Parser.Parse "DISPlay:WINDow1:STATE ON"
Parser.Parse "CALCulate:PARameter:DEFine:EXT 'MyMeas',S22"
Parser.Parse "DISPlay:WINDow1:TRACe1:FEED 'MyMeas'"

Rem Initialize state
Set Chan = App.ActiveChannel
Chan.StartFrequency = 200e6
Chan.StopFrequency = 1.5e9
Chan.IFBandwidth = 1000
step = 3

Rem Begin a guided calibration
Parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'Not used'"
Parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'Type N (50) male'"
Parser.Parse "SENS:CORR:COLL:GUID:INIT"

Rem Query the number of steps

Rem Display the number of steps
MsgBox("Number of steps is " + txtDat)

Rem Set the loop counter limit
step = txtDat

Rem Measure the standards
For i = 1 To step
  If i = 1 Then
    MsgBox(prompt)
    Parser.Parse("sens:corr:coll:guid:acq STAN1")
  ElseIf i = 2 then
    MsgBox(prompt)
    Parser.Parse("sens:corr:coll:guid:acq STAN2")
  ElseIf i = 3 then
    MsgBox(prompt)
    Parser.Parse("sens:corr:coll:guid:acq STAN3")
  End If
Next

Rem All standards have been measured. Save the result
Parser.Parse "SENS:CORR:COLL:GUID:SAVE"
MsgBox("The calibration has been completed")
Perform a Guided Calibration using SCPI

This VBScript program performs a Guided Calibration using ECal or Mechanical standards. This example includes optional ECal orientation features.

This example has been updated to include:

- Guided Power Cal (Oct 8, 2010)
- The setting of Unknown Thru or Adapter Removal adapter delay. (March 2006).
- The activation of a channel to be calibrated. (Aug. 2006).

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Guided.vbs. Learn how to setup and run the macro.

```vbs
' Performing a Guided 2-port cal (Ports 1 and 2)
TwoPortGuidedCal
Sub TwoPortGuidedCal

Dim app
Dim scpi
Dim connList
Dim selectedConn1, selectedConn2
Dim kitList
Dim selectedKit
Dim message

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

'The following demonstrates that the Active Channel is cal'd
```
'Preset the VNA
scpi.Execute "SYST:UPR"

'Create a new measurement on Chan 2

'Now there are two windows, channels and measurements
'This becomes the Active Measurement
scpi.Execute ("DISPlay:WINDow2:STATE ON")

'Define a measurement name, parameter
scpi.Execute ("CALCulate2:PARameter:DEFine:EXT 'MyMeas',S21")

"FEED" the measurement
scpi.Execute ("DISPlay:WINDow2:TRACe1:FEED 'MyMeas'")

'This is the Active Measurement

'Activate the 'Preset' measurement to cal chan 1
scpi.Execute ("CALC1:PAR:SEL 'CH1_S11_1'")

' Query the list of connectors that the VNA system recognizes

' Format the list with linefeed characters in place of the commas
connList = FormatList(connList)

message = "Enter your DUT connector for Port 1.  Choose from this list:
message = message & Chr(10) & Chr(10) & connList

' Select the connector for Port 1
selectedConn1 = InputBox(message)
If selectedConn1 = "" Then Exit Sub
scpi.Execute "sens:corr:coll:guid:conn:port1 " & selectedConn1 & ""
message = "Enter your DUT connector for Port 2.  Again, choose from this list:
message = message & Chr(10) & Chr(10) & connList

' Select the connector for Port 2
selectedConn2 = InputBox(message)
If selectedConn2 = "" Then Exit Sub
scpi.Execute "sens:corr:coll:guid:conn:port2 " & selectedConn2 & ""

' Note: If your VNA has more than 2 ports, then uncomment
' one or both of these next two lines.
'scpi.Execute "sens:corr:coll:guid:conn:port3 ""Not used"" "
'scpi.Execute "sens:corr:coll:guid:conn:port4 ""Not used"" "

' This next block of commented code demonstrates how to specify an adapter
' and its electrical delay, in situations where you are performing an
' Unknown Thru or Adapter Removal calibration. In most situations, the
' VNA is able to correctly determine an adapter's electrical length
' at the end of the calibration. However, there are scenarios where
' the VNA cannot correctly calculate the length -- such as when the channel
' has a relatively small number of measurement points (for example, 201 or less)
' and the adapter is significantly long (for example, a cable that is several
' feet).
' In these cases, the ADAP commands (below) enable you to explicitly specify
' the adapter you are using.
' Send these commands prior to the "sens:corr:coll:guid:init" command.
' Create adapter and return the adapter number
'The adapterNum string contains a '+' character.
' Here we convert to integer to remove that.
'adapterNum = CStr( CInt(adapterNum) )
'Specify that this adapter has 10 nanoseconds electrical delay (coaxial).
'scpi.Execute "sens:corr:coll:guid:adap" & adapterNum & ":del 10E-9"
'Text description of adapter
'scpi.Execute "sens:corr:coll:guid:adap" & adapterNum & ":desc 'My adapter'"
' Select to use this adapter specifically between ports 1 and 2
'scpi.Execute "sens:corr:coll:guid:adap" & adapterNum & ":path 1,2"
' End of adapter block

' Query the list of acceptable cal kits and
' ECal module characterizations for Port 1.
' Format the list with linefeed
' characters in place of the commas
kitList = FormatList(kitList)
message = "Enter your cal kit or ECal module characterization for Port 1. 
"message = message & "Choose from this list:
message = message & Chr(10) & Chr(10) & kitList
' Select the Cal Kit or ECal module
' characterization to use for Port 1.
selectedKit = InputBox(message)
If selectedKit = "" Then Exit Sub
scpi.Execute "sens:corr:coll:guid:ckit:port1 " & selectedKit & "'"
' Query the list of acceptable cal kits
' and ECal module characterizations for Port 2.
' Format the list with linefeed characters in place of the commas
kitList = FormatList(kitList)
message = "Enter your cal kit or ECal module characterization for Port 2. 
"message = message & "Choose from this list:
message = message & Chr(10) & Chr(10) & kitList
' Select the Cal Kit or ECal module
' characterization to use for Port 2.
selectedKit = InputBox(message)
If selectedKit = "" Then Exit Sub

scpi.Execute "sens:corr:coll:guid:ckit:port2 " & selectedKit & ""

' This determines whether the cal will be a "Guided Power Cal"
' or just a traditional S-parameter cal.

message = "On which port number shall power be measured?  
message = message & "For a traditional guided cal without power cal, enter 0"
Dim powerPort

powerPort = CInt( InputBox(message) )
If powerPort > 0 Then
    scpi.Execute("sens:corr:coll:guid:psen" & CStr(powerPort) & "":stat on")

Dim retVal

retVal = MsgBox("Is the power sensor's connector type or gender different from the DUT connector for that port?", vbYesNo)
If retVal = vbYes Then
    message = "Enter your power sensor's connector.  Choose from this list:
    message = message & Chr(10) & Chr(10) & connList

    ' Select the sensor's connector.
    selectedConn1 = InputBox(message)
If selectedConn1 = "" Then Exit Sub

    scpi.Execute "sens:corr:coll:guid:psen" & CStr(powerPort) & "":conn " & selectedConn1 & ""

    ' Query the list of acceptable cal kits and ECal module characterizations
    ' that are applicable for the sensor's connector.

    ' Format the list with linefeed
    ' characters in place of the commas
    kitList = FormatList(kitList)
    message = "Enter your cal kit or ECal module characterization to use for de-embed
of the sensor's connector. "

message = message & "Choose from this list:"
message = message & Chr(10) & Chr(10) & kitList

' Select the Cal Kit or ECal module characterization to use for de-embed of the sensor's connector.

selectedKit = InputBox(message)
If selectedKit = "" Then Exit Sub

scpi.Execute "sens:corr:coll:guid:psen" & CStr(powerPort) & ":ckit '' & selectedKit & ":''
Else
scpi.Execute("sens:corr:coll:guid:psen" & CStr(powerPort) & ":conn 'Ignored'"")
End If ' End of block that considers the sensor's connector

' Ask for the power level to perform the power cal at

' (if this command is omitted, the default is 0 dBm).

Dim powerLevel
powerLevel = InputBox("Enter the power level for the power cal to be performed at")
If powerLevel = "" Then Exit Sub

scpi.Execute "sens:corr:coll:guid:psen" & CStr(powerPort) & ":pow:lev " & powerLevel
Else
End If ' End of block that considers if the cal will include power calibration

' This next block of commented code

' shows optional functions when using ECal.

' Send these "sens:corr:pref" commands prior to the

' Read ECAL information from ECal module #1 on the USB bus

' about the Keysight factory characterization data
'module1Info = scpi.Execute("sens:corr:coll:kit:inf? ECAL1,CHAR0")

'MsgBox "Description of ECal Module #1:" & Chr(10) & Chr(10) & module1Info

' The following command enables auto orientation of
' the ECal module (The VNA senses which port of the
' module is connected to which port of the VNA).
'scpi.Execute "sens:corr:pref:ecal:ori ON"

' However, if you are measuring at very low power levels where
' the VNA may fail to sense the module's orientation, then turn auto
' orientation OFF and specify how the module is connected.
' "A1,B2" indicates Port A of the module is connected
' to VNA Port 1 and Port B is connected to VNA Port 2).
'scpi.Execute "sens:corr:pref:ecal:ori OFF"

' End of optional ECal setup

'Select the thru method of "Default". This instructs the VNA to
'determine which thru standard measurement technique to use
'based upon the selected connectors and
'calibration kit(s) and the VNA model number.
'with new CMET and TMET 'default' is set by not sending the commands
'
'Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"
MsgBox "Number of steps is " + CStr(numSteps)
'Measure the standards
For i = 1 To numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)


MsgBox strPrompt, vbOKOnly, step

scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)

Next

' Conclude the calibration
scpi.Execute "sens:corr:coll:guid:save"

MsgBox "Cal is done!"

End Sub

Function FormatList(list)

Dim tokens

' Strip the leading and trailing quotation marks from the list string
list = Mid(list, 2, Len(list) - 3)

' Tokenize the comma-delimited list string into an array of the individual substrings
tokens = Split(list, ",")

' Rebuild the list string, placing linefeed characters where the commas were,
' using Trim to remove leading and trailing spaces.
list = ""

For i = 0 To UBound(tokens)
tokens(i) = Trim(tokens(i))
list = list & tokens(i) & Chr(9)
If i < UBound(tokens) Then
    i = i + 1
tokens(i) = Trim(tokens(i))
list = list & tokens(i) & Chr(10)

End Function
End If
Next
FormatList = list
End Function
Perform Guided ECal using SCPI

This VBScript program performs a Guided ECal Calibration. While this example is good to use as a starting point for Guided ECal, the Guided comprehensive cal example has some advanced features that are not in this program.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Guided.vbs. Learn how to setup and run the macro.

```vbs
' Performing a 2-port cal (Ports 1 and 2)
Dim app
Dim scpi

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' Specify the DUT connectors
' (for each connector of your DUT, one of the ECal module's ports must have
' that same connector, or else you cannot achieve the cal using that module).
scpi.Execute "sens:corr:coll:guid:conn:port1 ""APC 3.5 female"" "
scpi.Execute "sens:corr:coll:guid:conn:port2 ""APC 3.5 male"" "

' Note: If your VNA has more than 2 ports, you would need to uncomment
' one or both of these next two lines, to explicitly specify this is
' just a 2-port cal.
'scpi.Execute "sens:corr:coll:guid:conn:port3 ""Not used"" "
'scpi.Execute "sens:corr:coll:guid:conn:port4 ""Not used"" "
MsgBox "Connectors defined for Ports 1 and 2"

' Specify ECal modules
scpi.Parse "sens:corr:coll:guid:ckit:port1 'N4691-60004 ECal'

' Non-factory characterizations are specified as follows:
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 User 1 ECal'

' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
```
When Disk Memory ECal user characterizations are used, specify both the User char and the serial number as follows:

```
'
MsgBox "Cal kits defined for Ports 1 and 2"

' Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"
MsgBox "Number of steps is " + CStr(numSteps)

' Measure the standards
For i = 1 To numSteps
    step = "Step " + CStr(i) + " of " + CStr(numSteps)
    MsgBox strPrompt, vbOKOnly, step
    scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
Next

' Conclude the calibration
scpi.Execute "sens:corr:coll:guid:save"
MsgBox "Cal is done!"
Perform Guided Mechanical Cal using SCPI

This VBScript program performs a Guided Calibration using Mechanical standards. While this example is good to use as a starting point for guided mechanical cal, the Guided comprehensive cal example has some advanced features that are not in this program.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Guided.vbs. Learn how to setup and run the macro.

' Performing a 2-port cal (Ports 1 and 2)

Dim app
Dim scpi

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' Specify the DUT connectors
scpi.Execute "sens:corr:coll:guid:conn:port1 ""APC 3.5 female""
scpi.Execute "sens:corr:coll:guid:conn:port2 ""APC 3.5 male"

' Note: If your VNA has more than 2 ports, you would need to uncomment
' one or both of these next two lines, to explicitly specify this is
' just a 2-port cal.
'scpi.Execute "sens:corr:coll:guid:conn:port3 ""Not used"
'scpi.Execute "sens:corr:coll:guid:conn:port4 ""Not used"
MsgBox "Connectors defined for Ports 1 and 2"

' Select the Cal Kit for each port being calibrated.
scpi.Execute "sens:corr:coll:guid:ckit:port1 ""85052D"
scpi.Execute "sens:corr:coll:guid:ckit:port2 ""85052D"
MsgBox "Cal kits defined for Ports 1 and 2"

' Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"
MsgBox "Number of steps is " + CStr(numSteps)

' Measure the standards
'The following series of commands shows that standards
'can be measured in any order. These steps acquire
'measurement of standards in reverse order.
'It is easiest to iterate through standards using
' a For-Next Loop.
For i = numSteps To 1
step = "Step " + CStr(i) + " of " + CStr(numSteps)
MsgBox strPrompt, vbOKOnly, step
scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
Next

' Conclude the calibration
scpi.Execute "sens:corr:coll:guid:save"
MsgBox "Cal is done!"
Perform Guided TRL Calibration

This VBScript file performs a 2-Port Guided TRL calibration on 2-port VNA analyzers. (See an example of TRL cal on a 4-port VNA.) This program does the following:

- Clear old measurements from the VNA
- Create a new S22 measurement
- Set an instrument state
- Select the connectors and cal kit
- Initiate a Guided calibration
- Display a prompt as each new standard must be connected
- Save the calibration to a newly created cal set.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as TRL.vbs. Learn how to setup and run the macro.

```vbs
Dim App
Dim Parser
Dim Chan
Dim txtDat
Dim step
Dim parserTxt
Dim prompt
Set App = CreateObject("AgilentPNA835x.Application")
' Clear old measurements
App.Reset
' Create a new Measurement
```
Set Parser = App.SCPIStringParser
Parser.Parse "DISPlay:WINDow1:STATE ON"
Parser.Parse "CALCulate:PARameter:DEFine:EXT 'MyMeas',S12"
Parser.Parse "DISPlay:WINDow1:TRACe1:FEED 'MyMeas'"
' Initialize state
Set Chan = App.ActiveChannel
Chan.StartFrequency = 18.0e9
Chan.StopFrequency = 20.0e9
Chan.IFBandwidth = 1000
' Begin a guided calibrations
Parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 male'"
Parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 female'"
' Select TRL cal method.
Parser.Parse "SENS:CORR:COLL:GUID:PATH:CMET 1,2,'TRL'"
txtDat = Parser.Parse("SENS:CORR:COLL:GUID:PATH:CMET? 1,2")
MsgBox("Method " + txtDat)
Parser.Parse "SENS:CORR:COLL:GUID:INIT"
' Query the number of steps
' Display the number of steps
MsgBox("Number of steps is " + txtDat)
' Set the loop counter limit
step = CInt(txtDat)
' Measure the standards
For i = 1 To step
parserTxt = "sens:corr:coll:guid:desc? " + CStr(i)
prompt = Parser.Parse(parserTxt)
MsgBox(prompt)
parserTxt = "sens:corr:coll:guid:acq STAN" + CStr(i)
Parser.Parse (parserTxt)
Next
' All standards have been measured. Save the result
Parser.Parse "SENS:CORR:COLL:GUID:SAVE"
MsgBox("The TRL calibration has been completed")
Perform an Unguided 1-Port Cal on Port 2

This VBScript program does the following:

1. Clear measurements from the VNA
2. Create a new S22 measurement
3. Set an instrument state
4. Select a cal kit
5. Initiate an Unguided calibration
6. Display a prompt to connect each standard
7. Save the calibration to a newly created cal set

Note: This example illustrates an important step when calibrating a reflection measurement in the reverse direction. You MUST create a reverse (S22) measurement and have it be the active (selected) measurement on the channel that is being calibrated. This is not necessary for any calibrating any other measurement parameter.

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Unguided.vbs. Learn how to setup and run the macro.

```vbs
Dim App
Set App = CreateObject("AgilentPNA835x.Application")
App.Preset
Dim Parser
Dim Chan

Rem Clear old measurements
App.Reset

Rem Create a new Measurement
Set Parser = App.SCPIStringParser
Parser.Parse "DISPLAY:WINDow1:STATE ON"
Parser.Parse "CALCulate:PARameter:DEFine:EXT 'MyMeas',S22"
Parser.Parse "DISPLAY:WINDow1:TRACe1:FEED 'MyMeas'"
```
Rem Initialize state
Set Chan = App.ActiveChannel
Chan.StartFrequency = 200e6
Chan.StopFrequency = 1.5e9
Chan.IFBandwidth = 1000

Rem Begin an unguided calibration
Rem Set the calibration method
Parser.Parse "SENSe:CORRection:COLLect:METHOD REFL3"

Rem Turn off continuous sweep
Parser.Parse "INITiate:CONTinuous OFF"

Rem Select a cal kit
Parser.Parse "SENSe:CORRection:COLLect:CKIT:SELECT 1"

Rem Measure the standards
MsgBox("Connect OPEN to port 2. Then press OK")
Parser.Parse ("sens:corr:coll:acq STAN1")

MsgBox("Connect SHORT to port 2. Then press OK")
Parser.Parse ("sens:corr:coll:acq STAN2")

MsgBox("Connect LOAD to port 2. Then press OK")
Parser.Parse ("sens:corr:coll:acq STAN3")

Rem All standards have been measured. Save the result
Parser.Parse "SENS:CORR:COLL:SAVE"

Rem Turn ON continuous sweep
Parser.Parse "INITiate:CONTinuous ON"
MsgBox("The calibration has been completed")
Perform an Unguided 2-Port Mechanical Cal

This VBScript program performs an Unguided, Full 2-Port, calibration using ONE set of mechanical calibration standards.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Unguided.vbs. Learn how to setup and run the macro.

```vbs
Set App = CreateObject("AgilentPNA835x.Application")
Set Scpi = App.SCPIStringParser

'Initialize state
Scpi.Execute ("SYSTem:PRESet")

'Select the Preset measurement
Scpi.Execute ("CALCulate:PARameter:SELect 'CH1_S11_1'")

'Set the calibration method
Scpi.Execute ("SENSe:CORRection:COLLect:METHod SPARSOLT")

'Select a cal kit
Scpi.Execute ("SENSe:CORRection:COLLect:CKIT:SELect 1")

'Set one set of standards
Scpi.Execute ("SENSe:CORRection:TSTandards OFF")

'Set acquisition to FORWARD
Scpi.Execute ("SENSe:CORRection:SFORward ON")

'Measure the standards in forward direction
MsgBox "Connect OPEN to Port 1; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan1")

MsgBox "Connect SHORT to Port 1; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan2")

MsgBox "Connect LOAD to Port 1; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan3")

'Set acquisition to REVERSE
Scpi.Execute ("SENSe:CORRection:SFORward OFF")

'Measure the standards in reverse direction
MsgBox "Connect OPEN to Port 2; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan1")
```
MsgBox "Connect SHORT to Port 2; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan2")

MsgBox "Connect LOAD to Port 2; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan3")

'Measure the thru standard
MsgBox "Connect THRU between Ports 1 and 2; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan4")

'OPTIONAL Measure Isolation
MsgBox "Connect LOADS to Port 1 AND Port 2; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan5")

'All standards have been measured. Save the result
Scpi.Execute ("SENS:CORR:COLL:SAVE")
MsgBox "The calibration has been completed"
Perform an Unguided ECal

This VBScript program performs an Unguided Full 2-Port ECal.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Unguided.vbs. Learn how to setup and run the macro.

See Sense:Correction commands.

See other SCPI Examples

```vbscript
Set pna = CreateObject("AgilentPNA835x.Application")
Set scpi = pna.ScpiStringParser
' Preset the analyzer
scpi.Execute "SYSTem:PRESet"

' Start frequency of 10 MHz
scpi.Execute "SENSe:FREQuency:STARt 10E6"

' Stop frequency of 9 GHz
scpi.Execute "SENSe:FREQuency:STOP 9E9"

' Select the preset S11 measurement
scpi.Execute "CALCulate:PARameter:SELect 'CH1_S11_1'"
' Read the information about the Keysight factory
' characterization data of ECal module #1 on the USB bus
module1Info = scpi.Execute("SENSe:CORRection:COLLect:CKIT:INFormation? ECAL1,CHAR0")

' Prompt for the ECal module
MsgBox "Description of ECal Module #1:" & Chr(10) & Chr(10) & module1Info & _Chr(10) & Chr(10) & "Make port connections to the ECal module, then press enter"
' ECal full 1 port and 2 port
' Choose a Calibration Type (comment out one of these)
scpi.Execute "SENSe:CORRection:COLLect:METHod refl3"
scki.Execute "SENSe:CORRection:COLLect:METHod SPARSOLT"
' Specify to have the VNA automatically determine which port of the
' ECal module is connected to which port of the VNA.
scki.Execute "SENSe:CORRection:PREFerence:ECAL:ORIentation ON"
' Alternatively, if you are measuring at very low power levels where
' the VNA fails to sense the module's orientation, you may need to turn
' off the auto orientation and specify how the module is connected (as in
' these next two commented lines of code -- "A1,B2" would indicate Port A
' of the module is connected to Port 1 and Port B is connected to Port 2).
```
Acquire and store the calibration terms. *OPC? causes a "+1" to be returned when finished. CHAR0 indicates to use the Keysight factory characterized data within the ECal module (as opposed to a user characterization).

```python
x = scpi.Execute("SENSe:CORRection:COLLect:ACQuire ECAL1,CHAR0;*OPC?"
      ' Note: if you have set up a slow sweep speed (for example, if you’re using a narrow IF bandwidth), and while this calibration is being acquired you wish to have your program perform other operations (like checking for the click event of a Cancel button) and you’re NOT using the COM ScpiStringParser, you can use the optional ASYNchronous argument with the ACQuire command as shown here below instead of sending that command in the way shown above. The SCPI parser then will return immediately while the cal acquisition proceeds (i.e., the parser will NOT block-and-wait for the cal to finish, so you can send additional commands in the meantime). So you can do "*ESR?" or "*STB?" queries to monitor the status register bytes to see when the OPC bit gets set, which indicates the cal has finished. That type of OPC detection works for all of the VNA’s SCPI parsers except the COM ScpiStringParser.
An alternative to querying the status register, is to setup an SRQ handler if your IO Libraries supports that.
When an SRQ event occurs, a call back will automatically
"SENSe:CORRection:COLLect:ACQuire ECAL1,CHAR0,ASYNchronous;*OPC"
MsgBox "Done with calibration."
Perform Unknown Thru or TRL Cal

The following program performs either a 2-port SOLT Unknown Thru Cal or a 2-port TRL Cal. The 85052C Cal Kit used in this program contains both types of standards. This program can be run on 2-port or 4-port VNAs. When run on select PNA-L models, a Delta Match Cal is required.

- See Delta Match Cal example program
- See the Guided Cal commands

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Unknown.vbs. Learn how to setup and run the macro.

```vbs
Sub PerformUnknownThruOrTRLCal()
    Set pna = CreateObject("AgilentPNA835x.Application")
    Set scpi = pna.ScpiStringParser
    ' Specify connectors for Ports 1 and 2
    scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 female'"
    scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 male'"
    ' If your VNA has 3 or 4 ports, uncomment one or both of
    ' these next two lines, to explicitly specify this is a 2-port cal.
    'scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT3 'Not used'"
    'scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT4 'Not used'"
    ' Specify cal kit for Ports 1 and 2
    ' Since the 85052C cal kit contains SOLT standards and also TRL
    ' standards, these next two lines set cal and thru method.
    ' Always send the init command before and after these two commands
    scpi.Parse "SENS:CORR:COLL:GUID:INIT"
    scpi.Parse "SENS:CORR:COLL:GUID:PATH:CMEThod 1,2,"SOLT"
End Sub
```
scli.Parse "SENS:CORR:COLL:GUID:PATH:TMETHod 1,2,"Undefined Thru"

' To set up the cal as TRL, comment the previous 'CMET' line and uncomment this next line. The TMETHod is set by default
'scli.Parse "SENS:CORR:COLL:GUID:PATH:CMETHod 1,2,"TRL"

' Initiate the calibration
scli.Parse "SENS:CORR:COLL:GUID:INIT"

' Query the list of ports that need delta match
portList = Split(retStr, ",")

' If portList contains just one element and it's value is 0, then that indicates none of the ports being calibrated require delta match data.
' Note: if each testport on the VNA has it's own reference receiver (R channel),
' then delta match is never needed, so portList will always be just 0.
lowerBound = LBound(portList)
If (UBound(portList) <> lowerBound) Or (CInt( portList(lowerBound) ) <> 0) Then

' Delta match data is required for at least one port.
' For this example, we assume a Global Delta Match Cal has previously been performed so the Global Delta Match CalSet exists.
' The Global Delta Match CalSet is used when the APPL command is invoked without a specific calset ID (GUID).
'scli.Parse "SENS:CORR:COLL:GUID:DMAT:APPL"
End If

' Query the number of calibration steps
retStr = scli.Parse("SENS:CORR:COLL:GUID:STEP?"")
numSteps = CInt(retStr)

' Measure the cal standards
For i = 1 To numSteps
retVal = MsgBox(prompt, vbOKCancel)
If retVal = vbCancel Then Exit Sub
retStr = scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN" & CStr(i) & ";*OPC?")
Next
' Compute the error coefficients and save the cal to CalSet, and turn it on
scpi.Parse "SENS:CORR:COLL:GUID:SAVE"
MsgBox "Cal is done!"
End Sub
Setting Up a Phase Noise Measurement

This example program creates a Phase Noise measurement setup.

This VBScript program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as PN_Setup.vbs. Learn how to setup and run the macro.

See Also

- Setting Up a Source
- Spurious Measurement
- Integrated Noise Measurement
- Spot Noise Measurement

See Other SCPI Example Programs

```
'

    ' Phase Noise - basic measurement
    '
    Dim app
    Dim scpi
    ' Create / Get the VNA application.
    Set app = CreateObject("AgilentPNA835x.Application")
    Set scpi = app.ScpiStringParser
    scpi.parse "SYST:PRES"
    ' Create a Phase Noise channel
    scpi.parse "CALC:PAR:DEL:ALL"
    scpi.parse "CALC:MEAS1:DEF 'PN:Phase Noise'"
```
scpi.parse "DISP:MEAS1:FEED 1"
' Set Carrier frequency to 3 GHz
scpi.parse "SENS:PN:SWEep:CARRier:FREQuency 3.0e9"
' Set Start/Stop Offset to 1 kHz and 10 MHz
scpi.parse "SENS:FREQuency:STARt 1e3"
scpi.parse "SENS:FREQuency:STOP 10e6"
' Set RBW Ratio to 5 %
scpi.parse "SENS:PN:BWIDth:RESolution:RATio 5"
' Set FFT Avg Factor to 2
scpi.parse "SENSe:PN:FAVerage:FACTor 2"
' Select Noise Mode to Normal
scpi.parse "SENS:PN:SWEep:NOISe:MODE NORMal"
' Select VNA Input to use for the measurement
scpi.parse "SENS:PN:RECeiver 'b2'"
**Setting Up a Source**

This example program sets up a source for a Phase Noise measurement.

This VBScript program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as PN_Setup.vbs. Learn how to setup and run the macro.

See Also

- Setting Up a Phase Noise Measurement
- Spurious Measurement
- Integrated Noise Measurement
- Spot Noise Measurement

---

```vb
'  Phase Noise - Set up the Source
'
Dim app
Dim scpi
' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
' Set source port1 to 3 GHz, -3 dBm
scpi.parse "SOURce:POWer1:MODE ON"
scpi.parse "SOURce:FREQuency1:FIXed 3e9"
scpi.parse "SOURce:POWer1:LEVel:IMMediate:AMPlitude -3"
```
Spurious Measurement

This example program creates a Phase Noise Spurious measurement.

This VBScript program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as PN_Spurious.vbs. Learn how to setup and run the macro.

See Also

- Setting Up a Phase Noise Measurement
- Setting Up a Source
- Integrated Noise Measurement
- Spot Noise Measurement

See Other SCPI Example Programs

```
'  Phase Noise - Spurious analysis

Dim app
Dim scpi
'  Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
'  Set sort order of spurious list
scpi.parse "CALCulate:MEASure:PN:SPURious:SORT OFFSet"
'  Enable spurious analysis
scpi.parse "CALCulate:MEASure:PN:SPURious:ANALysis:STATe ON"
'  Disable spurious omission
```
' Set sensibility of spurious detection to 2.5
scpi.parse "CALCulate:MEASure:PN:SPURious:SENSibility 2.5"

' Set minimum level of spurious to 140 dBc
scpi.parse "CALCulate:MEASure:PN:SPURious:THReshold:LEVel:MINimum -140.0"

' Set threshold table (***** but this feature is hidden at 1st release of MintP *****)
scpi.parse "CALCulate:MEASure:PN:SPURious:THReshold:TABle:DELete"
scpi.parse "CALCulate:MEASure:PN:SPURious:THReshold:TABle:DATA 1e3,-135.0,42.0,5e3,-145,56"

' Show spot noise table
scpi.parse "DISPlay:WINDow:TABle:SPURious:ENABle ON"

' Query analysis results
s1 = scpi.parse("CALCulate:MEASure:PN:SPUR:DATA?")
Wscript.Echo "Spurs Result  " & s1
Integrated Noise Measurement

This example program creates a Integrated Phase Noise measurement.

This VBScript program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as PN_Integ_Noise.vbs. Learn how to setup and run the macro.

See Also

- Setting Up a Phase Noise Measurement
- Setting Up a Source
- Spurious Measurement
- Spot Noise Measurement

See Other SCPI Example Programs

```
' Phase Noise - Integrated Noise analysis
'
Dim app
Dim scpi
' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
' Set analysis range
' Range1: Full span
' Range2: Custom range from 1.2 MHz to 6 MHz
scpi.parse "CALCulate:MEASure:PN:INTEGRal:RANGE1:TYPE FULL"
```
scpi.parse "CALCulate:MEASure:PN:INTegral:RANGe2:TYPE CUSTom"
scpi.parse "CALCulate:MEASure:PN:INTegral:RANGe2:STARt 1.2e6"
scpi.parse "CALCulate:MEASure:PN:INTegral:RANGe2:STOP 6.0e6"
' Show integrated noise table
scpi.parse "DISPlay:WINDow:TABle:INOise:ENABle ON"
' Query analysis results
s1 = scpi.parse("CALCulate:MEASure:PN:INT:RANGe1:DATA? IPN")
s2 = scpi.parse("CALCulate:MEASure:PN:INT:RANGe1:DATA? RPM")
s3 = scpi.parse("CALCulate:MEASure:PN:INT:RANGe1:DATA? RFM")
s4 = scpi.parse("CALCulate:MEASure:PN:INT:RANGe1:DATA? RMSJ")
s5 = scpi.parse("CALCulate:MEASure:PN:INT:RANGe2:DATA? IPN")
s6 = scpi.parse("CALCulate:MEASure:PN:INT:RANGe2:DATA? RPM")
s7 = scpi.parse("CALCulate:MEASure:PN:INT:RANGe2:DATA? RFM")
s8 = scpi.parse("CALCulate:MEASure:PN:INT:RANGe2:DATA? RMSJ")
Wscript.Echo " Range1" & vbCrLf & _
        "    Integ Noise    " & s1 & _
        "    Residual PM   " & s2 & _
        "    Residual FM   " & s3 & _
        "      Jitter      " & s4 & vbCrLf & _
Wscript.Echo " Range2" & vbCrLf & _
        "    Integ Noise    " & s5 & _
        "    Residual PM   " & s6 & _
        "    Residual FM   " & s7 & _
        "      Jitter      " & s8 & vbCrLf
Spot Noise Measurement

This example program creates a Spot Noise measurement.

This VBScript program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as PN_Spot_Noise.vbs. Learn how to setup and run the macro.

See Also

- Setting Up a Phase Noise Measurement
- Setting Up a Source
- Spurious Measurement
- Integrated Noise Measurement

See Other SCPI Example Programs

'  Phase Noise - Spot Noise
'  
Dim app
Dim scpi
' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
' Enable spot noise analysis
scpi.parse "CALCulate:MEASure:PN:SNOise:STATE ON"
' Enable spot noise of decade frequency
scpi.parse "CALCulate:MEASure:PN:SNOise:DEC:STAT ON"
' Enable spot noise of 120 kHz and 3 MHz offset frequency
scpi.parse "CALCulate:MEASure:PN:SNOise:USER1:STATE ON"
scpi.parse "CALCulate:MEASure:PN:SNOise:USER2:STATE ON"
scpi.parse "CALCulate:MEASure:PN:SNOise:USER1:X 120e3"
scpi.parse "CALCulate:MEASure:PN:SNOise:USER2:X 3e6"
' Show spot noise table
scpi.parse "DISPlay:WINDow:TABLE:SNOise:ENABLE ON"
' Query spot noise data
s1 = scpi.parse("CALCulate:MEASure:PN:SNO:USER1:X?")
s2 = scpi.parse("CALCulate:MEASure:PN:SNO:USER1:Y?")
s3 = scpi.parse("CALCulate:MEASure:PN:SNO:USER2:X?")
s4 = scpi.parse("CALCulate:MEASure:PN:SNO:USER2:Y?")
s5 = scpi.parse("CALCulate:MEASure:PN:SNO:DEC:X?")
s6 = scpi.parse("CALCulate:MEASure:PN:SNO:DEC:Y?")
Wscript.Echo "User1" & vbCrLf & _
   "  X = " & s1 & "  Y = " & s2 & vbCrLf & _
"User2" & vbCrLf & _
   "  X = " & s3 & "  Y = " & s4 & vbCrLf & _
"Decades Edges" & vbCrLf & _
   "  X = " & s5 & "  Y = " & s6
Power Meter Uncertainty

This VBScript program is an example of setting up power uncertainty on a power meter.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Guided.vbs. Learn how to setup and run the macro.

```vbs
' Keysight Technologies 2018

' Uncertainty on power meter example with SCPI

' This script executes and prints the results of all the
' commands related to the uncertainty on power meter

''''''''''''''''''''''''''''''''''''''''
' Create the Application and parser objects

option explicit
dim app, scpi
set app = CreateObject("AgilentPNA835x.Application")
set scpi = app.ScpiStringParser
''''''''''''''''''''''''''''''''''''''''

' query for the ID to verify the communication is established
wscript.echo scpi.execute("*IDN?")

' In the following 'Device2' is name of the used external device

'''''''''''''''''''''''''''''''''''''''' QUERY COMMANDS ''''''''''''''''''''''''''''''''''''''''
```
' Query for the available power meter models
',

dim catalog
wscript.echo catalog
' the return is a csv value
',

' Query for the file path. If the model is NOT Custom Model this function will return
' "Undefined"
',

',

' Query for the pwr mtr model used.
',

' Query for the uncertainty of the pwr meter at 0.GHz and with 10dBm power level
wscript.echo scpi.Execute("SYST:CONF:EDEV:PMAR:UNC:READ? 'Device2',0.1GHz,10.0 ")
',

' Query for the optimum power level (minimum uncertainty) for for the selected
' power meter
',

'''''''''''''''' SETTING COMMANDS ''''''''''''''''
',

' Setting a specific model:
' Setting a custom file

scpi.Execute("SYST:CONF:EDEV:PMAR:UNC:FILE 'Device2', 'C:\Users\Public\Documents\Network Analyzer\UncSensorExample.dat'")
Pulse Narrowband Setup

The following Python example sets up a narrowband pulsed configuration using the integrated pulse measurement option (N52xxA #008 or S93026A/B). The process defines a 500 ns pulse with a 10% duty cycle that is transmitted from port 1 to port 2 of the VNA. It then completes a guided 2 port calibration (unpulsed) with power and then switches to a couple power sweeps at different frequencies and then recalls the base state again. The program uses the non-default pulse generators 3 and 4 to control the gate and modulator drives in order to show the process for using the non-default settings.

See Other SCPI Example Programs

```python
import visa
import time

# start of PNA_Integrated_Pulse_Narrowband_Setup
rm = visa.ResourceManager()
PNA_X = rm.open_resource('TCPIP0::141.121.210.126::hislip0::INSTR')
PNA_X.timeout = 10000
# Preset PNA and wait for completion
PNA_X.write(':SYSTem:PRESet')
temp_values = PNA_X.query_ascii_values('*OPC?')
opc = int(temp_values[0])
PNA_X.write(':CALCulate:PARameter:DELeTe:ALL')
# Setup Measurements that are desired (S21, S12)
PNA_X.write(':CALCulate:PARameter:DEFine:EXTended "%s","s"' % ('MyS21', 'S21'))
PNA_X.write(':CALCulate:PARameter:DEFine:EXTended "%s","s"' % ('MyS12', 'S12'))
PNA_X.write(':CALCulate:PARameter:DEFine:EXTended "%s","s"' % ('MyB', 'B,1'))
PNA_X.write(':CALCulate:PARameter:DEFine:EXTended "%s","s"' % ('MyC', 'C,1'))
```

5399
('MyR1', 'R1,1'))

PNA_X.write(':DISPlay:WINDow1:TRACe1:FEED "%s" % ('MyS21'))
PNA_X.write(':DISPlay:WINDow1:TRACe2:FEED "%s" % ('MyS12'))
PNA_X.write(':DISPlay:WINDow1:TRACe3:FEED "%s" % ('MyB'))
PNA_X.write(':DISPlay:WINDow1:TRACe4:FEED "%s" % ('MyR1'))

# Setup Sweep Frequency and Number of Points
PNA_X.write(':SENSe:FREQuency:STARt %G GHZ' % (1.85))
PNA_X.write(':SENSe:FREQuency:STOP %G GHZ' % (5.55))
PNA_X.write(':SENSe:SWEep:POINts %d' % (9))

# Set PRF to Fixed Mode
PNA_X.write(':SENSe:SWEep:PULSe:PRF:AUTO %d' % (0))

# Set to Standard Pulsed mode and set PW (500 ns) and Period (5 us)
PNA_X.write(':SENSe:SWEep:PULSe:MASTer:WIDTH %G' % (5e-07))
PNA_X.write(':SENSe:SWEep:PULSe:MASTer:PERiod %G' % (5e-06))
PNA_X.write(':SENSe:SWEep:PULSe:MODE %s' % ('STD'))

# Disable Auto IFBW and Set IFBW (10 Hz)
PNA_X.write(':SENSe:SWEep:PULSe:IFBW:AUTO %d' % (0))
PNA_X.write(':SENSe:BANDwidth:RESolution %G' % (10.0))

# Disable Auto Pulse Detect Mode and set mode to Narrowband
PNA_X.write(':SENSe:SWEep:PULSe:DETectmode:AUTO %d' % (0))
PNA_X.write(':SENSe:SWEep:PULSe:WIDeband:STATe %d' % (0))

# Set Auto Timing of pulse width and delays
PNA_X.write(':SENSe:SWEep:PULSe:TIMing:AUTO %d' % (1))
PNA_X.write(':SENSe:SWEep:PULSe:DRIVE:AUTO %d' % (0))

# Get Calculated Default Modulator Drive Info
mod_width = float(PNA_X.query(':SENSe:PULSe1:WIDTh?'))
```python
mod_period = float(PNA_X.query(':SENSe:PULSe1:PERiod?'))
mod_delay = float(PNA_X.query(':SENSe:PULSe1:DELay?'))

# Get Calculated Default Gate Drive Info

gate_width = float(PNA_X.query(':SENSe:PULSe2:WIDTh?'))
gate_period = float(PNA_X.query(':SENSe:PULSe2:PERiod?'))
gate_delay = float(PNA_X.query(':SENSe:PULSe2:DELay?'))

# Setup which pulse gen is for modulation source (Pulse 4) and for IF gates (Pulse 3).

PNA_X.write(':SENSe:PATH:CONFig:ELEMent:STATe "%s","%s"' % ('Src1Out1PulseModEnable', 'Enable'))

PNA_X.write(':SENSe:PATH:CONFig:ELEMent:STATe "%s","%s"' % ('PulseModDrive', 'Pulse4'))

PNA_X.write(':SENSe:PATH:CONFig:ELEMent:STATe "%s","%s"' % ('IFGateA', 'Pulse3'))

PNA_X.write(':SENSe:PATH:CONFig:ELEMent:STATe "%s","%s"' % ('IFGateB', 'Pulse3'))

PNA_X.write(':SENSe:PATH:CONFig:ELEMent:STATe "%s","%s"' % ('IFGateR1', 'Pulse3'))

PNA_X.write(':SENSe:PATH:CONFig:ELEMent:STATe "%s","%s"' % ('IFGateR2', 'Pulse3'))

# Setup Pulse Generators that are being used

PNA_X.write(':SENSe:PULSe4:WIDTh %G' % (mod_width))
PNA_X.write(':SENSe:PULSe3:WIDTh %G' % (gate_width))
PNA_X.write(':SENSe:PULSe4:PERiod %G' % (mod_period))
PNA_X.write(':SENSe:PULSe3:PERiod %G' % (gate_period))
PNA_X.write(':SENSe:PULSe4:DELay %G' % (mod_delay))
PNA_X.write(':SENSe:PULSe3:DELay %G' % (gate_delay))
PNA_X.write(':SENSe:PULSe1:STATe %d' % (0))
PNA_X.write(':SENSe:PULSe2:STATe %d' % (0))
```
PNA_X.write(':SENSe:PULSe3:STATe %d' % (1))
PNA_X.write(':SENSe:PULSe4:STATe %d' % (1))

# Turn off pulsing to calibrate (optional)
PNA_X.write(':SENSe:PATH:CONFig:ELEMENT:STATe "%s","%s"' % ('Src1Out1PulseModEnable', 'Disable'))

# Set port connectors and cal kits
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:CONNector:PORT1:SELeCT "%s"' % ('1.85 mm female'))
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:CONNector:PORT2:SELeCT "%s"' % ('1.85 mm male'))
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:CONNector:PORT3:SELeCT "%s"' % ('Not used'))
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:CONNector:PORT4:SELeCT "%s"' % ('Not used'))
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:CKIT:PORT1:SELeCT "%s"' % ('N4694D ECal MY59410162'))
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:CKIT:PORT2:SELeCT "%s"' % ('N4694D ECal MY59410162'))

# Enable Power Calibration on Port 1
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:PSENsor1:STATe %d' % (1))
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:PSENsor1:CONNector "%s"' % ('Ignored'))

# Set Power Level for Power Calibration step of calibration
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:PSENsor1:POWer:LEVel %G' % (-10.0))

# Set cal thru method to analyzer default
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:METHod %s' % ('DEFAULT'))

# Determine the steps needed to complete the calibration
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:INITiate:IMMediate')
temp_values = 
PNA_X.query_ascii_values(':SENSe:CORRection:COLLect:GUIDed:STEPS?')
steps = int(temp_values[0])
PNA_X.timeout = 90000

# Measure Cal Standards

data = PNA_X.query(':SENSe:CORRection:COLLect:GUIDed:DESCription? %d' % (1))
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:ACQuire %s' % ('STAN1'))
temp_values = PNA_X.query_ascii_values('*OPC?')
opc1 = int(temp_values[0])

data1 = PNA_X.query(':SENSe:CORRection:COLLect:GUIDed:DESCription? %d' % (2))
PNA_X.write(':SENSe:CORRection:COLLect:GUIDed:ACQuire %s' % ('STAN2'))
temp_values = PNA_X.query_ascii_values('*OPC?')
opc2 = int(temp_values[0])
Setup Fast CW and FIFO

This example program does the following:

- Setup an A/R and B/R measurement
- Turn ON point averaging
- Set external edge triggering (commented out)
- Set FIFO and Fast CW
- Write data into FIFO data buffer
- Read FIFO data buffer

**IMPORTANT** - Because the IFBW is set to 600 kHz, data will NOT be sent to the FIFO after each acquisition. Learn more.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as FIFO.vbs. Learn how to setup and run the macro.

See the SCPI FIFO commands.

```
Dim returnStr
Dim app
Dim p
Dim start
Dim complete
Dim init
Dim finished
' Create / Get the VNA application.
```
Set app = CreateObject("AgilentPNA835x.Application")
Set p = app.ScpiStringParser

sub Write (command)
    if len(returnStr) <> 0 then
        err.Raise 55,"Write","Query Unterminated"
    end if
    returnStr = p.parse(command)
end sub

sub WriteIgnoreError(command)
    returnStr = p.Execute(command)
p.Parse("SYST:ERR?") ' clear error queue
end sub

function Read
    if len(returnStr) = 0 then
        err.Raise 55,"Read","Bad read"
    end if
    Read = returnStr
    returnStr = ""
end function

' Setup and measure A/R and B/R
Write "SYST:FPRESET"
Write "DISP:WIND ON"
Write "CALC:PAR:DEF:EXT 'meas1','A/R1,0'"
Write "DISP:WIND:TRACE:FEED 'meas1'"
Write "CALC:PAR:DEF:EXT 'meas2','B/R1,0'"
Write "DISP:WIND:TRACE2:FEED 'meas2'"
' Set IFBW to 600 khz (400 thousand pts/second)
Write "SENS:BWID:RES 600khz"

' Point Averaging Count = 1
Write "SENS:AVER:MODE POINT"
Write "SENS:AVER ON"
Write "SENS:AVER:COUNT 1"

' Edge triggering - positive edge
'Write "CONT:SIGN BNC1,TIEPOSITIVE"
'Write "TRIG:SOUR EXT"
'Write "SENS:SWE:TRIG:POIN ON"

' Setup FIFO and Fast CW count
Write "SENS:SWE:MODE HOLD"
Write "SYST:FIFO ON"
Write "SYST:FIFO:DATA:CLEAR"
Write "SENS:SWE:TYPE CW"
Write "SENS:SWE:TYPE:FACW 1000000" ' set the point count to 1 million
Write "SENS:SWE:MODE SING" ' start an asynchronous acquisition.
init = now()

' Gather data
'wait until end of sweep. Timeout needs to be very large here.
Write "*OPC?" '
opcCount = Read()
Dim points
Write "SYST:FIFO:DATA:COUNT?"
points = Read()
msgbox points

' points == 2000000 ' points = 2million. Took 5 seconds to acquire
For I = 0 to 1 ' 2 iterations (2 parameters * 2 sets of 1 million)
Dim data

Write "SYST:FIFO:DATA? 1000000"

Data = Read()

Next

' turn FIFO and FastCW OFF

Write "SYST:FIFO OFF"
Write "SENS:SWE:TYPE:FACW 0"

finished = now()

msgbox "Init =" & init & vbCrLf & "Done =" & finished
This VBScript program does the following:

- Preset the VNA
- Return active channel number and measurement string
- Create a marker
- Set X-axis value
- Read X, Y-axis values
- Set marker to trace Min
- Read X, Y-axis values

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Markers.vbs. Learn how to setup and run the macro.

See all Marker SCPI commands.

---

**See Other SCPI Example Programs**

```vbs
Dim na, vi, ret

Set na = CreateObject("AgilentPNA835x.Application")
Set vi = na.ScpiStringParser

'Get Identification String from Analyzer
ret=vi.Parse("*IDN?")
msgbox ret

'Preset VNA
ret=vi.Parse("SYST:PRES; *OPC?")

'Get Active Channel and Measurement
```
chan = vi.Parse("SYST:ACT:CHAN?")
meas = vi.Parse("SYST:ACT:MEAS?")

' Convert chan to a single number
chan=CStr(CInt(chan))

'Select Active Measurement
vi.Parse "CALC" + chan + ":PAR:SEL " + meas

'Turn Marker 1 on and set X value to 1 GHz
vi.Parse "CALC" + chan + ":MARK1:STAT ON"
vi.Parse "CALC" + chan + ":MARK1:X 1e9"

' Get X and Y marker values
x_val = vi.Parse("CALC" + chan + ":MARK1:X?")
y_val = vi.Parse("CALC" + chan + ":MARK1:Y?")

'Display Marker Values
msgbox "X Value = " + x_val + Chr(10) + "Y Value = " + y_val

' Use Marker 1 as a minimum search
vi.Parse "CALC" + chan + ":MARK1:FUNC:EXEC MIN"

' Get X and Y marker values
x_val = vi.Parse("CALC" + chan + ":MARK1:X?")
y_val = vi.Parse("CALC" + chan + ":MARK1:Y?")

'Display Marker Values
msgbox "X Value = " + x_val + Chr(10) + "Y Value = " + y_val
This program demonstrates how to change source and receive ports when measuring noise figure. It assumes that option 029 ("Fully Corrected Noise Figure") is installed.

If only option 028 ("Noise figure measurements using standard receivers") is installed, switching ports is simpler, since only one noise receiver selection is available.

This program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as NF.vbs. Learn how to setup and run the macro.

See the Noise figure commands.

```vbs
option explicit
dim app, hostname, parser
set app = CreateObject("Agilentpna835x.application")
set parser = app.ScpiStringParser

' Create Noise Figure measurement
parser.Parse "*RST"
parser.Parse "CALC:PAR:DEL:ALL"
parser.Parse "CALC:CUST:DEF 'NF', 'Noise Figure Cold Source', 'NF' "
parser.Parse "DISP:WIND:TRAC1:FEED 'NF'"

'To change from the default input/output port settings of
'source port = VNA1, receive port = VNA2, you must first
'change the noise receiver, then select the desired ports.
dim srcPort, rcvPort

' Set source=VNA port 3 and receiver=VNA port 4
srcPort = 3
rcvPort = 4

' use VNA receiver for noise measurements
```
set port mapping

To revert back to using the noise receiver, the receive port must
be set to the default value VNA2 BEFORE switching to the noise receiver.

Otherwise, the SCPI error
"The noise receiver is not available for the selected receive port"
will occur.

use dedicated noise receiver for noise measurements
Setup Phase Control

This VBScript program configures and displays Phase Sweep measurements.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as RxLev.vbs. Learn how to setup and run the macro.

See Also

Phase Control SCPI commands

About Phase Control

See Other SCPI Example Programs

'Assume port 1 is connected to port 3

Set pna = CreateObject("AgilentPNA835x.Application")
Set SCPI = PNA.ScpiStringParser

'Create 3 trace S33, R3/C(amp),R3/C(phase)

SCPI.Parse("SYST:FPR")
SCPI.Parse("DISP:WIND:STATE ON")
SCPI.Parse("CALC:PAR:DEF 'MyMeas1',S33")
SCPI.Parse("DISP:WIND1:TRAC1:FEED 'MyMeas1'")
SCPI.Parse("CALC:PAR:SEL 'MyMeas1'")
SCPI.Parse("CALC:FORM SMIT")
SCPI.Parse("CALC:PAR:DEF 'MyMeas2',R3C,3")
SCPI.Parse("DISP:WIND1:TRAC2:FEED 'MyMeas2'")
SCPI.Parse("CALC:PAR:SEL 'MyMeas2'")
SCPI.Parse("CALC:FORM MLOG")
SCPI.Parse("CALC:PAR:DEF 'MyMeas3',R3C,3")
SCPI.Parse("DISP:WIND1:TRAC3:FEED 'MyMeas3'")

SCPI.Parse("CALC:PAR:SEL 'MyMeas3'")

SCPI.Parse("CALC:FORM PHAS")

SCPI.Parse("SENS:SWE:TYPE PHAS")

'turn on 3 and 1

SCPI.Parse("SOUR:POW1:MODE ON")

SCPI.Parse("SOUR:POW3:MODE ON")

' set port3's control parameter to R3/C

SCPI.Parse("SOUR:PHAS3:PAR 'R3/C'")

' Set port3 to PAR mode

SCPI.Parse("SOUR:PHAS3:PAR:MODE PAR")

SCPI.Parse("SOUR:PHAS3:PAR:PORT 1")

SCPI.Parse("SOUR:PHAS3:POFF:FIX 3")

SCPI.Parse("SOUR:PHAS3:STAR 0")

SCPI.Parse("SOUR:PHAS3:STOP 180")
Setup PNOP and PSAT Marker Search

This example program does the following:

- Sets up measurement for either PNOP or PSAT marker search
- Sets parameters for search
- Reads a parameter for each

See PNOP and PSAT SCPI commands.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as SearchMkr.vbs. Learn how to setup and run the macro.

```vbs
Dim app
Set app = CreateObject("AgilentPNA835X.Application")
Dim scpi
set scpi = app.ScpiStringParser
scpi.Execute ("SYST:FPReset")
' View Power Out vs Power In
' Create and turn on window/channel 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")
' Define a measurement name, parameter
scpi.Execute ("CALCulate1:PARameter:DEFine:EXT 'MyMeas','B,1'")
' Associate ("FEED") the measurement name ("MyMeas") to WINDOW (1)
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")
s pci.Execute ("CALCulate1:PARameter:SELect 'MyMeas'")
```
'perform power sweep
scpi.Execute ("SENSe1:SWEep:TYPE POWer")
scpi.Execute ("SOURce1:POWer:STARt -5")
scpi.Execute ("SOURce1:POWer:STOP 0")
'-------------------
'Choose marker search
resp=Msgbox ("PNOP (yes) or PSAT (no)" , 4, "PNA Marker Search Demo")
if resp=6 then
    PNOP1()
Else
    PSAT1()
End If
'-------------------
'PSAT marker search
Sub PSAT1()
    scpi.Execute ("CALCulate1:MARKer:PSATuration:BACKoff 2")
    'Read PSAT Parameter
dim answer
    answer=scpi.Execute ("CALCulate1:MARKer:PSATuration:GAIN?")
    wscript.echo("Gain Sat: ", answer)
End Sub
'-------------------
'PNOP marker search
Sub PNOP1()
    scpi.Execute ("CALCulate1:MARKer:PNOP:BACKoff 2")
    scpi.Execute ("CALCulate1:MARKer:PNOP:POFFset 1")
    'Read PNOP Parameter
dim answer
answer = scpi.Execute("CALCulate1:MARKer:PNOP:GAIN?")

wscript.echo("PNOP Gain: " & answer)

End Sub
This VBScript program configures Receiver Leveling.

- Preset the VNA
- Make all receiver leveling settings

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as RxLev.vbs. Learn how to setup and run the macro.

See all Recevier Leveling SCPI commands.

---

```vbscript
Set pna = CreateObject("AgilentPNA835x.Application")
Set SCPI = pna.ScpiStringParser
  'set source port
dim srcP
srcP = "1"
  'Preset PNA
SCPI.Parse "SYST:PRES"
SCPI.Parse "sour1:pow" + srcP + "-15"
SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:ref 'R1'"  
SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:tol 0.02"
SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:iter 10"
SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:fast OFF"
SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:ifbw 100"
SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:offs 0"
```
SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:safe:max 20"

SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:safe:min -100"

SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:safe ON"

'Last, enable receiver leveling

SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec ON"
Setup Sweep Parameters using SCPI

This Visual Basic program sets up sweep parameters on the Channel 1 measurement. To run this program, you need:

- An established GPIB interface connection

See Other SCPI Example Programs

```vba
GPIB.Write "SYST:PRESET"
' Select the measurement
GPIB.Write "CALC:PAR:SEL 'CH1_S11_1'"
' Set sweep type to linear
GPIB.Write "SEN:SWE:TYPE LIN"

' Set IF Bandwidth to 700 Hz
GPIB.Write "SEN:BAND 700"

' Set Center and Span Freq's to 4 GHz
GPIB.Write "SEN:FREQ:CENT 4ghz"
GPIB.Write "SEN:FREQ:SPAN 4ghz"

' Set number of points to 801
GPIB.Write "SEN:SWE:POIN 801"

' Set sweep generation mode to Analog
GPIB.Write "SEN:SWE:GEN ANAL"

' Set sweep time to Automatic
GPIB.Write "SEN:SWE:TIME:Auto ON"

' Query the sweep time
SweepTime = GPIB.Read
```
Setup the Display using SCPI

This Visual Basic program:

- Sets data formatting
- Turns ON the Trace, Title, and Frequency Annotation
- Autoscales the Trace
- Queries Per Division, Reference Level, and Reference Position
- Turn ON and set averaging
- Turn ON and set smoothing

To run this program, you need:

- An established GPIB interface connection

```
GPIB.Write "SYSTem:PRESet"

'Select the measurement
GPIB.Write "CALCulate:PARameter:SELect 'CH1_S11_1'"

'Set the Data Format to Log Mag
GPIB.Write "CALCulate1:FORMat MLOG"

'Turn ON the Trace, Title, and Frequency Annotation
GPIB.Write "DISPlay:WINDow1:TRACe1:STATe ON"
GPIB.Write "DISPlay:WINDow1:TITLe:STATe ON"
GPIB.Write "DISPlay:ANNotation:FREQuency ON"

'Autoscale the Trace
GPIB.Write "DISPlay:WINDow1:TRACe1:Y:Scale:AUTO"

'Query back the Per Division, Reference Level, and Reference Position
GPIB.Write "DISPlay:WINDow1:TRACe1:Y:SCALe:PDIvision?"
Pdiv = GPIB.Read
GPIB.Write "DISPlay:WINDow1:TRACe1:Y:SCALe:RLEvel?"
Rlev = GPIB.Read
GPIB.Write "DISPlay:WINDow1:TRACe1:Y:SCALe:RPOSition?"
Ppos = GPIB.Read

'Turn ON, and average five sweeps
```
GPIB.Write "SENSe1:AVERage:STATe ON"
GPIB.Write "SENSe1:AVERage:Count 5"

'Turn ON, and set 20% smoothing aperture
GPIB.Write "CALCulate1:SMoothing:STATe ON"
GPIB.Write "CALCulate1:SMoothing:APERture 20"
Show Custom Cal Windows during a Guided Calibration

This VBScript program shows how to send commands that allow you to view specific 'custom' windows, and sweep specific channels, during a UI (Cal Wizard) or remote calibration.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as CalWindow.vbs. Learn how to setup and run the macro.

These commands are used to show and sweep windows and channels:

- **SENS:CORR:COLL:DISP:WIND**
- **SENS:CORR:COLL:SWE:CHAN**
- **SENS:CORR:COLL:DISP:WIND:AOFF**
- **SENS:CORR:COLL:SWE:CHAN:AOFF**
- **SENS:CORR:COLL:GUID:PACQuire**

See Other SCPI Example Programs

```
Dim app
Dim scpi

' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' A comment
'Preset the analyzer
'This creates an S11 measurement in channel 1, window 1
scpi.Execute "SYST:PRest"

' Create and turn on window 2
```
scpi.Execute "DISPlay:WINDow2:STATE ON"

'Define an S21 measurement in channel 2
scpi.Execute "CALCulate2:PARameter:DEFine:EXT 'MyMeas',S21"

'Associate ("FEED") the measurement name ('MyMeas') to WINDow2
'and give the new TRACe a number (1).
scpi.Execute "DISPlay:WINDow2:TRACe1:FEED 'MyMeas'"

'The following lines are all you need in order to:
'show and sweep the custom Cal windows during a UI Calibration
'If sending ONLY these commands, make sure you know the
'correct window and channel numbers to show and sweep.
'Flag windows 1 and 2 to show during Ch1 calibration
scpi.Execute "SENS:CORR:COLL:DISP:WIND1 ON"
scpi.Execute "SENS:CORR:COLL:DISP:WIND2 ON"

'Flag channels 1 and 2 to sweep during Ch1 calibration
scpi.Execute "SENS1:CORR:COLL:SWE:CHAN1 ON"
scpi.Execute "SENS1:CORR:COLL:SWE:CHAN2 ON"

'=================================================================

'The following code performs a remote guided Cal on Ch1.
'From a remote cal, the Cal window does not normally show and sweep
'after the previous standard has been acquired.
'This shows how to include the PACQuire (preview) to view and sweep the Cal Window.
'The Custom window also shows and sweeps due to the flag commands above.
'The flags are cleared at the end of this section.

'Specify the DUT connectors
scpi.Execute "sens:corr:coll:guid:conn:port1 ""APC 3.5 female"" "
scpi.Execute "sens:corr:coll:guid:conn:port2 ""APC 3.5 male"" "
' Select the Cal Kit for each port being calibrated.
scpi.Execute "sens:corr:coll:guid:ckit:port1 ""85052D"" "
scpi.Execute "sens:corr:coll:guid:ckit:port2 ""85052D"" "
' Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"
MsgBox "Number of steps is " + CStr(numSteps)
' Measure the standards
For i = 1 to numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
' send the Preview Acquire command, then prompt
scpi.Execute "sens:corr:coll:guid:PACquire STAN" + CStr(i)
' Do NOT send any Guided Cal commands here or the cal window will not sweep
MsgBox strPrompt, vbOKOnly, step
scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
Next
' Conclude the calibration
scpi.Execute "sens:corr:coll:guid:save"
MsgBox "Cal is done!"
'
'Remove the Custom Window flags
scpi.Execute "SENS:CORR:COLL:DISP:WIND:AOFF"
'Remove the channel sweep flags
Perform a Sliding Load Calibration using GPIB

This Visual Basic program does a **only** the sliding load portion of a Calibration. To run this program, you need:

- An established GPIB interface connection
- A measurement and calibration routine to call this sub-program
- STAN3 set up as a sliding load standard

### See Other SCPI Example Programs

```vba
Sub slide()
    'Measure the sliding load for at least 5 and no more than 7 slides
    'Note that "SLSET" and "SLDONE" must be executed before the actual acquisition of a slide
    MsgBox "Connect Sliding Load; set to Position 1; then press OK"
    GPIB.Write "SENS:CORR:COLL SLSET"
    GPIB.Write "SENS:CORR:COLL STAN3;"
    MsgBox "Set Sliding Load to position 2; then press OK"
    GPIB.Write "SENS:CORR:COLL SLSET"
    GPIB.Write "SENS:CORR:COLL STAN3;"
    MsgBox "Set Sliding Load to position 3; then press OK"
    GPIB.Write "SENS:CORR:COLL SLDONE"
    GPIB.Write "SENS:CORR:COLL STAN3;"
End Sub
```
Dim app

Set app = CreateObject("AgilentPna835x.application","hostname")
Set scpi = app.ScpiStringParser
scpi.Parse("SYST:PRESET")
scpi.Parse("calc:par:del:all")
scpi.Parse("calc:cust:define "ENR","Noise Figure Cold Source","ENR""
scpi.Parse("disp:wind:trac1:feed "ENR""
scpi.Parse("SENSe:FREQ:Y:STOP 8.5E+9")
scpi.Parse("SENSe:NOISE:USBSource:SELECT "U1831C MY12345678""
scpi.Parse("SENSe:NOISE:USBSource?")
scpi.Parse("SENSe:NOISE:USBSource:TEMPerature? "U1831C MY12345678""
scpi.Parse("SENSe:NOISE:ENR INTERNAL")
scpi.Parse("SENSe:NOISE:ENR:FILENAME?")
scpi.Parse("SENSe:NOISE:CAL:METHOD "Scalar""
scpi.Parse("SENSe:NOISE:CAL:METHOD?")
scpi.Parse("SENSe:CORR:COLL:GUID:INIT")
scpi.Parse("SENSe:CORR:COLL:GUID:STEP?")
scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN1")
scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN2")
scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN3")
scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN4")
scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN5")
scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN6")
scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN7")
scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN8")
Socket Client

The following C# example demonstrates how to send SCPI commands to the VNA via a TCP socket connection and how to use a TCP 'control' connection. If the command is a query, the program will read the instrument's response. You can add or replace the SCPI commands in this program with your own.

Learn how to enable Sockets communication on the VNA.

For both of the following methods, first copy the example text below into a Notepad file and name it SocketClient.cs.

To run using Microsoft Visual Studio 2003 or 2005

1. From the Visual Studio File menu, select New, then Project.

2. In the New Project window, select the following items (noting the location of the file folder it is creating for you) then click OK.

   - Project Type: Visual C#
   - Template: Console Application
   - Project Name: SocketClient

1. Copy SocketClient.cs into the folder that was created in the previous step.

2. In the Solution Explorer window pane, right-click Class1.cs (if Visual Studio 2003) or Program.cs (if Visual Studio 2005). Select Delete to delete that file.

3. In the Solution Explorer, right-click SocketClient, and select Add, then Existing Item....

4. Browse to select SocketClient.cs and click OK.

You should then be able to build the project, and test the resulting SocketClient.exe from a command prompt (shell) window.

To run using Mono

Mono is a cross-platform version of .NET. You can download a free version of Mono at http://www.mono-project.com. Once downloaded and installed:

1. Run the Mono command prompt (shell) window.
2. Navigate to the directory where the example SocketClient.cs is stored.

3. Type: **MCS SocketClient.cs** (builds the .exe and saves in that same folder.)

4. Type **mono SocketClient.exe** <VNA name or IP address>

This example was compiled and tested successfully with Mono version 1.1.13. It was run on a PC using the Red Hat version 9.0 distribution of the Linux operating system. It was also run on a PC using Windows XP. This program has not been tested with other versions of Mono, or on other operating systems.

**To run with Keysight T&M Toolkit**

Keysight T&M Toolkit 2.0 is the first version to support communication using Sockets.

Use the following to address the Sockets port: **TCPIP0::<VNA name or IP address>::5025::SOCKET**

```csharp
using System;
using System.Net;
using System.Net.Sockets;

// This C# "Console Application" example program demonstrates SCPI
// communication with a Keysight TCP socket-enabled instrument that
// supports socket "control connections" (such as VNA network analyzers,
// which have support for control connections in their socket
// implementation as of VNA Firmware A.08.33.01).
namespace CSharpSocketClient
{
    /// <summary>
    /// The class supporting the main entry point for the application.
    /// </summary>
    class MainClass
    {
        ...
    }
}
```
static AsyncCallback m_pCallbackFunc;
static string m_AsyncReply;

/// <summary>
/// The main entry point for the application.
/// </summary>

[STAThread]
static int Main(string[] args)
{
    try
    {
        if (args.Length != 1)
        {
            Console.WriteLine(""");
            Console.WriteLine("Usage -- with Microsoft's .NET runtime:");
            Console.WriteLine("SocketClient servernameoraddress");
            Console.WriteLine("Example: SocketClient 192.168.0.1");
            Console.WriteLine("");
            Console.WriteLine("Usage -- with Mono's (www.mono-project.com) .NET runtime:");
            Console.WriteLine("mono SocketClient.exe servernameoraddress");
            Console.WriteLine("Example: mono SocketClient.exe 192.168.0.1");
            return 1;
        }

        string server = args[0];
        Int32 port = 5025; // default socket port number for the VNA
/ Create the primary client socket instance
    Socket client = new Socket(AddressFamily.InterNetwork, SocketType.Stream,
                             ProtocolType.Tcp);

    // Get the DNS IP addresses associated with the instrument.
    // (if 'server' string contains the IP address rather than DNS name, this
    // still works)
    IPHostEntry hostInfo = Dns.Resolve(server);
    IPAddress[] IPaddresses = hostInfo.AddressList;
    if (IPaddresses.GetLength(0) < 1)
        return 1;

    // Create an endpoint to use for opening the socket connection
    IPEndPoint endpoint1 = new IPEndPoint(IPaddresses[0], port);

    // Open the connection to the server instrument
    client.Connect(endpoint1);
    if(!client.Connected)
        return 1;

    // Query the instrument's ID string.
    string id = Parse(client, "*IDN?");

    // Clear the instrument's Status Byte
    Parse(client, "*CLS");

    // Enable for the OPC bit (bit 0, which has weight 1) in the instrument's
    // Event Status Register, so that when that bit's value transitions from 0
to 1
    // then the Event Status Register bit in the Status Byte (bit 5 of that
    byte)
// will become set.
Parse(client, "*ESE 1");

// Enable for bit 5 (which has weight 32) in the Status Byte to generate an
// SRQ when that bit's value transitions from 0 to 1.
Parse(client, "*SRE 32");

// Ask the instrument for the number of a port on which a 'control'
// socket connection can be opened.
string controlPortNumStr = Parse(client, "SYSTem:COMmunicate:TCPip:CONTrol? ");

Int32 controlPortNum = System.Convert.ToInt32(controlPortNumStr);

// Create the client "control connection" socket instance
Socket controlClient = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);

// Create an endpoint to use for opening the control connection
IPEndPoint endpoint2 = new IPEndPoint(IPaddresses[0], controlPortNum);
// Connect to the server instrument via the port number that was returned
// by the instrument.
controlClient.Connect(endpoint2);
if(!controlClient.Connected)
    return 1;

// Start the control connection listening for an SRQ message.
BeginListeningForAsyncReply(controlClient);

// Now send a preset command to the instrument, accompanied by 'OPC' such
Parse(client, "SYSTem:PRESet;*OPC");

// Normally at this point you would want to have your program do other things
// right here until the SRQ callback occurs, instead of just idling here waiting
// for it.

do {} while (m_AsyncReply == null);

// Now that the SRQ has occurred, we can issue a Device Clear via the control connection.
Parse(controlClient, "DCL");

// The instrument will respond back with "DCL" (and linefeed character appended
// on the end) via the control connection when it has finished processing the
// Device Clear request. Note that this 'Response' method uses the synchronous
// form of 'Receive', so it could potentially time out if the instrument were
// to take a long time to process the Device Clear. So alternatively the
// 'BeginListeningForAsyncReply' could be used for this instead of 'Response'.
string deviceClearResponse = Response(controlClient);

// Close both of the socket client sessions.
controlClient.Close();
client.Close();
}
catch (ArgumentNullException e)
{
    Console.WriteLine("ArgumentNullException: {0}", e);
}
catch (SocketException e)
{
    Console.WriteLine("SocketException: {0}", e);
}

Console.WriteLine("/n Press Enter to continue...");
Console.Read();
return 0;
}

static string Parse(Socket client, string command)
{
    // Translate the passed command into ASCII and store it as a Byte array.
    Byte[] data = System.Text.Encoding.ASCII.GetBytes(command);
    // Send the command to the socket-enabled instrument.
    client.Send(data);
    // Has to be followed by a linefeed character as terminator.
    Byte[] lf = {(Byte)'/n'};
    client.Send(lf);
    Console.WriteLine("Sent: {0}", command);
    // If the message was a query (involved a question mark), receive the instrument response.
    if (command.IndexOf("?") >= 0)
    {

return Response(client);

}
return "";
}

static string Response(Socket client)
{
    // Buffer to store the response bytes.
    // For simplicity of this example, we allocate just for a 256-byte maximum
    // response size.
    Byte[] data = new Byte[256];
    // Read the batch of response bytes.
    Int32 byteCount = client.Receive(data);
    // String to store the response ASCII representation.
    string responseData = System.Text.Encoding.ASCII.GetString(data, 0, byteCount);
    Console.WriteLine("Received: {0}", responseData);
    return responseData;
}

static void BeginListeningForAsyncReply(Socket client)
{
    if (m_pCallbackFunc == null)
    {
        m_pCallbackFunc = new AsyncCallback(OnMessageReceived);
    }
    SocketPacket socPkt = new SocketPacket();
    socPkt.thisSocket = client;
// Start asynchronously listening for a response from this client

IAAsyncResult result = client.BeginReceive (socPkt.data,
0, socPkt.data.Length,
SocketFlags.None,
m_pCallbackFunc,
socPkt);
}

class SocketPacket
{
    public Socket thisSocket;

    // For simplicity of this example, we allocate just for a 256-byte maximum response size.
    public Byte[] data = new Byte[256];
}

static void OnMessageReceived (IAsyncResult asyn)
{
    SocketPacket socPkt = (SocketPacket)asyn.AsyncState;
    Int32 byteCount = socPkt.thisSocket.EndReceive (asyn);
    m_AsyncReply = System.Text.Encoding.ASCII.GetString (socPkt.data, 0, byteCount);
    Console.WriteLine("Received: {0}", m_AsyncReply);
}
}
Create a Spectrum Analyzer Measurement

This example program creates a Spectrum Analyzer measurement setup.

This VBScript program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as SA.vbs. Learn how to setup and run the macro.

See the Spectrum Analyzer commands.

See Other SCPI Example Programs

`' Demonstration of basic Spectrum Analyzer measurement setup.`

```vbscript
set pna=CreateObject("AgilentPNA835x.Application","hostname")
set scpi = pna.ScpiStringParser

CreateSAMeasurement

SetupLinearSweep

ConfigureAdvancedSettings

' Create a Spectrum Analyzer measurement

Sub CreateSAMeasurement

' Create a B measurement on channel 1

scpi.Parse "SYST:FPR"

scpi.Parse "DISP:WIND ON"

scpi.Parse "CALC:CUST:DEF 'sa_meas', 'Spectrum Analyzer', 'B'"

scpi.Parse "DISP:WIND:TRAC:FEED 'sa_meas'"

scpi.Parse "CALC:PAR:SEL 'sa_meas'"

' Set frequency range

scpi.Parse "SENS:FREQ:CENTER 3 GHz"

scpi.Parse "SENS:FREQ:SPAN 2 GHz"

' Center frequency step size
```

5438
' Set to Auto mode with SENS:FREQ:CENTER:STEP:AUTO ON
scpi.Parse "SENS:FREQ:CENTER:STEP:SIZE 20 MHz"
'
' RBW filter shape
' Choices are GAUSsian|FLATtop|KAISer|BLACKman|NONE
scpi.Parse "SENS:SA:BAND:SHAPE KAIS"
'
' RBW and VBW values
scpi.Parse "SENS:SA:BAND:RES 100 kHz"
scpi.Parse "SENS:SA:BAND:VID 10 kHz"
'
' Detector type
' Choices are AVERage|SAMPLE|PEAK|NORMAL|NEGPeak|PSAMple|PAverage
scpi.Parse "SENS:SA:DET:FUNC PEAK"
'
' Video averaging type
' Choices are POWer|LOG|VOLTage|VMAX|VMIN
'
' RBW/VBW and Span/RBW ratios
scpi.Parse "SENS:SA:BAND:VID:RAT 1.23"
'
' ADC Filter
' Choices are 11MHz|38MHz
' Enable auto mode with SENS:SA:ADC:FILT:AUTO.
scpi.Parse "SENS:SA:ADC:FILT:38MHz"
End Sub
'
' Configure a Spectrum Analyzer measurement for Linear sweep mode on Port 1.
Sub SetupLinearSweep
' Turn Port 1 ON
`scpi.Parse "SOURCE:POW:MODE ON"
``
  ' Set Port 1 sweep type to Linear
```
`scpi.Parse "SENS:SA:SOURCE1:SWEEP:TYPE LIN"
```
  ' Set start and stop frequencies
```
`scpi.Parse "SENS:SA:SOURCE1:FREQ:START 2E9"
```
`scpi.Parse "SENS:SA:SOURCE1:FREQ:STOP 4E9"
```
  ' Set 'Source Number of Steps'. This is the number of frequencies to use between start and stop (inclusive).
```
  ' This setting is channel-wide.
```
```
  ' Set 'SA Sweeps per Source Steps'. This is the number of sweeps to take at each measurement frequency.
```
  ' This setting is also channel-wide.
```
```
End Sub
```
  ' Configure a few of the Advanced Settings for SA.
```
Sub ConfigureAdvancedSettings
```
  ' Set the 'Image Reject' selection.
```
  ' Choices are MIN MAX NORM NLOW NHIGH
```
`scpi.Parse "SENS:SA:IMAGE:REJ MAX"
```
  ' Enable display of ImageReject traces.
```
`scpi.Parse "SENS:SA:TRACE:IMAGE:STATE ON"
```
  ' Enable point mode.
```
  ' This forces the number of display points to match the FFT point count.
```
`scpi.Parse "SENS:SA:DET:BYPASS ON"
```
End Sub
See Other SCPI Example Programs

Status Reporting using SCPI

This Visual Basic program demonstrates two methods of reading the analyzer’s status registers:

- **Polled Bit Method** - reads the Limit1 register continuously.
- **SRQ Method** - enables an interrupt of the program when bit 6 of the status byte is set to 1. The program then queries registers to determine if the limit line failed.

To run this program, you need:

- An established GPIB interface connection
- A form with two buttons: Poll and SRQ Method
- A means of causing the limit line to fail, assuming it passes initially.

```vba
Private Sub Poll_Click()
    ' POLL THE BIT METHOD
    ' Clear status registers
    GPIB.Write "*CLS"

    'Loop FOREVER
    Do
        DoEvents
        GPIB.Write "STATus:QUEStionable:LIMit1:EVENt?"
        onn = GPIB.Read
    Loop Until onn = 2

    MsgBox "Limit 1 Failed "
End Sub

Private Sub SRQMethod_Click()
    ' SRQ METHOD
    GPIB.Write "SYSTem:PRESet"
    GPIB.Write "CALCulate:PARameter:SELect 'CH1_S11_1'"
    ' slow down the trace
    GPIB.Write "SENS:BWID 150"

    ' Setup limit line
    GPIB.Write "CALC:LIMIT:DATA 2,3e9,6e9,-2,-2"'
    GPIB.Write "CALC:LIMIT:DISP ON"
    GPIB.Write "CALC:LIMIT:STATe ON"

    ' Clear status registers.
```
GPIB.Write "*CLS;*wai"
' Clear the Service Request Enable register.
GPIB.Write "*SRE 0"
' Clear the Standard Event Status Enable register.
GPIB.Write "*ESE 0"

' Enable questionable register, bit(10) to report to the status byte.
GPIB.Write "STATus:QUEStionable:ENABle 1024"

' Enable the status byte register bit3 (weight 8) to notify controller
GPIB.Write "*SRE 8"

' Enable the onGPIBNotify event
GPIB.NotifyMask = cwGPIBRQS
GPIB.Notify
End Sub

----------------------------------------------------
Private Sub GPIB_OnGPIBNotify(ByVal mask As Integer)
' check to see what failed
' was it the analyzer?
GPIB.Write "*STB?"
onn = GPIB.Read
If onn <> 0 Then
' If yes, then was it the questionable register?
  GPIB.Write "STATus:QUEStionable:EVENT?"
  onn = GPIB.Read
  ' Determine if the limit1 register, bit 8 is set.
  If onn = 1024 Then
    ' if yes, then was it trace 1?
    GPIB.Write "STAT:QUES:LIMIT1:EVENT?"
    onn = GPIB.Read
    If onn = 2 Then MsgBox ("Limit Linel Failed")
  End If
End If
End If
End Sub
TDR/TDT Measurement

This example program demonstrates how to perform TDR/TDT setup written in Excel VBA (VISA-COM).

See the TDR commands.

See Other SCPI Example Programs

```vba
Dim rm As VisaComLib.ResourceManager
Dim vna As VisaComLib.FormattedIO488

Sub Message(msg As String)
    Dim NumDmy As Integer
    'Syncronize to VNA
    vna.WriteString "*OPC?"
    NumDmy = vna.ReadNumber
    'Write Message
    MsgBox msg, vbOKOnly
End Sub

Sub TDRTDTMeasure()
    On Error GoTo errorhandler
    Set rm = New VisaComLib.ResourceManager
    Set vna = New VisaComLib.FormattedIO488

    ' Set VNA Address
    Set vna.IO = rm.Open("TCPIP0::K-N5232B-40046::hislip0::INSTR")
    vna.IO.Timeout = 90000
End Sub
```
' Clear Excel Sheet Cells
Range("D5:F5").ClearContents

With vna
    Dim i As Integer

    ' Set DUT Topology to Differential 2-Port
    .WriteString "CALC:TDR:DEV DIF2"

    ' Desckew and Loss Compensation

    ' Desckew
    .WriteString "SENSe:CORR:TDR:EXTension:AUTO:IMMediate", True
    .WriteString "*OPC?", True
    NumDmy = .ReadNumber
    ' Loss Compensation, thru
    MsgBox "Connect thru between ports 1 and 3.", vbOKOnly
    .WriteString "SENSe:CORR:TDR:COLL:DLC:THRU 1,3", True
    .WriteString "*OPC?", True
    NumDmy = .ReadNumber

    MsgBox "Connect thru between ports 2 and 4.", vbOKOnly
    .WriteString "SENSe:CORR:TDR:COLL:DLC:THRU 2,4", True
    .WriteString "*OPC?", True
    NumDmy = .ReadNumber
' Loss Compensation, thru
MsgBox "Connect a Load to the ports 1,2,3,4.", vbOKOnly
.WriteString "*OPC?", True
NumDmy = .ReadNumber
.WriteString "*OPC?", True
NumDmy = .ReadNumber
.WriteString "*OPC?", True
NumDmy = .ReadNumber
.WriteString "*OPC?", True
NumDmy = .ReadNumber

' Save the data to finish the Deskew and Loss Compensation
.WriteString "*OPC?", True
NumDmy = .ReadNumber

' Set Rise Time
.WriteString "CALC:PAR:COUN?" 'Get number of traces
For i = 1 To .ReadNumber
    .WriteString ":CALC:TDR:MEAS" + CStr(i) + ":TIME:STEP:RTIM:THR T2_8" 'Threshold
.WriteString ":CALC:TDR:MEAS" + CStr(i) + ":TIME:STEP:RTIM 50e-12" 'Rise Time

Next i

' Measure
Message "Connect DUT to cables." + vbCrLf + "Press OK to execute measurement."

.WriteString ":SENS:TDR:SWE:SING;*OPC?"
NumDmy = .ReadNumber
' Auto Scale for all traces
.WriteString ":DISP:TDR:SCAL:AUTO"

' Read Rise Time of Tr 5 (Tdd21)
Cells(5, 4) = "Rise Time"
.WriteString ":CALC:TDR:MEAS5:TTIM:STAT ON"
.WriteString ":CALC:TDR:MEAS5:TTIM:THR T2_8"
.WriteString ":CALC:TDR:MEAS5:TTIM:DATA?"
Cells(5, 6) = .ReadNumber

' Restore TDR GUI
.WriteString ":DISP:TDR:MIN:STAT OFF"
Message "Finished TDT/TDT Measurement."

End With

vna.IO.Close
Exit Sub
errorhandler:

End Sub
Simulated Eye Diagram

This example program demonstrates how to perform a simulated eye diagram written in Excel VBA (VISA-COM).

See the TDR commands.

See Other SCPI Example Programs

```vba
Dim rm As VisaComLib.ResourceManager
Dim vna As VisaComLib.FormattedIO488
Sub Message(msg As String)
    Dim NumDmy As Integer
    ' Syncronize to VNA
    vna.WriteString "*OPC?"
    NumDmy = vna.ReadNumber
    ' Write Message
    MsgBox msg, vbOKOnly
End Sub
Sub SimEyeDiagram()
    On Error GoTo errorhandler
    Set rm = New VisaComLib.ResourceManager
    Set vna = New VisaComLib.FormattedIO488

    ' Set your VNA address
    Set vna.IO = rm.Open("TCPIP0::K-N5232B-40046::hislip0::INSTR")
    vna.IO.Timeout = 90000
End Sub
```
' Clear Excel Sheet Cells

Range("A1:F22").ClearContents

With vna

    Dim i As Integer

    ' Set DUT Topology to Differential 2-Port
    .WriteString ":CALC:TDR:DEV DIF2"

    ' Execute Deskew using TDR Setup Wizard
    .WriteString ":DISP:TDR:MIN:STAT OFF"

    Message "[Deskew]" + vbCrLf + vbCrLf + vbCrLf + vbCrLf + vbCrLf
        "1. Press 'TDR Setup Wizard' on TDR GUI." + vbCrLf + vbCrLf + vbCrLf
        "2. Select 'Deskew' and complete the wizard with 'Finish' button." + vbCrLf + vbCrLf
        "3. Confirm 'TDR [Deskew]' indicator appears on VNA status line." + vbCrLf + vbCrLf
        "4. Press OK button to continue."

    ' Minimize TDR GUI
    .WriteString ":DISP:TDR:MIN:STAT ON"

    ' Set Rise Time
    .WriteString "CALC:PAR:COUN?" 'Get number of traces

    For i = 1 To .ReadNumber
        .WriteString "CALC:TDR:MEAS" + CStr(i) + vbCrLf
        .WriteString ":TIME:STEP:RTIM:THR T2_8" 'Threshold
    Next
".TIME:STEP:RTIM 50e-12" 'Rise Time

Next i

' Set Bit Pattern Parameters
.WriteString ":CALC:TDR:EYE:INP:BPAT:TYPE PRBS" 'Type
.WriteString ":CALC:TDR:EYE:INP:BPAT:LENG 7" 'Length
.WriteString ":CALC:TDR:EYE:INP:OLEV 200e-3" 'One Level
.WriteString ":CALC:TDR:EYE:INP:DRAT 1e9" 'Data Rate

' Set Rise Time/Threshold
.WriteString ":CALC:TDR:EYE:INP:RTIM:DATA 50e-12"
.WriteString ":CALC:TDR:EYE:INP:RTIM:THR T2_8"

' Activate Trace 5 (Tdd21)
.WriteString ":CALC:PAR:MNUM:SEL 5"

' Execute Draw Eye
Message "Connect DUT to cables." + vbCrLf + "Press OK to draw Eye diagram."

.WriteString ":CALC:TDR:EYE:STAT ON"
.WriteString ":CALC:TDR:EYE:EXEC"

' Read Eye Results
.Cells(3, 4) = "Eye Results"

Dim EyeResult() As Double

Dim Labels() As Variant

.WriteString ":CALC:TDR:EYE:RES:DATA?"

EyeResult() = .ReadList(ASCIIType_R8, ",")

For i = 0 To 17 'Number of results = 18
    Cells(i + 5, 4) = Labels(i)
    Cells(i + 5, 6) = EyeResult(i)

Next i

' Restore TDR GUI
.WriteString ":DISP:TDR:MIN:STAT OFF"

Message "Finished Simulated Eye Diagram."

End With

vna.IO.Close

Exit Sub

errorhandler:

End Sub
2 Channel Measurement

This example program demonstrates a 2 channel measurement that allows you to make TDR measurement on channel 1 and a more customized S-parameter measurement on channel 2. This is a program example of the 2 Channel Measurement Example.

This program example is written in Excel VBA (VISA-COM).

See the TDR commands.

See Other SCPI Example Programs

Dim rm As VisaComLib.ResourceManager
Dim vna As VisaComLib.FormattedIO488
Private Sub Message(msg As String)
    Dim NumDmy As Integer
    ' Syncronize to VNA
    vna.WriteString "*OPC?"
    NumDmy = vna.ReadNumber
    ' Write Message
    MsgBox msg, vbOKOnly
End Sub
Private Sub SetupCh1Measurement()
    On Error GoTo errorhandler
    With vna
        ' Execute ECal using TDR Setup Wizard
        .WriteString "*:DISP:TDR:MIN:STAT OFF"
        Message "[Full Calibration (ECal)]" + vbCrLf + _
        "1. Press 'TDR Setup Wizard' on TDR GUI." + vbCrLf + _
"2. Select 'Full Calibration (ECal)' and complete the wizard." + vbCrLf + _

"3. Confirm 'TDR [Full]' indicator appears on VNA status line." + vbCrLf + _

"4. Press OK button to continue."

' Minimize TDR GUI
currentState = "DISP:TDR:MIN:STAT ON"

' Set Rise Time/Threshold
For i = 1 To 8
    .WriteString "CALC:TDR:MEAS" + CStr(i) + "TIME:STEP:RTIM:THR T2_8"
    .WriteString "CALC:TDR:MEAS" + CStr(i) + "TIME:STEP:RTIM 50e-12"
Next i

' Setup Target Trace
currentState = "CALC:TDR:MEAS1:PAR TDD11" 'Parameter
currentState = "CALC:TDR:MEAS1:FORM IMP" 'Format

' Setup limit line
.currentState = "CALC:MEAS1:LIM:DATA 1,0,1e-9,105,105,2,0,1e-9,75,75"
.currentState = "CALC:MEAS1:LIM ON"
.currentState = "CALC:MEAS1:LIM:DISP ON"
End With
Exit Sub
errorhandler:
End Sub

Private Sub ExecCh1Measurement()
    On Error GoTo errorhandler

    Dim TimeData() As Double, Impedance() As Double
    Dim Nop As Integer, PassFail As Integer, i As Integer, k As Integer

    With vna
        ' Exec Trigger and Synchronize
        .WriteString ":SENS:TDR:SWE:SING;*OPC?"
        NumDmy = .ReadNumber
        ' Auto Scale
        .WriteString ":DISP:TDR:SCAL:AUTO"
        ' Get Number Of Points
        .WriteString ":SENS1:SWE:POIN?"
        Nop = .ReadNumber
        ReDim TimeData(Nop - 1)
        ReDim Impedance(Nop - 1)
        ' Get X-axis data
        .WriteString ":CALC:MEAS1:X:VAL?"
        TimeData() = .ReadList(ASCIIType_R8, ",")
        ' Get Y-axis(Impedance) data
        .WriteString ":CALC:MEAS1:DATA:FDAT?"
Impedance() = .ReadList(ASCIIType_R8, ",")
' Get limit line test result
.WriteString ":CALC:MEAS1:LIM:FAIL?"
PassFail = .ReadNumber
' Write Results
Cells(4, 4) = "CH1"
Cells(6, 4) = "Time"
Cells(6, 5) = "Data"
For i = 0 To Nop - 1
    Cells(i + 7, 4) = TimeData(i)
    Cells(i + 7, 5) = Impedance(i)
Next i
If PassFail = 1 Then
    Cells(4, 5) = "FAILED"
Else
    Cells(4, 5) = "PASSED"
End If
End With
Exit Sub

errorhandler:
    MsgBox Err.Description, vbExclamation, "Error Occurred",
End Sub

Private Sub SetupCh2Measurement()
    On Error GoTo errorhandler
    With vna
' Create new measurement in CH2
.WriteString "CALC:PAR:MNUM:SEL 8" 'Activate the last trace
.WriteString "DISP:WIND5:STAT ON" 'Create new window
.WriteString "CALC2:PAR:DEF:EXT 'ch2_meas','S11'"
.WriteString "DISP:WIND5:TRAC9:FEED 'ch2_meas'"

' Setup stimulus
.WriteString "SENS2:FREQ:STAR 1e9"
.WriteString "SENS2:FREQ:STOP 3e9"
.WriteString "SENS2:SWE:POIN 201"
.WriteString "SENS2:BWID 1e3"

' Execute 4-port ECAl using 2-port ECAl module
Dim ecalCmd As String
calCmd = "SENS2:CORR:COLL:GUID:"
For i = 1 To 4
    .WriteString calCmd + "CONN:PORT" + CStr(i) + " 'APC 3.5 female'
    .WriteString calCmd + "CKIT:PORT" + CStr(i) + " 'N4691-60006 ECAl'
Next i
.WriteString calCmd + "INIT"
.WriteString calCmd + "STEPS?"
nSteps = .ReadNumber
For i = 1 To nSteps
    .WriteString calCmd + "DESC? " + CStr(i)
Private Sub ExecCh2Measurement()

    On Error GoTo errorhandler

    Dim FreqData() As Double, InsersionLoss() As Double
    Dim Nop As Integer, PassFail As Integer, i As Integer, k As Integer
With vna

' Exec Trigger and Synchronize
.WriteString ":SENS2:SWE:MODE SING;*OPC?"
NumDmy = .ReadNumber

' Get Number of Points
.WriteString ":SENS2:SWE:POIN?"
Nop = .ReadNumber

' Get X-axis data
ReDim FreqData(Nop - 1)
.WriteString ":CALC2:MEAS9:X:VAL?"
FreqData() = .ReadList(ASCIIType_R8, ",")

' Get Y-axis (Sdd21) data
ReDim InsersionLoss(Nop - 1)
.WriteString ":CALC2:MEAS9:DATA:FDATA?"
InsersionLoss() = .ReadList(ASCIIType_R8, ",")

' Get limit line test result
.WriteString ":CALC2:MEAS9:LIM:FAIL?"
PassFail = .ReadNumber

' Write Results
.Cells(4, 7) = "CH2"
.Cells(6, 7) = "Freq."
.Cells(6, 8) = "Data"
For i = 0 To Nop - 1
.Cells(i + 7, 7) = FreqData(i)
.Cells(i + 7, 8) = InsersionLoss(i)
Next i
If PassFail = 1 Then
    Cells(4, 8) = "FAILED"
Else
    Cells(4, 8) = "PASSED"
End If
End With
Exit Sub

errorhandler:
End Sub

Sub TwoChannelsMeasurement()
    On Error GoTo errorhandler
    Set rm = New VisaComLib.ResourceManager
    Set vna = New VisaComLib.FormattedIO488

    ' Set VNA Address
    Set vna.IO = rm.Open("TCPIP0::K-N5232B-40046::hislip0::INSTR")
    vna.IO.Timeout = 90000

    ' Clear Excel Sheet Cells
    Range("A1:H10010").ClearContents
    With vna
        'Set DUT Topology first
        .WriteString ":CALC:TDR:DEV DIF2"
' Setup Measurement
Call SetupCh1Measurement
Call SetupCh2Measurement
'
' Exec Measurement
Message "Connect DUT to cables." + vbCrLf + "Press OK to execute measurement."
Call ExecCh1Measurement
Call ExecCh2Measurement
'
'Restore TDR GUI
.WriteString "+DISP:TDRE:MIN:STAT OFF"
Message "Finished 2 Channels Measurement."
End With

vna.IO.Close
Exit Sub
errorehandler:
End Sub
Transfer Data using GPIB

The following RMB examples transfer data to and from a remote PC using the MMEM:TRANsfer command.

Transferring data FROM the VNA -- TO a remote PC:

30  !
40  ! Set up I/O paths
50  !
60  ! Network analyzer address
70  ASSIGN @Na TO 716
75  !
77  ! File to be stored on local computer
80  ! First time -- need to create the file.
90  ! After file name, number records set to 0 (ignored by WinOS)
95  ! Use "PURGE" command to delete if desired.
100 CREATE "mytestdata.s2p",0
110 ASSIGN @File TO "mytestdata.s2p"
120  !
122  ! TRANSFER the data (download)
123  !
125  ! Analyzer has file 'testdata.s2p' in default directory
130 OUTPUT @Na;":MMEM:TRAN? ":"testdata.s2p"
135  !
137  ! Now read the bytes coming back from the analyzer in four steps
138  ! (1) Read and dump the first character - '#'
140  ENTER @Na USING ":,A";A$
141  !
142  ! (2) Read the next character which is the number of digits in the file size
145  ENTER @Na USING ":,A";Digit$
146  !
149  ! (3) Use the value of the number of digits to read back the file byte size
150  ! Create query string using this number of digits
155  Img$="",&Digit$&"A"
159  !
160  ! Byte$ holds the number of bytes in string format
160  ENTER @Na USING Img$;Byte$
161  !
163  ! (4) Read the file contents into a buffer and store the buffer contents to a local file
165  ! Allocate a buffer for holding the data
168  ALLOCATE Dat$[VAL(Byte$)]
170  !
172  ! Set up a different image for filling the buffer
175  Img$=Byte$&"A"
178  !
180  ! Retrieve the actual file data
185  ENTER @Na USING Img$;Dat$
305  !
307  ! Now save the file locally.
310  OUTPUT @File;Dat$
320  END

Transferring data FROM the remote PC - TO the VNA:

40  ! Set up I/O paths
50  !
60  ! Network analyzer address
70  ASSIGN @Na TO 716
77  ! File to be retrieved from local computer
78  ASSIGN @File TO "mytestdata.s2p"
79  !
120  !
122  ! TRANSFER the data
123  !
230  ! Allocate a buffer for holding the data
240  ALLOCATE Dat$[26236]
250  !
260  ! Get data from the file and fill Dat$
270  ENTER @File;Dat$
280  !
325  ! Data to be transferred to analyzer file 'testupld.s2p'
326  ! in default directory.
327  ! A specific block transfer designator must follow the
328  ! file name:
329  ! '7' specifies a block transfer.
330  ! '6' specifies 6 digits to follow.
331  ! '026236' matches the buffer size allocated above
332  ! not counting <NL><END> (new line and end of file).
430  OUTPUT @Na;":MMEM:TRAN ""testupld.s2p"",#6026236",Dat$
520  END
Triggering the Analyzer using SCPI

To understand how to trigger the analyzer using SCPI, it is very important to understand the trigger model. Here is a very simple explanation. These three separate functions control triggering:

1. **Trigger:Source** - Where the trigger signals originate:
   - Internal Continuous
   - Internal Manual (Single)
   - External - a trigger source that is connected to the rear panel.

2. **Trigger:Scope** - what gets triggered:
   - Global - each signal triggers all channels in turn.
   - Channel - each signal triggers ONE channel.

3. **Channel settings (Sense<ch>:Sweep:Mode)** How many triggers will each channel accept before going into hold.
   - HOLD - channel will not trigger.
   - CONTinuous - channel triggers indefinitely.
   - GROups - channel accepts the number of triggers specified with the last SENS:SWE:GRO:COUN <num>.
   - SINGle - channel accepts ONE trigger, then goes to HOLD.
   - Point trigger SENS1:SWE:TRIG:POINt

When controlling the VNA using SCPI, a SINGLE trigger is used to ensure that a complete sweep is taken. This example demonstrates how to Single trigger the VNA using the following two methods:

- **Simplest Triggering**
  - This method uses the default Trigger Source = Internal to send a stream of trigger signals.
  - The channel is configured to ACCEPT only a single trigger signal, then HOLD (Sense<ch>:Sweep:Mode SINGle). This is the ONLY required command.
This method can also be used when an External trigger source sends a continuous stream of trigger signals.

**Advanced Triggering**

- This method SENDS a single trigger from the Source, which can be from either Internal (using INIT:IMM) or External triggering.
- Each channel is configured to accept an unlimited number of triggers. This method is the only way to perform point triggering.
- When you require some channels to accept continuous triggers and other channels to accept single triggers, see INIT:IMM Advanced to learn how.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the analyzer. To do this, copy the following code into a text editor file such as Notepad and save it on the analyzer hard drive as Trigger.vbs. Learn how to setup and run the macro.

**Measurement setup example:** This section of code can be used at the start of both methods. It sets up:

- S11 traces on two channels
- 10 data points
- Sweep time of 2 seconds - this is slow enough to allow us to watch as each trace is triggered.

```vbs
Dim app
Dim scpi
' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
'============================================
'Setup the VNA
'Preset the analyzer
scpi.Execute ("SYST:FPReset")
```
' Create and turn on window/channel 1
scpi.Execute("DISPlay:WINDow1:STATE ON")
' Define a measurement name, parameter
scpi.Execute("CALCulate1:PARameter:DEFine:EXT 'MyMeas1',S11")
' Associate ("FEED") the measurement name ("MyMeas") to WINDow (1)
scpi.Execute("DISPlay:WINDow1:TRACe1:FEED 'MyMeas1'")
' Create and turn on window/channel 2
scpi.Execute("DISPlay:WINDow2:STATE ON")
' Define a measurement name, parameter
scpi.Execute("CALCulate2:PARameter:DEFine:EXT 'MyMeas2',S11")
' Associate ("FEED") the measurement name ("MyMeas") to WINDow (2)
scpi.Execute("DISPlay:WINDow2:TRACe2:FEED 'MyMeas2'")

' Set slow sweep so we can see
scpi.Execute("SENS1:SWE:TIME 2")
scpi.Execute("SENS2:SWE:TIME 2")
' set number of points to 10
scpi.Execute("SENS1:SWE:POIN 10")
scpi.Execute("SENS2:SWE:POIN 10")

'=============================================================================

' Put both channels in Hold
scpi.Execute("SENS1:SWE:MODE HOLD")
scpi.Execute("SENS2:SWE:MODE HOLD")

'=============================================================================

' Pick Single Send or Single Accept
resp=Msgbox("Single Send? - Click No for Single Accept", 4, "PNA Trigger Demo")
If resp=6 Then
SingleSend()
Else
Simple Triggering  The following example sends a continuous stream of trigger signals and each
VNA channel is set to ACCEPT only a signal trigger signal, then HOLD.

- This example can be used to configure External triggering where the trigger source sends a continuous
  stream of trigger signals. Configure the type of trigger signal that the VNA responds to using the
  CONTrol:SIGNal command. The command in this example sets the VNA to respond to HIGH TTL signals at
  the rear-panel BNC1 trigger IN connector. This command also automatically sets Trigger Source to External
  Trigger.

- The TRIG SCOPE (Global or Channel) setting is NOT necessary with a continuous stream of trigger signals.
  The example program directly controls when each channel is triggered.

- Point triggering can NOT be used with a continuous stream of trigger signals because in point triggering the
  channel will accept as many triggers as necessary to complete ONE full sweep. Use the single SEND
  example for point triggering.

Sub SingleAccept()

'VNA sends continuous trigger signals
scpi.Execute ("TRIG:SOUR IMMEDIATE")

'Uncomment the following to set External triggering
'scpi.Execute ("CONT:SIGN BNC1,TILHIGH")

AcceptOne()

End Sub

Sub AcceptOne()

'The following command makes the channel immediately sweep
'*OPC? allows the measurement to complete before the controller sends another
command
scpi.Execute ("SENS1:SWE:MODE SINGLe;*OPC?")

' You could do something to ch2 here before sweeping it
scpi.Execute ("SENS2:SWE:MODE SINGLe;*OPC?")
Advanced Trigger This example section performs Single Send triggering. Here, single triggering is accomplished by SENDING one trigger signal from the Trigger source and each channel is setup to accept unlimited trigger signals. See the `INIT:IMM` command for more details.

- Using this method, it is possible to change Trigger:Scope to Global or Channel. Set trigger scope to channel if there is some code to execute between channel measurements. Similarly, this method can be used to set Point triggering. Use this method if there is some code to execute between data point measurements.

- In addition, this method can also be used to perform External triggering if the external trigger source is capable of SENDING single triggers. See the `CONT:SIGNal` command to set the type of signal to which the VNA will respond.

- If the external source can only send a continuous stream of trigger signals, then the Single Accept section must be used.

```vba
Sub SingleSend()
    'Set Source Internal - Manual Triggering
    scpi.Execute("TRIG:SOUR MANual")
    'If using an External trigger source that is capable of
    'sending SINGLE trigger signals, then uncomment the following.
    'This command automatically sets trigger source to External
    'scpi.Execute("CONT:SIGN BNC1,TILHIGH")
    'Setup Trigger Scope
    'WHAT gets triggered
    'Pick one using comments
    'Set Channel triggering
    'scpi.Execute("TRIG:SCOPe CURRent")
End Sub
```
'Set Global triggering (Default)
scpi.Execute ("TRIG:SCOPE ALL")

'Set Channel Settings
'The channels respond to UNLIMITED trigger signals (Default)
scpi.Execute ("SENS1:SWE:MODE CONTinuous")
scpi.Execute ("SENS2:SWE:MODE CONTinuous")

'To do Point trigger on one or more channels, uncomment the following.
'Point trigger automatically sets Trig:Scope to Current/Channel
'scpi.Execute ("SENS1:SWE:TRIG:POINT ON")
'scpi.Execute ("SENS2:SWE:TRIG:POINT ON")

IntTrig()
End Sub

Sub IntTrig()
'If External triggering, replace this Sub with code
to single trigger the External Trig Source
Dim resp
'*OPC? allows the measurement to complete before the controller sends another command
scpi.Execute ("INITiate:IMMediate;*OPC?")
resp=Msgbox ("Another trigger?", 1, "PNA Trigger Demo")
If resp=1 Then
IntTrig()
End If
End Sub
Perform an Unguided Cal on Multiple Channels

This VBScript program performs an Unguided Calibration simultaneously on two channels.

This could be used in the following cases:

- If you need more than the current number of data points per trace, so the additional points must be added to a different channel.
- If you need several channels with independent settings, but you want to calibrate all channels with a minimal number of standard connections. This would be especially critical for on wafer calibration.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Unguided.vbs. Learn how to setup and run the macro.

```vbs
Dim app
Dim scpi
Dim NumberOfActiveChannels
NumberOfActiveChannels = 2
' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
' Query the list of connectors that the VNA system recognizes
scpi.Execute("SYST:PRES")
'Wait for successful preset before continuing
done=scpi.Execute("*OPC?")
'The following section sets up 2 channels with different frequency ranges
scpi.Execute("DISP:WIND1:STATE OFF")
'Reset Windows
scpi.Execute("DISP:WIND1:STATE ON")
```
scpi.Execute("DISP:WIND2:STATE ON")

' Assign a measurement to the first window
scpi.Execute("CALC1:PAR:DEF:EXT 'Meas1', S21")
scpi.Execute("DISP:WIND1:TRAC1:FEED 'Meas1'")

'Assign a measurement to the second window
scpi.Execute("CALC2:PAR:DEF:EXT 'Meas2', S21")
scpi.Execute("DISP:WIND2:TRAC1:FEED 'Meas2'")

'Set up two channels with independent parameters
scpi.Execute("SENS1:FREQ:SPAN 1e9")
scpi.Execute("SENS2:FREQ:SPAN 1e6")

'Wait for changes before continuing
done=scpi.Execute("*OPC?")

'This section sets the calibration kits for channel 1 and channel 2
'Select a trace from channel 1 and set calibration type and cal kit
scpi.Execute("CALC1:PAR:SEL 'Meas1'")
scpi.Execute("SENS1:CORR:COLL:METH SPARSOLT")
scpi.Execute("SENS1:CORR:COLL:CKIT 2") '85056D for default settings

'Same standards for forward and reverse direction
scpi.Execute("SENS1:CORR:TST OFF")

'Select a trace from channel 2 and set calibration type and cal kit
scpi.Execute("CALC2:PAR:SEL 'Meas2'")
scpi.Execute("SENS2:CORR:COLL:METH SPARSOLT")
scpi.Execute("SENS2:CORR:COLL:CKIT 2") '85056D for default settings

'Same standards for forward and reverse direction
scpi.Execute("SENS2:CORR:TST OFF")
'Set both channels to manual triggering
scpi.Execute("INIT1:CONT OFF")
scpi.Execute("INIT2:CONT OFF")
'
'The following assumes female port connector on port 1
' and male port connector on port 1
'Step through all active channels and calibrate and measure all standards.
scpi.Execute("SENS1:CORR:SFOR ON")  'Set acquisition to forward
scpi.Execute("SENS2:CORR:SFOR ON")  'Set acquisition to forward
MsgBox("Connect OPEN standard to port 1")
For CurrentChannel = 1 To NumberOfActiveChannels
    scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":"")
    scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan1")
done= scpi.Execute("*OPC?")
Next

MsgBox("Connect SHORT standard to port 1")
For CurrentChannel = 1 To NumberOfActiveChannels
    scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":"")
    scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan2")
done=scpi.Execute("*OPC?")
Next

MsgBox("Connect LOAD standard to port 1")
For CurrentChannel = 1 To NumberOfActiveChannels
    scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":"")
    scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan3")
done=scpi.Execute("*OPC?")
Next
scpi.Execute("SENS1:CORR:SFOR OFF")  'Set acquisition to reverse
scpi.Execute("SENS2:CORR:SFOR OFF")  'Set acquisition to forward

MsgBox("Connect OPEN standard to port 2")
For CurrentChannel = 1 To NumberOfActiveChannels
    scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":')
    scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan1")
    done=scpi.Execute("*OPC?")
Next

MsgBox("Connect SHORT standard to port 2")
For CurrentChannel = 1 To NumberOfActiveChannels
    scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":')
    scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan2")
    done=scpi.Execute("*OPC?")
Next

MsgBox("Connect LOAD standard to port 2")
For CurrentChannel = 1 To NumberOfActiveChannels
    scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":')
    scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan3")
    done=scpi.Execute("*OPC?")
Next

' Measure thru standard for all channels in both forward and reverse direction
MsgBox("Connect THRU between ports 1 and 2")
scpi.Execute("SENS1:CORR:SFOR ON")  'Set acquisition to forward
scpi.Execute("SENS2:CORR:SFOR ON") 'Set acquisition to forward

For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":"")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan4")
done=scpi.Execute("*OPC?")
Next

scpi.Execute("SENS1:CORR:SFOR OFF") 'Set acquisition to reverse
scpi.Execute("SENS2:CORR:SFOR OFF") 'Set acquisition to reverse

For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":"")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan4")
done=scpi.Execute("*OPC?")
Next

For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":"")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL:SAVE")
done=scpi.Execute("*OPC?")
Next

' Set both channels to continuous triggering
scpi.Execute("INIT1:CONT ON")
scpi.Execute("INIT2:CONT ON")
Upload and Download a Segment List

This VBScript program creates two segments, then uploads the segment data to the VNA.

The second part downloads the segment list from the VNA.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Unguided.vbs. Learn how to setup and run the macro.

See all Segment SCPI commands.

Create and Upload a Segment List

```vbscript
Option Explicit

Dim app

Set app = CreateObject("AgilentPNA835x.Application")

' Preset the VNA
app.Preset

Dim scpi

Set scpi = app.ScpiStringParser

' In case of a measurement receiver VNA like N5264B
' which has no source ports, "SOURce:CATalog?" will
' return an empty list (just a pair of quotation marks)

Dim srcPortNames

srcPortNames = Split( scpi.Execute("SOURce:CATalog?") , "," )

Dim numberOfSrcPorts

If Left( srcPortNames(0) , 2 ) = Chr(34) & Chr(34) Then
    numberOfSrcPorts = 0
Else
    numberOfSrcPorts = UBound(srcPortNames) + 1
```
End If

' Building up a string consisting of the sweep segment data
' we want to set up. This example will create two segments.
Dim segData

' Set state of first segment to be ON (1 = ON, 0 = OFF),
' 101 points, start freq of 10 MHz, stop freq of 1 GHz
segData = "1,101,10E6,1E9"

' If you want to include one or more of: IF bandwidth, Dwell Time
' or Port Power, remove the comments from these next two lines
'TurnOnOptions 1 ' Call the subroutine
'segData = AddOptionalSettings(segData, numberOfSrcPorts)

' Set state of second segment to be ON, 201 points,
' start freq of 1 GHz, stop freq of 3 GHz
segData = segData & ",1,201,1E9,3E9"

' Uncomment this line below only if you uncommented the
' AddOptionalSettings line above for the first segment.
'segData = AddOptionalSettings(segData, numberOfSrcPorts)

Const numSegs = 2

' Upload our segment list to the channel
scpi.Execute "SENSe1:SEGMent:LIST STOP," & numSegs & ",," & segData

' Set segment sweep type on Channel 1
scpi.Execute "SENSe1:SWEep:TYPE SEGment"

' Having the VNA display the segment sweep table for the channel
scpi.Execute "DISPlay:WINDow1:TABLE SEGment"

Sub TurnOnOptions(ByVal chan)
    scpi.Execute "SENSe" & chan & ":SEGment:BWIDth:CONTrol ON"
    scpi.Execute "SENSe" & chan & ":SEGment:SWEep:TIME:CONTrol ON"
    scpi.Execute "SENSe" & chan & ":SEGment:POWer:CONTrol ON"
End Sub
' Turning off coupling allows power to vary per each port
scpi.Execute "SOURce"&chan&":POWer:COUPle OFF"

End Sub

Function AddOptionalSettings(ByVal inStr, ByVal numSrcPorts)
' Specifying 1 kHz IF bandwidth and Dwell Time of 0
inStr = inStr & ",1E3,0"
' -10 dBm power for each of the source ports
Dim i
For i = 0 To numSrcPorts - 1
    inStr = inStr & ",-10"
Next
AddOptionalSettings = inStr
End Function

Download a Segment List

This example assumes that the active trace is in Window 1

Option Explicit

Dim app
Set app = CreateObject("AgilentPNA835x.Application")

Dim scpi
Set scpi = app.ScpiStringParser

' Set the display-active channel's sweep type to segment sweep
' (if the VNA's currently active measurement window doesn't
' contain any traces, this querying for active channel will
' result in a SCPI error which scpi.Parse will trap and throw)
Dim chan
chan = CLng( scpi.Parse("SYSTem:ACTive:CHANnel?") )
scpi.Execute "SENSe"&chan&":SWEep:TYPE SEGment"
' Having the VNA display the segment sweep table for the channel
scpi.Execute "DISPlay:WINDow1:TABLE SEGMen" 

' Get the total number of segments
Dim numSegs
numSegs = CLng( scpi.Execute("SENSe"&chan&":SEGMen:COUNt?"))

' Read the segment listing
Dim segDataStr
segDataStr = scpi.Execute("SENSe"&chan&":SEGMen:LIST?"

Dim segData
segData = Split(segDataStr, ",""

' Get upper bound of the array of data values
' (lower bound of array resulting from VB 'Split' function is 0)
Dim segArrayUB
segArrayUB = UBound(segData)

Dim numDataElementsPerSeg
numDataElementsPerSeg = (segArrayUB + 1) / numSegs

WScript.Echo "Number of segments = " & numSegs
WScript.Echo "Number of data values per segment = " & numDataElementsPerSeg

Dim segInfStr
segInfStr = "Segment 1: state = " & CBool(segData(0))
segInfStr = segInfStr & ", num points = " & CLng(segData(1))
segInfStr = segInfStr & ", start freq = " & CDbl(segData(2))
segInfStr = segInfStr & ", stop freq = " & CDbl(segData(3))
segInfStr = segInfStr & ", IFBW = " & CDbl(segData(4))
segInfStr = segInfStr & ", dwell time = " & CDbl(segData(5))

' In case of a measurement receiver VNA like N5264B
' which has no source ports, "SOURce:CATalog?" will
' return an empty list
Dim srcPortNames
srcPortNames = Split( scpi.Execute("SOURce&chan&":CATalog?") , "","")
Dim srcPortNamesUB
srcPortNamesUB = UBound(srcPortNames)
' First source port name will be preceded by a quotation mark
' and the last name will be followed by one of those, so stripping
' those off now.
srcPortNames(0) = Right( srcPortNames(0), Len(srcPortNames(0)) - 1 )
srcPortNames(srcPortNamesUB) = Left( srcPortNames(srcPortNamesUB),
InstrRev(srcPortNames(srcPortNamesUB), Chr(34)) - 1 )
Dim firstPortIndex
firstPortIndex = 6
Dim lastPortIndex
lastPortIndex = numDataElementsPerSeg - 1
Dim j
For j = firstPortIndex To lastPortIndex
    segInfStr = segInfStr & ", " & srcPortNames(j - firstPortIndex) & " power = " & CDbl(segData(j))
Next
WScript.Echo segInfStr

Example in Excel VBA with VISA-COM

Sub SampleSegmentSetup()
    '*** The variables of the resource manager and the instrument I/O are declared.
    Dim ioMgr As VisaComLib.ResourceManager
    Dim GPIB As VisaComLib.FormattedIO488
    '
    '*** The memory area of the resource manager and the instrument I/O are acquired.
    Set ioMgr = New VisaComLib.ResourceManager
    Set GPIB = New VisaComLib.FormattedIO488
    '
    '*** Open the instrument.
    Set GPIB.IO = ioMgr.Open("GPIB0::16::INSTR")
    GPIB.IO.timeout = 10000
Dim Buf As String * 100
Dim srcPortNames As Variant
Dim numberOfSrcPorts As Integer
Dim segData As Variant
Const numSegs = 2
Const Chan = 1
Const addIFBW_PWR = 0
' In case of a measurement receiver VNA like N5264A
' which has no source ports, "SOURce:CATalog?" will
' return an empty list (just a pair of quotation marks)
GPIB.WriteString "SOURce:CATalog?", True
Buf = GPIB.ReadString
srcPortNames = Split(Buf, ",")
If Left(srcPortNames(0), 2) = Chr(34) & Chr(34) Then
    numberOfSrcPorts = 0
Else
    numberOfSrcPorts = UBound(srcPortNames) + 1
End If
' Building up a string consisting of the sweep segment data
' we want to set up. This example will create two segments.
' Set state of first segment to be ON (1 = ON, 0 = OFF),
' 101 points, start freq of 10 MHz, stop freq of 1 GHz
segData = "1,101,10E6,1E9"
' If you want to include one or more of: IFbandwidth, Dwell Time
' or Port Power, set Const addIFBW_PWR = 1
If addIFBW_PWR = 1 Then
    GPIB.WriteString "SENSe" & Chan & ":SEGMenT:BWIDth:CONTrol ON"
    GPIB.WriteString "SENSe" & Chan & ":SEGMenT:SWEep:TIME:CONTrol ON"
    GPIB.WriteString "SENSe" & Chan & ":SEGMenT:POWer:CONTrol ON"
    ' Turning off coupling allows power to vary per each port
    GPIB.WriteString "SOURce" & Chan & ":POWer:COUPlle OFF"
    segData = AddOptionalSettings(segData, numberOfSrcPorts)
End If
' Set state of second segment to be ON, 201 points,
' start freq of 1 GHz, stop freq of 3 GHz
segData = segData & ",1,201,1E9,3E9"

' Uncomment this line below only if you uncommented the
' AddOptionalSettings line above for the first segment.
' segData = AddOptionalSettings(segData, numberOfSrcPorts)
' Upload our segment list to the channel
GPIB.WriteString "SENSe1:SEGMenT:LIST SSTOp," & numSegs & "," & segData
' Set segment sweep type on Channel 1
GPIB.WriteString "SENSe1:SWEep:TYPE SEGmenT"
' Having the PNA display the segment sweep table for the channel
GPIB.WriteString "DISPlay:WINDow1:TABLe SEGmenT"

'*** End procedure
GPIB.IO.Close
Function AddOptionalSettings(ByVal pStr As String, ByVal numSrcPorts As Integer) As String
    Dim i
    ' Specifying 1 kHz IF bandwidth and Dwell Time of 0
    pStr = pStr & ", 1E3, 0"
    ' -10 dBm power for each of the source ports
    For i = 0 To numSrcPorts - 1
        pStr = pStr & ",-10"
    Next
    AddOptionalSettings = pStr
End Function
Uploading a Source Power Cal using SCPI

Programming the VNA using COM or using SICL/VISA over LAN (as in this example) leaves the VNA free to control GPIB devices as needed. This Visual Basic program demonstrates:

- Uploading a source power calibration of Port 2 for Channel 1.
- Reading the calibration data.

Learn more about Power Calibrations

Other SCPI Example Programs

To run this program, you need:

- Your PC and VNA both connected to a LAN (if using VISA LAN server / client).
- The SICL and VISA components of Keysight I/O Libraries software installed on your PC (both are included when you install the software, unless you already have another vendor's VISA installed. Then specify Full SICL and VISA installation to overwrite the other vendor's VISA.
- The module visa32.bas added to your VB project.
- A form with two buttons: cmdRun and cmdQuit.
- A VISA interface configured on your remote PC to control the VNA. This could be GPIB interface or a VISA LAN Client.

Private defRM As Long
Private viPNA As Long
Private status As Long
Private Sub Form_Load()
defRM = 0
End Sub
Private Sub cmdRun_Click()
' Session to VISA Default Resource Manager
' Dimensioned large enough to receive scalar comma-delimited values for 21 frequency points (20 ASCII characters per point)
Dim strReply As String * 420
Dim strPower As String, strCalPower As String
Dim strStimulus, strCalValue
Dim strResult As String

' Open the VISA default resource manager
status = viOpenDefaultRM(defRM)
If (status < VI_SUCCESS) Then HandleVISAErr

' Open a session (viPNA) to the VNA at "address 16" on the VISA
' interface configured as "GPIB0" on this PC.
status = viOpen(defRM, "GPIB0::16::INSTR", 0, 0, viPNA)
If (status < VI_SUCCESS) Then HandleVISAErr

' Set the number of sweep points to 2 on Channel 1.
status = myGPIBWrite(viPNA, "SENS1:SWE:POIN 2")
If (status < VI_SUCCESS) Then HandleVISAErr

' Ensure there's currently no source power cal on for this channel and port.
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR OFF")
If (status < VI_SUCCESS) Then HandleVISAErr

' Specify if the cal power level is offset (positive value for a gain, negative
' value for a loss) from the VNA port power setting on the channel when no source
' power cal is active. This is to account for components between the VNA test
' port and cal reference plane. In this example, let's set up our calibration
' at the output of an amplifier with 15 dB gain.
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:OFFS 15 DB")
If (status < VI_SUCCESS) Then HandleVISAErr

' Prepare for doing data transfer in ASCII format.
status = myGPIBWrite(viPNA, "FORM:DATA ASCII")
If (status < VI_SUCCESS) Then HandleVISAErr

' Send our source power correction data to the VNA. For purpose of simplicity
' in this example, we'll set up for no correction (0) at our start stimulus and
' 0.5 dB at our stop stimulus (recall that our sweep currently has just 2 points).
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:DATA 0,0.5")
If (status < VI_SUCCESS) Then HandleVISAErr

' Set the number of sweep points to 21 on Channel 1.
status = myGPIBWrite(viPNA, "SENS1:SWE:POIN 21")
If (status < VI_SUCCESS) Then HandleVISAErr

' Read the fixed power level for this port on Channel 1.
status = myGPIBWrite(viPNA, "SOUR1:POW2:LEV?")
If (status < VI_SUCCESS) Then HandleVISAErr
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAErr
strPower = strReply

' Turn the source power cal on.
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR ON")
If (status < VI_SUCCESS) Then HandleVISAErr
' Again read the fixed power level for this port on Channel 1
' (with our calibration turned on, this should now include the 15 dB offset
' we indicated our power amplifier provides).
status = myGPIBWrite(viPNA, "SOUR1:POW2:LEV?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError
strCalPower = strReply

' Read the stimulus values from Channel 1.
status = myGPIBWrite(viPNA, "SENS1:X?"")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Tokenize the reply string into an array containing the values
strStimulus = Split(strReply, ",")

' Read back the source power correction data, now interpolated for 21 points
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:DATA?"")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Tokenize the reply string into an array containing the values
strCalValue = Split(strReply, ",")

' Print the data using a message box (here, Chr returns the ASCII characters
' for Tab (9) and Linefeed (10)).
strResult = "PNA port power = " & Val(strPower) & Chr(10)
strResult = strResult & "Power at reference plane = " & Val(strCalPower) & Chr(10)
strResult = strResult & "Stimulus" & Chr(9) & Chr(9) & "Cal Value" & Chr(10)
For i = 0 To UBound(strStimulus)
    strResult = strResult & Val(strStimulus(i)) & Chr(9) & Val(strCalValue(i)) & Chr(10)
Next
MsgBox strResult
End Sub

Private Sub cmdQuit_Click()
    ' Close the resource manager session (which also closes
    ' the session to the VNA).
    If defRM <> 0 Then Call viClose(defRM)

    ' End the program
    End
End Sub

Private Function myGPIBWrite(ByVal viHandle As Long, ByVal strOut As String) As Long
    ' The "+ Chr$(10)" appends an ASCII linefeed character to the
' output, for terminating the write transaction.
myGPIBWrite = viVPrintf(viHandle, strOut + Chr$(10), 0)
End Function

Private Function myGPIBRead(ByVal viHandle As Long, strIn As String) As Long
myGPIBRead = viVScanf(viHandle, "%t", strIn)

' Remove trailing linefeed character
If Right(strIn, 1) = Chr(10) Then strIn = Left(strIn, Len(strIn) - 1)
End Function

Sub HandleVISAError()
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr, vbExclamation

' Close the resource manager session (which also closes
' the session to the VNA).
If defRM <> 0 Then Call viClose(defRM)
End
End Sub
The VISA (Virtual Instrument Software Architecture) **System Communicate** commands used in this example allow you to send SCPI commands to another device through the VNA. VISA is used to communicate with most instrumentation buses including the following:

- GPIB
- USB
- Serial
- Ethernet

**See Other SCPI Example Programs**

```vbnet
option explicit
dim app
set app = CreateObject("AgilentPNA835x.Application")
set scpi = app.ScpiStringParser

's Open a new VISA session and set timeout to 10 ms
scpi.Parse "SYST:COMM:VISA:RDEV:OPEN 'TCPIP0::A-N5242A-10096::hislip1::INSTR',10"

's Retrieve the session ID number
sessionNum = scpi.Parse "SYST:COMM:VISA:RDEV:OPEN?"

's Send session ID number and send RBW command to the device
scpi.Parse "SYST:COMM:VISA:RDEV:WRIT & sessionNum & 'RBW:ARB 1'"

's Send the "*IDN?" query
scpi.Parse "SYST:COMM:VISA:RDEV:WRITE & sessionNum & '*IDN?'"

's Read its results
idnstr = scpi.Parse "SYST:COMM:VISA:RDEV:READ?" & sessionNum

's Close the VISA session
scpi.Parse "SYST:COMM:VISA:RDEV:CLOSE" & sessionNum
```
The General Purpose Interface Bus (GPIB) is a system of hardware and software that allows you to control test equipment to make measurements quickly and accurately. This topic contains the following information:

- The GPIB Hardware Components
- The GPIB / SCPI Programming Elements
- Specifications
- GPIB Interface Capability Codes

**Note:** All of the topics related to programming assume that you already know how to program, preferably using a language that can control instruments.

---

### Other Topics about GPIB Concepts

#### The GPIB Hardware Components

The system bus and its associated interface operations are defined by the IEEE 488 standard. The following sections list and describe the main pieces of hardware in a GPIB system:

---

Early VNA models had only ONE GPIB connector. These models could control other GPIB devices using one of, or a combination of, the following methods:

- Use the SCPI `SYST:COMM:GPIB:RDEV:` commands.
- Use VISA or SICL over LAN to accomplish this. See an example.
- Use USB / GPIB Interface

**Note:** Current VNA models have dedicated Controller and Talker/Listener GPIB ports. See how to configure these ports.

#### Controllers

Controllers specify the instruments that will be the talker and listener in a data exchange. The controller of the bus must have a GPIB interface card to communicate on the GPIB.
The **Active Controller** is the computer or instrument that is currently controlling data exchanges.

The **System Controller** is the only computer or instrument that can take control and give up control of the GPIB to another computer or instrument, which is then called the active controller.

**Talker / Listener Instruments and GPIB Addresses**

- **Talkers** are instruments that can be addressed to send data to the controller.
- **Listeners** are instruments that can be addressed to receive a command, and then respond to the command. All devices on the bus are required to listen.

Every GPIB instrument must have its own unique address on the bus. The VNA address (default = 716) consists of two parts:

1. **The Interface select code** (typically 7) indicates which GPIB port in the system controller is used to communicate with the device.

2. **The primary address** (16) is set at the factory. You can change the primary address of any device on the bus to any number between 0 and 30. To change the analyzer address click System / Configure / SICL-GPIB.

**A secondary address** is sometimes used to allow access to individual modules in a modular instrument system, such as a VXI mainframe. The VNA does NOT have a secondary address.

**Cables**

GPIB Cables are the physical link connecting all of the devices on the bus. There are eight data lines in a GPIB cable that send data from one device to another. There are also eight control lines that manage traffic on the data lines and control other interface operations.

You can connect instruments to the controller in any arrangement with the following limitations:

- Do not connect more than 15 devices on any GPIB system. This number can be extended with the use of a bus extension.
- Do not exceed a total of 20 meters of total cable length or 2 meters per device, whichever is less.
- Avoid stacking more than three connectors on the back panel of an instrument. This can cause unnecessary strain on the rear-panel connector.

**The GPIB / SCPI Programming Elements**

The following software programming elements combine to become a GPIB program:
The GPIB command is the basic unit of communication in a GPIB system. The analyzer responds to three types of GPIB commands:

1. **IEEE 488.1 Bus-management Commands**

These commands are used primarily to tell some or all of the devices on the bus to perform certain interface operations.

All of the functions that can be accomplished with these commands can also be done with IEEE 488.2 or SCPI commands. Therefore, these commands are not documented in this Help system. For a complete list of IEEE 488.1 commands refer to the IEEE 488 standard. **Examples** of IEEE 488.1 Commands

- **CLEAR** - Clears the bus of any pending operations
- **LOCAL** - Returns instruments to local operation

2. **IEEE 488.2 Common Commands**

These commands are sent to instruments to perform interface operations. An IEEE 488.2 common command consists of a single mnemonic and is preceded by an asterisk (*). Some of the commands have a query form which adds a "?" after the command. These commands ask the instrument for the current setting. See a complete list of the **Common Commands** that are recognized by the analyzer. **Examples** of IEEE 488.2 Common Commands

- **OPC** - Operation Complete
- **RST** - Reset
- **OPT?** - Queries the option configuration

3. **SCPI Commands**

The Standard Commands for Programmable Instruments (SCPI) is a set of commands developed in 1990. The standardization provided in SCPI commands helps ensure that programs written for a particular SCPI instrument are easily adapted to work with a similar SCPI instrument. SCPI commands tell instruments to do device specific functions. For example, SCPI commands could tell an instrument to make a measurement and output data to a controller. **Examples** of SCPI Commands:
For more information on SCPI:

- The Rules and Syntax of SCPI Commands provides more detail of the SCPI command structure.

- SCPI Command Tree is a complete list of the SCPI commands for the analyzer

**Programming Statements**

SCPI commands are included with the language specific I/O statements to form program statements. The programming language determines the syntax of the programming statements. SCPI programs can be written in a variety of programming languages such as VEE, HP BASIC, or C++. **Example** of a Visual Basic statement:

- `GPIB.Write "SOURCE:FREQUENCY:FIXED 1000 MHz"`

**Note about examples**

**Instrument Drivers**

Instrument drivers are subroutines that provide routine functionality and can be reused from program to program. GPIB industry leaders have written standards for use by programmers who develop drivers. When programmers write drivers that comply with the standards, the drivers can be used with predictable results. To comply with the standard, each instrument driver must include documentation describing its functionality and how it should be implemented.

**GPIB Specifications**

**Interconnected devices** - Up to 15 devices (maximum) on one contiguous bus.

**Interconnection path** - Star or linear (or mixed) bus network, up to 20 meters total transmission path length or 2 meters per device, whichever is less.

**Message transfer scheme** - Byte-serial, bit-parallel, asynchronous data transfer using an interlocking 3-wire handshake.

**Maximum data rate** - 1 megabyte per second over limited distances, 250 to 500 kilobytes per second typical maximum over a full transmission path. The devices on the bus determine the actual data rate.

**Address capability** - Primary addresses, 31 Talk and 31 Listen; secondary addresses, 961 Talk and 961 Listen. There can be a maximum of 1 Talker and up to 14 Listeners at a time on a single bus. See also previous section on **GPIB addresses**.
The IEEE 488.1 standard requires that all GPIB compatible instruments display their interface capabilities on the rear panel using codes. The codes on the analyzer, and their related descriptions, are listed below:

- **SH1** full source handshake capability
- **AH1** full acceptor handshake capability
- **T6** basic talker, serial poll, no talk only, unaddress if MLA (My Listen Address)
- **TEO** no extended talker capability
- **L4** basic listener, no listen only, unaddress if MTA (My Talk Address)
- **LEO** no extended listener capability
- **SR1** full service request capability
- **RL1** full remote / local capability
- **PPO** **no parallel poll capability**
- **DC1** full device clear capability
- **DT1** full device trigger capability
- **C1** system controller capability
- **C2** send IFC (Interface Clear) and take charge controller capability
- **C3** send REN (Remote Enable) controller capability
- **C4** respond to SRQ (Service Request)
The Rules and Syntax of SCPI

Most of the commands used for controlling instruments on the GPIB are SCPI commands. The following sections will help you learn to use SCPI commands in your programs.

- Branches on the Command Tree
- Command and Query
- Multiple Commands
- Command Abbreviation
- Bracketed (Optional) Keywords
- Vertical Bars (Pipes)
- MIN and MAX Parameters

Other Topics about GPIB Concepts

Branches on the Command Tree

All major functions on the analyzer are assigned keywords which are called ROOT commands. (See GPIB Command Finder for a list of SCPI root commands). Under these root commands are branches that contain one or more keywords. The branching continues until each analyzer function is assigned to a branch. A root command and the branches below it is sometimes known as a subsystem.

For example, under `SOURCE:POWER` are several branch commands.

Sometimes the same keyword, such as `STATE`, is used in several branches of the command tree. To keep track of the current branch, the analyzer's command parser uses the following rules:

- **Power On and Reset** - After power is cycled or after `*RST`, the current path is set to the root level commands.

- **Message Terminators** - A message terminator, such as a `<NL>` character, sets the current path to the root command level. Many programming language output statements send message terminators automatically. Message terminators are described in Sending Messages to the Analyzer.

- **Colon (:)** - When a colon is between two command keywords, it moves the current path down one level in the command tree. For example, the colon in `:SOURCE:POWER` specifies that `POWER` is one level below `SOURCE`. When the colon is the first character of a command, it specifies that the following keyword is a root level command. For example, the colon in `:SOURCE` specifies that source is a root level command.
Note: You can omit the leading colon if the command is the first of a new program line. For example, the following two commands are equivalent:
SOUR:POW:ATT:AUTO
:SOUR:POW:ATT:AUTO

- **<WSP>** - Whitespace characters, such as <tab> and <space>, are generally ignored. There are two important exceptions:
  - Whitespace inside a keyword, such as :CALCULATE, is not allowed.
  - Most commands end with a parameter. You must use whitespace to separate these ending parameters from commands. **Always refer to the command documentation.** In the following example, there is whitespace between STATE and ON.

CALCULATE1:SMOOTHING:STATE ON

- **Comma (,)** - If a command requires more than one parameter, you must separate adjacent parameters using a comma. For example, the SYSTEM:TIME command requires three values to set the analyzer clock: one for hours, one for minutes, and one for seconds. A message to set the clock to 8:45 AM would be SYSTEM:TIME 8,45,0. Commas do not affect the current path.

- **Semicolon(;)** - A semicolon separates two commands in the same message without changing the current path. See **Multiple Commands** later in this topic.

- **IEEE 488.2 Common Commands** - Common commands, such as *RST, are not part of any subsystem. An instrument interprets them in the same way, regardless of the current path setting.

**Command and Query**

A SCPI command can be an Event command, Query command (a command that asks the analyzer for information), or both. The following are descriptions and examples of each form of command. GPIB Command Finder lists every SCPI command that is recognized by the analyzer, and its form.

<table>
<thead>
<tr>
<th>Form</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Event commands</strong> - cause an action to occur inside the analyzer.</td>
<td>:INITIATE:IMMEDIATE</td>
</tr>
<tr>
<td><strong>Query commands</strong> - query only; there is no associated analyzer state to set.</td>
<td>:SYSTem:ERRor?</td>
</tr>
<tr>
<td><strong>Command and query</strong> - set or query an analyzer setting. The query form appends a question mark (?) to the set form</td>
<td>:FORMat:DATA ! Command :FORMat:DATA? ! Query</td>
</tr>
</tbody>
</table>
Multiple Commands

You can send multiple commands within a single program message. By separating the commands with semicolons the current path does not change. The following examples show three methods to send two commands:

1. **Two program messages:**

   ```
   SOURCE:POWER:START 0DBM
   SOURCE:POWER:STOP 10DBM
   ```

2. **One long message.** A colon follows the semicolon that separates the two commands causing the command parser to reset to the root of the command tree. As a result, the next command is only valid if it includes the entire keyword path from the root of the tree:

   ```
   SOURCE:POWER:START 0DBM;:SOURCE:POWER:STOP 10DBM
   ```

3. **One short message.** The command parser keeps track of the position in the command tree. Therefore, you can simplify your program messages by including only the keyword at the same level in the command tree.

   ```
   SOURCE:POWER:START 0DBM;STOP 10DBM
   ```

Common Commands and SCPI Commands

You can send Common commands and SCPI commands together in the same message. (For more information on these types of commands see [GP-IB Fundamentals](#).) As in sending multiple SCPI commands, you must separate them with a semicolon.

**Example** of Common command and SCPI commands together

```
*RST;SENSE:FREQUENCY:CENTER 5MHZ;SPAN 100KHZ
```
The following is unacceptable - The first three keywords use neither short or long form.

```
SOURc: Powe: Atten: Auto on
```

The following is acceptable - All keywords are either short form or long form.

```
SOUR: POWer: ATT:AUTO on
```

In addition, the analyzer accepts lowercase and uppercase characters as equivalent as shown in the following equivalent commands:

```
source: POW:att: auto ON
Source: Pow: Att: Auto on
```

Optional [Bracketed] Keywords

You can omit some keywords without changing the effect of the command. These optional, or default, keywords are used in many subsystems and are identified by brackets in syntax diagrams.

**Example of Optional Keywords**

The HCOPY subsystem contains the optional keyword IMMEDIATE at its first branching point. Both of the following commands are equivalent:

```
"HCOPY: IMMEDIATE"
"HCOPY"
```

The syntax in this Help system looks like this:

```
HCOPY[: IMMEDIATE]
```

**Vertical Bars | Pipes**

Vertical bars, or "pipes", can be read as "or". They are used in syntax diagrams to separate alternative parameter options.

**Example of Vertical Bars:**

```
SOURce: POWer: ATTenuation: AUTO <on|off>
```

Either ON or OFF is a valid parameter option.

**MIN and MAX Parameters**

The special form parameters "MINimum" and "MAXimum" can be used with commands that specify single frequency (Hz) and time (seconds) as noted in the command documentation. **Note:** Also with these commands, kHz, MHz, and GHz are accepted as suffixes/units.

The short form (min) and long form (minimum) of these two keywords are equivalent.
- MAXimum refers to the largest value that the function can currently be set to
- MINimum refers to the smallest value that the function can currently be set to.

For example, the following command sets the start frequency to the smallest value that is currently possible:

```
SENS:FREQ:START MIN
```

In addition, the max and min values can also be queried for these commands.

For example, the following command returns the smallest value that Start Frequency can currently be set to:

```
SENS:FREQ:START? MIN
```

An error will be returned if a numeric parameter is sent that exceeds the MAX and MIN values.

For example, the following command will return an "Out of range" error message.

```
SENS:FREQ:START 1khz
```
Configure for GPIB, SCPI, and SICL

The following settings are used to configure the analyzer for remote control using SCPI commands.

**How to Configure for SICL / GPIB Operation**

**Using Hardkey/SoftTab/Softkey**

1. Press **System > System Setup > Remote Interface...**

**Using a mouse**

1. Click **Utility**
2. Select **System**
3. Select **System Setup**
4. Select **Remote Interface...**

**SICL / GPIB** dialog box help

![Remote Interface dialog box help](image-url)
**GPIB**

**Talker/Listener Address**  Sets the VNA address used to send and receive GPIB/SCPI commands to the system controller (external computer).

Use the National Instruments interface or the ACE (Keysight Connection Expert) interface to change the System Controller address. Use the VNA as the system controller of external devices. Learn about the [VNA as controller](#).

See the rear panel of the [VNA-X](#) and N522x models.

---

**SICL**

**SICL Enabled**  When checked, the analyzer is capable of running GPIB programs on its computer to control analyzer functions. The programs must be run from a GPIB-capable programming environment (VEE, Visual Basic). This mode does not allow control of external GPIB instruments. To uncheck this box, exit the VNA application - (Click File, then Exit). The VNA restarts with the SICL enabled box unchecked unless **Automatically Enable on Startup** is checked.

Learn more about [Configuring for VISA and SICL](#).

**Note:** When **SICL Enabled** is checked, the VNA VXI-11.2 interface is enabled, and if the VNA hard disk image is new enough to have the VXI-11.3 interface, it also enables that. Learn more about [LXI/VXI](#).

**Address**  Sets the VNA address.

**Automatically Enable on Startup**  When checked, SICL Enabled is automatically selected when starting the VNA application.

---

**LAN Sockets/Telnet**

Provides ability to communicate with the VNA from a PC that uses a Windows, or non-Windows, operating system.

- These settings are checked by default. If you have security concerns, clear these check boxes.

- These settings remain after the VNA is shutdown and restarted.

**Sockets Enabled**  When checked, provides the ability to control the VNA from a remote SCPI program using port number 5025. See the C# example that illustrates how this is done.
**Telnet Enabled**  When checked, provides the ability to send single SCPI commands from a remote Windows, or non-Windows, PC to the VNA using port number 5024.

How to send single SCPI commands using Telnet:

1. On the remote PC, click **Start**, then type **cmd** in the **Search programs and files** text field.

2. **Type: telnet <computer name> 5024**
   where <computer name> is the full computer name of the VNA. See how to find the computer name of the VNA.

3. A Telnet window with a **SCPI>** prompt should appear on the remote PC screen.

4. From the SCPI prompt:
   - Type single SCPI commands
   - If an invalid SCPI command is sent, the prompt will disappear. Press **Enter** or **Ctrl C** to recover the SCPI prompt.
   - To exit the telnet window click **X** in the upper-right corner.
   - To get a normal telnet prompt, press **Ctrl │** (closing bracket).
   - To close the normal telnet window, type **Quit** and press **Enter**.

**HiSLIP**

HiSLIP has the same functionality as VXI 11 (SICL) with better performance. Therefore, it is enabled by default on the analyzer.

**Address** 0 by default. No change is necessary.

On the remote computer, use address string TCPIP0::<hostname>::hislip<address>. ("hislip" is case-sensitive).

**Security**

**Enable Remote Drive Access**  When checked allows access to the hard disk. The default is unchecked. When unchecked, hard disk access is blocked and any of the following commands will return an error:

**MMEM:CAT:FILE?**

**MMEM:CAT:STAT?**
Show SCPI Parser Console  Launches a window that is used to send single SCPI/GPIB commands from the analyzer keyboard. This window can also be used to capture the SCPI traffic used over HiSLIP.

- Type a valid command, with appropriate arguments and press enter.
- Use the arrow keys to recall previous commands.

IO Monitor enables monitoring activity on the remote control.

Clicking on the Configure button accesses the IO Monitor Configuration dialog to enable/disable this function.
Enabling the IO monitor provides detailed error message with debug information.

Example of Error message:

- On: -109, "Missing parameter; calc:par:def aaa<Err>
- Off: -109, "Missing parameter"

Disabling the IO monitor increases measurement speed.

Local and Remote Operation

The analyzer LCL and RMT (Local and Remote) operation labels appear in the lower right corner of the status bar.

Note: The status bar is NOT visible when the analyzer is preset. See how to make the status bar visible.

- **LCL** appears when NOT under SCPI control
- **RMT** appears when under SCPI control. The RMT label does NOT appear when under COM control. Remote operation disables the front panel keys except for the Macro/Local key.

To return to Local (front panel) operation, press the Macro / Local key

Sending the GPIB "GTL" (go to local) command also returns the analyzer to Local operation.

Sending the GPIB "LLO" (local lockout) command disables the front panel Local button.
Getting Data from the Analyzer

Data is sent from the analyzer in response to program queries. Data can be short response messages, such as analyzer settings, or large blocks of measurement data. This topic discusses how to read query responses and measurement data from the analyzer in the most efficient manner.

- Response Message Syntax
- Clearing the Output Queue
- Response Data Types
- Transferring Measurement Data

**Note:** Some PCs use a modification of the IEEE floating point formats with the byte order reversed. To reverse the byte order for data transfer into a PC, use the FORMat:BORDER command.

**Other Topics about GPIB Concepts**

**Response Message Syntax**

Responses sent from the analyzer contain data, appropriate punctuation, and message terminators. <NL><^END> is always sent as a response message terminator. Most programming languages handle these terminators transparent to the programmer.

Response messages use commas and semicolons as separators in the following situations:

- a comma separates response data items when a single query command returns multiple values

  ```
  FORM:DATA? 'Query
  ASC, +0 'Analyzer Response
  ```

- a semicolon separates response data when multiple queries are sent within the same messages

  ```
  SENS:FREQ:STAR?;STOP? --Example Query
  +1.2300000000E+08; +7.8900000000E+08<NL><^END> 'Analyzer Response
  ```

**Clearing the Output Queue**
After receiving a query, the analyzer places the response message in its output queue. Your program should read the response immediately after the query is sent. This ensures that the response is not cleared before it is read. The response is cleared when one of the following conditions occur:

- When the query is not properly terminated with an ASCII carriage return character or the GPIB <^END> message.
- When a second program query is sent.
- When a program message is sent that exceeds the length of the input queue
- When a response message generates more response data than fits in the output queue.
- When the analyzer is switched ON.

Response Data Types

The analyzer sends different response data types depending on the parameter being queried. You need to know the type of data that will be returned so that you can declare the appropriate type of variable to accept the data. For more information on declaring variables see your programming language manual. The GPIB Command Finder lists every GPIB command and the return format of data in response to a query. The analyzer returns the following types of data:

- **Numeric Data**
- **Character Data**
- **String Data**
- **Block Data**

**Numeric Data**

All numeric data sent over the GPIB is ASCII character data. Your programming environment may convert the character data to numeric data for you. Boolean data (1 | 0 ) is a type of numeric data.

**Character Data**

Character data consists of ASCII characters grouped together in mnemonics that represent specific analyzer settings. The analyzer always returns the short form of the mnemonic in upper-case alpha characters. Character data looks like string data. Therefore, refer to the GPIB Command Finder to determine the return format for every command that can be queried.

**Example of Character Data**

MLOG
String Data

String data consists of ASCII characters. String parameters can contain virtually any set of ASCII characters. When sending string data to the analyzer, the string must begin with a single quote (') or a double quote (" ) and end with the same character (called the delimiter).

**Note:** The analyzer responds best to all special characters if the string is enclosed in single quotes. If quotes are not used, the analyzer will convert the text to uppercase. The analyzer may not respond as you expect.

The analyzer always encloses data in double quotes when it returns string data.

**Example** of String Data

```
GPIB.Write "DISP:WINDow:TITLe:DATA?"
"This is string response data."
```

Block Data

Block data is used to transfer measurement data. Although the analyzer will accept either definite length blocks or indefinite length blocks, it always returns definite length block data in response to queries unless the specified format is ASCII. The following graphic shows the syntax for definite block data:

```
#  <num_digits>  <byte_count>  data byte  <NL>  <End>
```

<num_digits> specifies how many digits are contained in <byte_count>

<byte_count> specifies how many data bytes will follow in <data bytes>

**Example** of Definite Block Data

```
#210ABCDE+WXYZ<nl><end>
```

Where:

- # - always sent before definite block data
- 2 - specifies that the byte count is two digits (2)
- 10 - specifies the number of data bytes that will follow, not counting <NL><END>
Transferring Measurement Data

Measurement data is blocks of numbers that result from an analyzer measurement. Measurement data is available from various processing arrays within the analyzer. For more information on the analyzer's data processing flow, see Accessing Data Map. Regardless of which measurement array is read, transferring measurement data is done the same.

See an example.

When transferring measurement data, the FORMat:DATA command allows you to choose from the following two data types:

- REAL
- ASCII

The following graphic shows the differences in transfer times between the two:

![Data Transfer Times Graph]

REAL Data

REAL data (also called floating-point data) types transfer faster. This is because REAL data is binary and takes about half the space of ASCII data. The disadvantage of using REAL data is that it requires a header that must be read. See definite length block data. The binary floating-point formats are defined in the IEEE 754-1985 standard. The following choices are available in REAL format:

- **REAL,32** - IEEE 32-bit format - single precision (not supported by HP BASIC)
- **REAL,64** - IEEE 64-bit format - double precision

ASCII Data

The easiest and slowest way to transfer measurement data is to use ASCII data. ASCII data is sent if the
data contains both numbers and characters (the setting of FORMat:DATA is ignored). ASCII data is separated by commas.
Synchronizing the VNA and Controller

Synchronizing the VNA (Vector Network Analyzer) and Controller means to keep VNA and the controller working at approximately the same pace. In this topic:

- The Problem and the Solution
- VNA Queues
- Synchronization Methods
- When To Synchronize the Analyzer and Controller
  - Completion of a Measurement
  - Measurements with External Trigger
  - Averaged Measurements
  - During Calibration Acquire

See Also

- Synchronize an External PSG Source
- Triggering the VNA using SCPI

The Problem

The controller sends commands to the VNA as fast as the bus will allow. The VNA stores these commands in the VNA Input queue. However, the VNA executes those commands at a slower rate than they are accepted. If left unchecked, the VNA input buffer will contain a long list of commands waiting to be executed.

At some point, the controller will send a query command which requires a response from the VNA. The controller will not send more commands until a response is received. It will wait for a response from the VNA for the amount of time set by the Timeout setting. If the VNA is working off a long list of commands in the input buffer, it may not execute and respond to the query command until the controller has quit waiting, or "timed out".

The Solution

The easiest way to keep the controller and the VNA "synched" is to send query commands often. This
stops the controller from sending more commands until the VNA executes and responds to the query. This limits the number of commands that are waiting in the VNA input queue to be processed.

Although any query will stop the controller from sending more commands, a good practice is to send *OPC? Most of the time, as soon as this query is executed, the VNA will immediately reply. The exception to this is the Overlapped command.

- **Sequential** commands are executed quickly and in the order in which they are received.

- **Overlapped** (also known as Asynchronous) commands take longer to execute. Therefore, they allow the VNA to execute other commands while waiting. However, the programmer may want to prevent the analyzer from processing new commands until the overlapped command has completed. If the VNA is executing an overlapped command when a *OPC? is received, it will wait until the overlapped command is complete before processing new commands.

**Note:** The analyzer has two overlapped commands:

- INITiate:IMMediate
- SENSE:SWEep:MODE GROUPS (when INIT:CONT is ON)

Several calibration commands have an optional ASYNcronous argument which allows them to behave like overlapped commands. Learn more.

### Analyzer Queues

Queues are memory buffers that store messages until they can be processed. The analyzer has the following queues:

- **Input Queue**
- **Output Queue**
- **Error Queue**

### Input Queue

The controller sends statements to the analyzer without regard to the amount of time required to execute the statements. The input queue is very large (31k bytes). It temporarily stores commands and queries from the controller until they are read by the analyzer's command parser. The input queue is cleared when the analyzer is switched ON.

### Output Queue

When the analyzer parses a query, the response is placed in the output queue until the controller reads
it. Your program should immediately read the response or it may be cleared from the output queue. The following conditions will clear a query response:

- When a second query is sent before reading the response to the first. This does not apply when multiple queries are sent in the same statement.
- When a program statement is sent that exceeds the length of the input queue.
- When a response statement generates more data than fits in the output queue.
- When the analyzer is switched ON.

**Error Queue**

Each time the analyzer detects an error, it places a message in the error queue. When the `SYSTEM:ERROR?` query is sent, one message is moved from the error queue to the output queue so it can be read by the controller. Error messages are delivered to the output queue in the order they were received. The error queue is cleared when any of the following conditions occur:

- When the analyzer is switched ON.
- When the `*CLS` command is sent to the analyzer.
- When all of the errors are read.

If the error queue overflows, the last error is replaced with a "Queue Overflow" error. The oldest errors remain in the queue and the most recent error is discarded.

**Synchronization Methods**

The following common commands are used to synchronize the analyzer and controller. Examples are included that illustrate the use of each command in a program. See the SCPI command details to determine if a command is an overlapped command.

- `*WAI`
- `*OPC?`
- `*OPC`

**WAI**

The *WAI command:
• **Stops the analyzer** from processing subsequent commands until all overlapped commands are completed.

• **It does NOT stop the controller** from sending commands to this and other devices on the bus. This is the easiest method of synchronization.

**Example** of the *WAI command

```
"ABORT;:INITIATE:IMMEDIATE" 'Restart the measurement.
"CALCULATE:MARKER:SEARCH:MAXIMUM" 'Search for max amplitude.
"CALCULATE:MARKER:X?" 'Which frequency?
```

The following time line shows how the processing times of the three commands relate to each other:

```
ABORT;:INITIATE:IMMEDIATE

CALCULATE:MARKER:MAXIMUM

CALCULATE:MARKER:X?
```

**INITIATE:IMMEDIATE** is an overlapped command. It allows the immediate processing of the sequential command, **CALCULATE:MARKER:SEARCH:MAXIMUM**. However, the **INITIATE:IMMEDIATE** is not considered complete until the measurement is complete. Therefore, the marker searches for maximum amplitude before the measurement completes. **The **CALCULATE:MARKER:X?** query could return an inaccurate value.**

To solve the problem, insert a *WAI command.

```
"ABORT;:INITIATE:IMMEDIATE" 'Restart the measurement.
"*WAI" 'Wait until complete.
"CALCULATE:MARKER:MAXIMUM" 'Search for max amplitude.
"CALCULATE:MARKER:X?" 'Which frequency
```

The time line now looks like this:

```
ABORT;:INITIATE:IMMEDIATE

*WAI

CALCULATE:MARKER:MAXIMUM

CALCULATE:MARKER:X?
```

The **WAI command keeps the MARKER:SEARCH:MAXIMUM from taking place until the measurement is completed. The **CALCULATE:MARKER:X?** query returns the correct value.**
**Note:** Although *WAI stops the analyzer from processing subsequent commands, it does not stop the controller. The controller could send commands to other devices on the bus.

**OPC?**

The *OPC? query stops the controller until all pending commands are completed.

In the following example, the `Read` statement following the *OPC? query will not complete until the analyzer responds, which will not happen until all pending commands have finished. Therefore, the analyzer and other devices receive no subsequent commands. A "1" is placed in the analyzer output queue when the analyzer completes processing an overlapped command. The "1" in the output queue satisfies the `Read` command and the program continues.

**Example of the *OPC? query**

This program determines which frequency contains the maximum amplitude.

```
"ABORT; :INITIATE:IMMEDIATE"! Restart the measurement
"*OPC?" 'Wait until complete
Meas_done = GPIB.Read 'Read output queue, throw away result
"CALCULATE:MARKER:MAX" 'Search for max amplitude
"CALCULATE:MARKER:X?" 'Which frequency?
Marker_x = GPIB.Read
PRINT "MARKER at " & Marker_x & " Hz"
```

**OPC**

The *OPC command allows the analyzer and the controller to process commands while processing the overlapped command.

When the analyzer completes processing an overlapped command, the *OPC command sets bit 0 of the standard event register to 1. This requires polling of status bytes or use of the service request (SRQ) capabilities of your controller. See *Reading the Analyzer's Status Registers* for more information about the standard event status register, generating SRQs, and handling interrupts.

**Note:** Be careful when sending commands to the analyzer between the time you send *OPC and the time you receive the interrupt. Some commands could jeopardize the integrity of your measurement. It also could affect how the instrument responds to the previously sent *OPC.

**Example of polled bit and SRQ processes.**

**When To Synchronize the Analyzer and Controller**

The need to synchronize depends upon the situation in which the overlapped command is executed. The following section describes situations when synchronization is required to ensure a successful
operation.

- Completion of a Measurement
- Measurements with External Trigger
- Averaged Measurements

**Completion of a Measurement**

To synchronize the analyzer and controller to the completion of a measurement, use the \texttt{ABORT; INITIATE; IMMEDIATE} command sequence to initiate the measurement.

This command sequence forces data collection to start (or restart) under the current measurement configuration. A restart sequence, such as \texttt{ABORT; INITIATE; IMMEDIATE} is an overlapped command. It is complete when all operations initiated by that restart command sequence, including the measurement, are finished. The \texttt{*WAI}, \texttt{*OPC?} and \texttt{*OPC} commands allow you to determine when a measurement is complete. This ensures that valid measurement data is available for further processing.

**Measurements with External Trigger**

See Triggering the VNA using SCPI.

**External Triggering**

**Averaged Measurements**

Averaged measurements are complete when the average count is reached. The average count is reached when the specified number of individual measurements is combined into one averaged measurement result. Use synchronization to determine when the average count has been reached.

If the analyzer continues to measure and average the results after the average count is reached, use synchronization to determine when each subsequent measurement is complete.

**During Calibration Acquire**

During a calibration with slow sweep speeds, such as when using a narrow IF bandwidth, you may want to have your program perform other operations, such as checking for the click event of a Cancel button.

To do this, use the optional ASYNchronous argument with the ACQuire command as shown in several calibration example programs. The VNA parser returns immediately while the cal step measurement proceeds. It does NOT block commands and wait for the measurement step to finish. You can send \texttt{*ESR?} or \texttt{*STB?} queries to monitor the status register bytes to see when the OPC (operation complete) bit gets set, which indicates the cal measurement step has finished. Learn more about status registers.
Note: Do NOT issue the *OPC? query when using the ASYN argument. Using the *OPC? query during the ASYN command will cause the program to block immediately. Instead, use the default SYNC argument.

When using the ASYN argument, set the timeout value in the IO settings to at least 5 seconds. There are intervals during the cal acquires when the VNA takes a several seconds to respond to additional commands, such as when the processor is calculating error terms.

The following commands have this argument:

<table>
<thead>
<tr>
<th>Command</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:COLL:GUID:ACQuire (Guided Cal)</td>
<td>Guided 2-Port or 4-Port Cal</td>
</tr>
<tr>
<td>SENS:CORR:COLL:ACQuire (Unguided Cal)</td>
<td>Perform Unguided ECAL</td>
</tr>
<tr>
<td></td>
<td>(shows polling loop)</td>
</tr>
</tbody>
</table>

In addition, the SENS:CORR:COLL:GUIDed:INITialize command has this optional argument for long calibration initialization, such as a CalAll calibration.
Calibrating the Analyzer Using SCPI

There are several ways to calibrate the analyzer using SCPI depending on your measurement needs. As from the Cal Wizard, you can perform a Guided Cal, Unguided Cal, or ECal. This topic explains the differences in these calibration choices when using SCPI commands.

- Guided Calibrations
- ECal
- Creating Cal Sets
- Applying Cal Sets and Cal Types
- Uploading Error Terms
- Unguided Cals and Calibration Classes

Note: ALWAYS send ALL measurement setup commands BEFORE initializing a remote calibration.

See Also

Synchronizing the Analyzer and Controller (During a calibration)

See SCPI Calibration Examples

Guided Calibrations

Guided versus Unguided is the style of calibration that is selected on the first page of the Calibration Wizard. A remote 'guided' cal does not present the cal wizard, but prompts for specific standards to be connected. In a remote 'Unguided', the steps must be 'hard-coded'.

- To perform a Guided Calibration, use ONLY Sens:Corr:Coll:Guided commands.

- These commands calibrate the ACTIVE channel. Activate a channel by selecting a measurement on the channel to be calibrated using Calc:Par:Select.

- Full 1,2,3,4-port SOLT and TRL calibrations - No response cals.

- All of the advanced calibration features (Thru method, specify DUT connectors and Cal kits for each port, port pairings).

- A Cal Set is applied to the channel and saved at the completion of a guided cal according to the preference
setting SENS:CORR:PREF:CSET:SAVE

Note: To perform an **Unguided Calibration**, use ONLY the Sens:Corr commands (NOT Guided).

**ECal**

From the Cal Wizard or from a SCPI program, ECal is fast, accurate, and very repeatable. Unlike from the Cal Wizard, you can use SCPI to perform ECal using either the Guided or Unguided commands. The Unguided commands are easiest to use. However, the following situations require that you use the Guided commands.

- To maximize accuracy, all ECal calibrations on the analyzer perform an Unknown Thru measurement of the ECal module Thru state **IF** the analyzer model being used has 1 reference receiver per port. If your analyzer does NOT have 1 reference receiver per port, use Guided ECal commands and specify a Thru method.

- If your ECal module connectors do NOT match the DUT connectors, and you choose not to perform a User Characterization, use Guided ECal commands and specify the Thru method.

**ECAL Notes:**

- When using either Guided or Unguided ECal commands under low power situations, use the Orientation settings. The Guided example shows the use of these commands. When using Unguided, they must appear before the Acquire command.

- The frequency range of the measurement must be within the range of the ECal module. Otherwise, the calibration will fail.

- You do NOT have to send the ECal module state 'switch' commands. The ECal algorithm switches ECal states automatically.

- All of these ECal choices are listed in the Programming Command Finder function in this Help file.

See **Using ECal** to learn about all of the ECal features.

**Creating Cal Sets**

There are several ways to store guided cal data into a unique Cal Set. The following is probably the easiest. It does not require the name of an existing Cal Set and it allows you to name the Cal Set.
Applying Cal Sets and Cal Types

A Cal Set is applied to the channel and saved at the completion of a guided cal according to the preference setting SENS:CORR:PREF:CSET:SAVE.

When you select a Cal Set to apply to an uncalibrated channel, the analyzer attempts to find the most comprehensive calibration type in the Cal Set and turn it ON. In addition, changing a measurement parameter (for example, from S11 to S21) will also initiate an attempt to apply the best Cal Type and turn correction ON.

There may be times when you do not want the most comprehensive Cal Type. For example, say there is a Full 2-port Cal Set applied, but there is only an S11 measurement displayed. If measurement speed is a concern, you can apply a Full 1-Port Cal Type from that same Cal Set and save time by not doing the extra background sweeps. Learn more about background sweeps.

If you change the measurement parameter, the analyzer will reapply the Full 2-Port Cal Type.

See the SCPI and COM commands for Cal Sets and Cal Types.

Uploading Error Terms

Note: There was a method described here for WinCal 3.x that involved a preference setting. That method is no longer supported.

To upload error terms into a created or selected Cal Set:

SENS:CORR:CSET:CREAtE or SENS:CORR:CSET:GUID
SENS:CORR:CSET:Data <term> <port> <port> <data>
SENS:CORR:CSET:SAVE

This method puts error terms into a Cal Set, outside of a Guided or Unguided calibration session.

The Cal Set can then be applied at any time.

See SENS:CORR:CSET commands.
Unguided Cals and Calibration Classes

- Use Sens:Correction commands.
- 1-port, 2-port, Response.
- Can select 2 sets of standards.
- TRL is NOT recommended.

The following describes how to perform an unguided calibration using SCPI. The objective here is to make clear the relationship between the physical port on which a standard is being measured, the actual device in the cal kit, and the SCPI command used to acquire the device.

Calibration standards classes are ‘categories’ of standard types. To perform a 2 port calibration, the cal wizard requires the following types of standards to be measured:

3 reflection standards on the forward port:

- Class S11A typically an open
- Class S11B typically a short
- Class S11C typically a load

Likewise, 3 reflection standards are required for the reverse port:

- Class S22A typically an open
- Class S22B typically a short
- Class S22C typically a load

There is also a transmission standard that is measured in both directions:

- Class S21T typically a thru

The following illustrates the relationship between cal kit physical standards and calibration classes. Here is a list of the physical devices in my calibration kit.

Standard #1 = "3.5 mm male short"
Standard #2 = "3.5 mm male open"
Standard #3 = "3.5 mm male broadband load"
Standard #4 = "Insertable thru standard"
Standard #5 = "3.5 mm male sliding load"
Standard #6 = "3.5 mm male lowband load"
Standard #7 = "3.5 mm female short"
Standard #8 = "female to female characterized thru adapter"
Standard #9 = "0-2 Load"
Standard #10 = "Open"
Standard #11 = "Non-insertable thru"
Standard #12 = "3.5 mm female lowband load"
Standard #13 = "3.5 mm female sliding load"
Standard #14 = "3.5 mm female broadband load"
Standard #15 = "3.5 mm female open"

When you perform a calibration remotely using SCPI, you don’t specify the device number directly. Rather, you specify the class you want to measure. Each device in the calibration kit is assigned to a class. And since more than one device can be assigned to the same class, each class contains an ordered list of devices. The class assignments are set using the Advanced Modify Cal Kit dialog or the SCPI command:

SENS:CORR:COLL:CKIT:ORDER <class>, <std>, <std>, <std>, <std>,<std>,<std>,<std>

The 85052B kit used in the example program has the following standard list for each class: The list was obtained by issuing the corresponding SCPI query:

SENS:CORR:COLL:CKIT:OLIST1? S11A = +2,+15,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST2? S11B = +1,+7,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST3? S11C = +6,+5,+3,+12,+13,+14,+0
SENS:CORR:COLL:CKIT:OLIST4? S21T = +4,+8,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST5? S22A = +2,+15,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST6? S22B = +1,+7,+0,+0,+0,+0,+0
When you perform the calibration, you acquire data by issuing the `ACQuire` command:

```
SENS:CORR:COLL:ACQ <class>[, <subst> ]
```

For example:

```
SENS:CORR:COLL:SFOR 1

SENS:CORR:COLL:ACQ STANA, SST2
```

The `SFOR` command tells the wizard to make the next acquisition in the forward direction. The `ACQuire` command specifies that we are measuring the 2nd device in the list for STANA. And since we are measuring `SFOR`ward, then STANA refers to class #1 or S11A. The list of devices for this class are specified in the `OLIST1` query above.

Alternately, you could modify the device order for the S11A class to move device #15 into the first position (SENS:CORR:COLL:CKIT:ORDER1). When the desired device is in the first position, you need not specify the order number in the `ACQuire` command. The default is the first device in the `OLIST`. This works well for two port network analyzers where the order for S11A,B,C classes is set up for port 1 and the order for S22A,B,C is set up for port 2. With the kit set up in the proper order, you eliminate the need to specify the substandard number (SST<n>).

See an example: Perform an Unguided 2-port Cal on a 4-port analyzer.
The VNA as a USB Device

The VNA can be controlled as a USB Device using SCPI. This is done through the Keysight I/O Libraries which must be installed on your remote computer.

All data types, especially Binary block data, transfer MUCH faster using USB as compared to GPIB.

To communicate with the VNA as a USB device

1. Connect the VNA to the remote computer using the rear-panel device-side USB connector.
2. The 'Found New Hardware' wizard is launched. Follow the prompts to install the VNA driver software.
3. The Keysight I/O Libraries will recognize the VNA as a Test and Measurement device and show the following dialog.

Note: The VNA is not a USB Mass Storage Device. Therefore, Windows Explorer does NOT recognize it as a USB device. You can NOT use Windows Explorer to transfer files to and from the VNA. or file transfer, use the SCPI command MMEM:TRANsfer.

Alias name  Change this to a name that is easy to recognize. Once configured, use the Alias name to communicate with the USB device using applications such as VISA and SICL:

- VISA: viOpen (...,"UsbDevice1",...)
- **SICL**: `iopen ("UsbDevice1")`

For more information, see the Connectivity Guide in the Keysight I/O libraries.
Reading the Analyzer's Status Register

The VNA has several status registers that your program can read to know when specific events occur. There are two methods of reading the status registers in the analyzer: the Polled Bit method and the Service Request method.

- The Status Registers
- Setting and Reading Bits in Status Registers
- Polled Bit Method
- Service Request Method

See Also

IEE 482 Common commands
Example: Status Reporting
Status Commands

Other Topics about GPIB Concepts

Important Notes:

- A new Limit Line Fail command that makes it easy to determine if Limit Line testing has failed.
- *OPC? can be used to easily determine when a channel has completed a sweep. This requires no interaction with the Status Register system. Most VNA programming examples use *OPC?.
- Most of the Status Register system can NOT be used with the SCPIStringParser Object. However, *OPC? can be used.

The Status Registers

Most of the status registers in the analyzer have sixteen bits. For simplicity, this topic will illustrate their use with 8-bit registers. Bits in registers represent the status of different conditions inside of the analyzer. In the following graphic, a register is represented by a row of boxes; each box represents a bit. Bit 3 is ON.
Each VNA Status Register is actually comprised of the following registers. See an image of the VNA Status registers.

- **Enable Registers** - When using the SRQ method of polling, you first set bits in the enable register which tells the VNA which events to monitor. This is not necessary using the Polled Bit method, as you can only monitor a single event. A *CLS (clear status) command will not clear the enable register. The *ESE and *ESE? commands are used to set and query Enable bits, while *ESR is used to read and clear an Enable register. Learn how to set bits.

- **Condition Registers** - A condition register continuously monitors events in the VNA. Bits in the condition register change real time as conditions occur. These bits are not latched, so this register is used mainly for diagnostic purposes. The registers that only summarize lower level registers do NOT have a condition register.

- **Event Registers** - This is the register that is read to determine if an event has occurred. An event register latches the bits from the corresponding condition register. When an event register bit is set, subsequent changes to the corresponding condition register bit are ignored. The bit remains set until a query command such as *CLS clears the bit. Learn how to read the Event Register.

- **Positive and Negative Transition Registers** - Transition registers control what type of condition register will set the corresponding bit in the event register.
  - **Positive** transitions (0 to 1) are only reported to the event register if the corresponding positive transition bit is set to 1.
  - **Negative** transitions (1 to 0) are only reported to the event register if the corresponding negative transition bit is set to 1.
  - Setting both transition bits to 1 causes both positive and negative transitions to be reported.

Transition registers are read-write and are unaffected by *CLS (clear status) or queries. They are reset to their default settings at power-up and after *RST and SYSTem:PRESet commands. The following are the default settings for the transition registers:

- All Positive Transition registers = 1
- All Negative Transition registers = 0

This means that, by default, the analyzer will latch all event registers on the negative to positive transition (0 to 1).

The following is an example of why you would set transition registers:

A critical measurement requires that you average 10 measurements and then restart averaging. You decide to poll the averaging bit. When averaging is complete, the bit makes a positive transition. After restart, you poll the bit to ensure that it is set back from 1 to 0, a negative
transition. You set the negative transition bit for the averaging register.

### Setting and Reading Bits in Status Registers

Both the Polled-Bit method and Service Request method require that you set and read status register bits. Most of the VNA status registers contain 16 bits, numbered 0 to 15. Each bit has a weighted value. The following example shows how to set the bits in a 8-bit status register.

<table>
<thead>
<tr>
<th>Bit</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>16</td>
<td>32</td>
<td>64</td>
<td>128</td>
</tr>
</tbody>
</table>

How to set bits 4 and 5 in the Standard Event Status Enable register:

<table>
<thead>
<tr>
<th>Step</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Determine the weighted bit value for these weights 16 and 32 (respectively) bits</td>
<td></td>
</tr>
<tr>
<td>2. Add these values together</td>
<td>16 + 32 = 48</td>
</tr>
<tr>
<td>3. Send this number as an argument in the appropriate command. (see Status Commands)</td>
<td>STAT:QUES:LIMIT1:ENAB 48</td>
</tr>
</tbody>
</table>

### The Polled Bit Method

With the Polled Bit Method, your program monitors a bit in the status register that represents the condition of interest to you. When the VNA sets the bit to 1, your program sees it and responds accordingly.

- If your program **periodically** monitors a bit in the status register, it is free to do other things as well. However, your program can respond only as fast as the bit is polled.

- If your program **continually** monitors a bit, it can respond immediately, but will be unavailable to do anything other than poll the bit.

**Advantage:** This method requires very little programming.

**Procedure:**

1. Decide which condition to monitor. The Status Commands topic lists all of the possible conditions that can be monitored in the analyzer.

2. Determine the command to be used to monitor the bit.
3. Determine how often to poll the bit until it is set.

4. Construct the routine to respond when the bit is set.

---

**The Service Request (SRQ) Method**

Your program enables the bits in the status registers representing the condition of interest. When the condition occurs, the VNA actively interrupts your program from whatever it is doing, and an event handler in your program responds accordingly. Do this method if you have several conditions you want to monitor or the conditions are such that it is not practical to wait for the condition to occur.

**Advantage:** This method frees your program to do other things until the condition occurs. The program is interrupted to respond to the condition.

**Disadvantage:** This method can require extensive programming depending on the number and type of conditions that you want to monitor.

**Procedure:**

1. Decide which conditions to monitor. The Status Commands topic lists all of the possible analyzer conditions that can be monitored.

2. Set the enable bits in the **summary** registers and the **status byte** register.

   **Enabling** is like making power available to a light. Without power available, the switch can be activated, but the light won't turn ON. In the analyzer, without first enabling a bit, the condition may occur, but the controller won't see it unless it is enabled.

   The condition, and the bit in the **summary** registers in the reporting path, must be enabled. This is like streams (conditions) flowing into rivers (summary registers), and rivers flowing into the ocean (controller). See the diagram of status registers in Status Commands.

   Bit 6 of the **status byte** register is the only bit that can interrupt the controller. When any representative bit in the status byte register goes ON, bit 6 is automatically switched ON.

3. Enable your program to interrupt the controller. This is done several ways depending on the programming language and GPIB interface card you use. An example program is provided showing how this is done with in Visual Basic with a National Instruments GPIB card.

4. Construct a subroutine to handle the interrupt event. If you are monitoring more than one condition in your system, your event handler must determine which condition caused the interrupt. Use the *SPE command to determine the instrument that caused the interrupt, then poll the summary registers, then poll condition registers to determine the cause of the interrupt.
Referring to Traces, Measurements, Channels, and Windows Using SCPI

Sometimes in a SCPI program you may need to refer to traces that you have not created. This can be a bit confusing in the VNA. Here are the THREE ways to refer to a specific measurement trace.

**Note:** The terms "Trace" and "Measurement" effectively mean the same thing in this discussion.

1. **The Measurement Name** is picked by you when you first create a trace using the `CALCulate<cnum>:PARameter[:DEFine]:EXTended <Mname>,<param>` command. The measurement name is only used by SCPI.

2. **The Trace Number** is also picked by you when ‘feeding’ a newly-created measurement name to a window number using `DISP:WINDow<wmun>:TRACe<tnum>:FEED`. The trace number is used ONLY by SCPI and is mainly used to refer to traces in the DISPlay node. This is NOT the number that appears as Tr# on the screen. While you can assign any Trace number you want, when a measurement is created from the GUI, the VNA assigns numbers to the traces sequentially, starting with one in each window. Therefore, when there is more than one window, these numbers are not unique.

3. **The Tr#** that appears on the VNA screen is the third and most visible way to refer to a trace. Since we already have a "Trace Number", we call this the **Measurement Number** in the VNA Help file. This number is issued sequentially by the VNA regardless of channel and window. It is therefore unique among all traces. Use `CALC<ch>:PAR:MNUM?` just after the trace is created to read the measurement number.

The concept of the **Active measurement** versus **Selected Measurement** is also a bit confusing. As seen on the screen, the Active measurement has the highlighted Tr#. While there can only be ONE active measurement, every channel has a selected measurement. The target measurement must first be selected before most CALC node settings can be made. There are two ways to select a measurement for each channel:

1. Use `CALC<ch>:PAR:SEL <measName>` which requires the channel number and measurement name.
2. Use `CALC<ch>:PAR:MNUM <measNum>` which requires the channel and measurement (Tr) number.

Here are other relevant commands for referring to traces, measurements, channels, and windows:

- `CALC<cnum>:PAR:TNUMber?` - Returns the Trace Number of the selected trace.
- `CALC<cnum>:PAR:WNUMber?` - Returns the window number of the selected trace.
- `SYSTem:ACTive:CHANnel?` - Returns the number of the active channel. The active channel is the channel number that contains the active measurement.
- **SYSTem:ACTive:MEAS?** - Returns the name of the active measurement. As seen on the screen, the Active measurement has the highlighted Tr#.

- **SYSTem:CHANnels:CATalog?** - Returns the channel numbers currently in use.

- **SYSTem:WINDows:CATalog?** - Returns the window numbers that are currently being used.

- **SYSTem:MEAS:CATalog? [chan]** - Returns ALL measurement numbers, or optionally measurement numbers from a specified channel.

- **SYSTem:MEAS<n>:NAME?** - Returns the name of the specified measurement (Tr#) number.

- **SYSTem:MEAS<n>:TRACe?** - Returns the trace number of the specified measurement number.

- **SYSTem:MEAS<n>:WINDow?** - Returns the window number of the specified measurement number.
SCPI Control of USB and GPIB Devices Connected to a VNA

See GPIB Devices below.

USB Devices

The following procedure is the USB equivalent of the VNA "GPIB Pass-Through" feature. This allows you to send SCPI commands from a remote PC to a USB device (such as a USB power sensor) that is connected to the VNA.

The PC must have the Keysight I/O Libraries installed.

1. On the VNA, press Utility, System, System Setup, then Remote Interface...

2. Check SICL Enabled.

3. On your PC, start Keysight Connection Expert which is the wizard for Keysight I/O Libraries.

4. In the dialog, select the Manual Configuration tab.

5. Click Remote USB interface.

6. In the Remote USB Interface dialog (shown above), do EITHER of the following:
6. Select **Hostname**, then enter the Full Computer Name of the VNA.

7. Select **IP address**, then enter the IP address of the VNA.

In the above dialog, the VISA interface ID is **USB1**. Therefore, a VISA program on the PC could send commands to the sensor that is connected to the VNA at the VISA resource string beginning with **USB1::<interface>**.

7. Click the **Accept** button. The **Instruments** tab is selected listing available instruments.

8. Click on the device you want to communicate with (U2000A in this example) then click **Send Commands To This Instrument**.

9. Type a command, then click **Send Command** or **Read Response**.

---

**SCPI Control of GPIB Devices Connected to a VNA**

Access the VNA GPIB System Controller port from your PC as though it were a local GPIB card on your PC.

This allows you to send SCPI commands from a remote PC to a GPIB device such as a Power Meter, that is connected to the VNA.

The PC must have the Keysight I/O Libraries installed.

1. On the VNA, press **Utility**, **System**, **System Setup**, then **Remote Interface**...

2. Check SICL Enabled.

3. On your PC, start Keysight Connection Expert which is the wizard for Keysight I/O Libraries.

4. In the dialog, select the **Manual Configuration** tab.
5. Click **Remote GPIB interface**.

![Remote GPIB interface dialog](image)

In the above dialog, use the default settings EXCEPT where specified here:

- **Interface name on remote host** - Enter ‘gpib0’
- Do **EITHER** of the following:
  - Select **Hostname**, then enter the Full Computer Name of the VNA.
  - Select **IP address**, then enter the IP address of the VNA.
- Click the **Accept** button.

Then access the VNA GPIB System Controller port using the SICL interface ID shown in the dialog (gpib1 in above dialog image).

For example, with a power meter at address 13, you would open a VISA session on the PC to `GPIB1::13::INSTR` and then send commands to it while the device is connected to the VNA Controller port.

**Important:** Close any open VISA session handles to that interface before the VNA controls device.
Configure for SCPI LAN using SICL / VISA

- VNA Supported Interfaces
- Keysight I/O Libraries
- SICL / VISA Programs Running on the VNA
- Configure the VNA for SICL / VISA
- Configure the External Controller

Other Topics about GPIB Concepts

VNA Supported Interfaces

The VNA supports the following interfaces for SICL / VISA communication:

- **LAN** - as a remote GPIB interface. The VNA LAN is presented as a virtual GPIB interface. It does NOT support simple TCPIP-based control. Therefore, when configuring the Keysight IO libraries on your PC, add a REMOTE GPIB interface, which uses the LAN client interface.

- **GPIB** - requires that your external controller have a GPIB card.

**Note:** For optimum LAN interface performance, use COM to control the VNA. SCPI commands can be sent to the VNA using the COM SCPIStringParser object.

The following interfaces are NOT supported:

- **USB**
- **Serial**
Important Note:

To enable VISA or SICL communication over LAN, you must do the following:

1. On the VNA, click **Utility, System, System Setup...**, then select **Remote Interface...**.

2. Check **SICL Enabled**. To automatically enable SICL when the VNA is booted, check **Automatically enable on Startup**.

3. Click **OK**.

The VNA is now ready to be controlled over LAN.

Learn more about this dialog box.

**Keysight I/O Libraries**

The Keysight I/O libraries includes the drivers to allow you to communicate with Keysight test instruments. Every VNA is shipped with the Keysight I/O libraries installed. We recommend you do NOT upgrade the Keysight I/O libraries on the VNA as unexpected results may occur. If you choose to upgrade the Keysight I/O libraries on the VNA, do NOT change the default folder path in the InstallShield Wizard.

To communicate with the VNA, the Keysight I/O libraries must also be installed on your external controller. To purchase the Keysight I/O libraries, or download a free upgrade, go to [www.Keysight.com](http://www.Keysight.com) and search for IO Libraries. Scroll to find Software, Firmware & Drivers.

**SICL / VISA Programs Running on the VNA**

You can run your SICL / VISA program on the VNA to control the VNA. Although the Keysight I/O libraries are already installed on the VNA, it is configured as the **Host**. You must also configure a SICL or VISA LAN **Client** interface on the VNA, specifying the LAN hostname of that same VNA.

If your program uses the COM interface to VISA, and is compiled on a PC with the Keysight IO Libraries Suite (version 14 or later), and the resulting executable is copied and run on the VNA, it will produce a “type mismatch error”. This is because the VNA has the ‘M’ version of Keysight I/O libraries. The following Visual Basic code is an example of how to avoid this error when communicating with the VNA from within the VNA:

```vbnet
Dim rm As IResourceManager
Dim fmio As IFormattedIO488
Set rm = CreateObject("AgilentRM.SRMCls")
```
Controlling the VNA over LAN while controlling other instruments over GPIB

The VNA can NOT be both a controller and talker/listener on the same GPIB bus. Using SICL / VISA, you can use LAN to control the VNA, leaving the VNA free to use the rear-panel GPIB interface to control other GPIB devices.

Configure the VNA for SICL / VISA

1. Open the Keysight Connection Expert.

2. Select each GPIB Interface and verify (or make) the default settings in the following table. These settings are REQUIRED when using a 82357A USB / GPIB Interface with the VNA.

3. When complete, click Accept to close the Keysight Connection Expert.

<table>
<thead>
<tr>
<th>VISA Interface Name</th>
<th>SICL Interface Name</th>
<th>Dialog box title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPIB0</td>
<td>gpib0</td>
<td>GPIB Using NI-488.2</td>
<td>VNA Rear-panel GPIB connector. This GPIB interface can be used to control the VNA OR for the VNA to control external equipment. IT CAN NOT DO BOTH IN THE SAME PROGRAM. Learn more about pass-through options.</td>
</tr>
<tr>
<td>GPIB1</td>
<td>hpib7</td>
<td>Internal Instrument Configuration</td>
<td>Internal interface for programs running on the VNA to control itself.</td>
</tr>
<tr>
<td>GPIB4</td>
<td>inst0</td>
<td>Internal Instrument Configuration</td>
<td>Used for LXI compliance. Do NOT delete this interface.</td>
</tr>
</tbody>
</table>

Configure the External Controller

Please refer to the Keysight I/O libraries documentation to learn how to configure your controller to communicate with the VNA. These links can show you how to find the following VNA information:

- VNA full computer name
- GPIB Address
- IP Address

This example program can help test your VISA configuration.
Beginning in Dec. 2005, Keysight VEE Pro RunTime is installed on new VNAs. This means that programs written with Keysight VEE (.vxe files) can be run directly on the VNA.

VNAs without Keysight VEE installed can go to the Keysight VEE website and download Keysight VEE Pro 6.2 RunTime to the VNA and begin to run VEE programs directly on the VNA. This version does not require Keysight I/O Libraries suite 14. Do NOT upgrade to Keysight I/O libraries suite 14 on the VNA.

With Keysight VEE Pro RunTime installed on the VNA, the following examples can be run directly on the VNA:

- Basic Control of the VNA

For more VEE examples, see the PNA support website.

For more information on Keysight VEE, see www.Keysight.com/find/VEE
This VEE Pro 6.0 example does the following:

- Controls VNA windows and traces.
- Changes stimulus settings.
- Measures all four S parameters.
- Create markers and displays marker readout.

If this Help file is on a VNA and VEE Pro RunTime is installed, then:

1. Run the BasicControl.vxe example

2. Then click Open on the following dialog box to run the program.

Otherwise, you can modify the example program using VEE, save the VEE BasicControl.vee

Learn how to run this program as a Macro on the VNA.

The following dialog box will be visible on the VNA when the example program is running.

- Click Fwd to activate the Forward (S11 and S21) measurements.
- Click **Rev** to activate the Reverse (S22 and S12) measurements.
- Click **Update Markers** to sweep the VNA.
- Type values to change Marker Frequencies.
ECal with Confidence Check using VEE

This VEE Pro 6.0 example performs an ECal and subsequent ECal confidence Check.

If this Help file is on a VNA and VEE Pro RunTime is installed:

- Run the .vxe example
- Then click Open on the following dialog box to run the program.

Or to modify the example program using VEE, save the VEE BasicControl.vee

Learn how to run this program as a Macro on the VNA.

The following dialog box will be visible on the VNA when the example program is running.

- Click Fwd to activate the Forward (S11 and S21) measurements.
- Click Rev to activate the Reverse (S22 and S12) measurements.
- Click Update Markers to sweep the VNA.
- Type values to change Marker Frequencies.
Interface Control

The Interface Control feature allows you to send remote commands and data to the following VNA rear-panel Interfaces: GPIB, Material Handler I/O, Test Set I/O, and Auxiliary I/O.

- Overview
- How to Access Interface Control Settings
- Interface Control Dialog Box
- Z5623A H08 Test Set Commands

Other System Configuration Topics

Overview

The Interface Control feature allows you to send data to control external equipment such as GPIB instruments, a material handler, test set, or other equipment, without needing to create a remote program. The VNA manages the timing and required interface setup. See Rear Panel Tour.

- A unique set of control data can be sent for each channel. In addition, a unique set of control data can be sent before the channel sweep starts, and after the sweep ends.
- Interface Control settings can be saved and recalled from the Interface Control dialog box, or with Instrument State Save and Recall.
- Interface Control settings can be copied to other channels using Copy Channels.
- Control data can only be WRITTEN to the interfaces, NOT READ from the interfaces.
- Control data is sent in the following order. This order cannot be changed.

1. GPIB Interface
2. Material Handler Interface
3. Test Set Interface
4. Aux Interface
5. Dwell Time
How to access Interface Control settings

Using Hardkey/SoftTab/Softkey

1. Press Setup > Internal Hardware > Interface Control...

Using a mouse

1. Click Instrument
2. Select Setup
3. Select Internal Hardware
4. Select Interface Control...

Interface Control dialog box help

GPIB and VISA Commands

Notes:

- GPIB instruments CAN be connected to the VNA using a USB/GPIB adapter.
- Any type of interface (LAN, USB, GPIB) is available through the VISA connection string.
- GPIB/VISA Queries are NOT supported. Commands can be sent only.

Enable Control  Enables and disables sending commands out the GPIB or VISA interface.

Multi-line edit control  Each line contains a GPIB or VISA command using the following syntax:

```
address  command
```

Where:

address

- A number between 0 and 31. The VNA will look through all of the GPIB interfaces for an instrument connected to the specified address. If an instrument with that address is not recognized, an error is returned.

- A valid VISA connection string.
**command**: a SCPI command, with or without enclosing quotes. Enclosing quotes are ignored.

Address and command are separated by at least one space.

Commands should be separated by a new line, or carriage return. For example:

```
19 "init:cont off"
16 init:imm
```

```
TCPIP0::141.121.78.100::inst0::INSTR outp:ON
```

The front-panel **Enter** key inserts a new line into the field.

The number of GPIB/VISA commands that can be entered is limited only by the available memory of the VNA.

See [Z5326A H08 Test Set Commands](#).

**Material Handler I/O**

**Enable Control**: Enables and disables sending data out the **Material Handler I/O connector** (PNA), **Handler IO Connector (ENA)**

**Ports A, B, C, D**: Sends values to the respective Handler I/O port. Although ports C and D are normally bidirectional, ONLY Output mode is allowed using the Interface Control feature. It cannot read from these, or any other, ports.

**Test Set I/O**

Note: The VNA has a separate interface for **controlling the E5091A Test Set**.

**Enable Control**: Enables and disables sending data out the **External Test Set I/O connector**.

**Multi-line edit control**: Each line contains a Write command using the following syntax:

```
address.value
```

Where:

- **address**: any positive integer.
- **value**: numeric character. Entries that require **alpha** characters should use the **GPIB**
Address and value are separated by a period. For example:

```
18.2
27.3
```

Entries should be separated by a new line, or carriage return. The VNA front-panel Enter key inserts a new line into the field.

All entries are sent out the Test Set I/O port using the WriteData Method.

The number of entries is limited only by the available memory of the VNA.

**Aux I/O Output Voltage**

**Note:** The 9-pin PWR I/O (Power I/O) D connector on the rear-panel replaces much of the functionality of the AUX I/O connector on older VNA models. The Power I/O voltages can be set using the following methods:

- `CONTrol:AUXiliary:OUTPut:VOLTage` or `put_OutputVoltage Method` (no GUI equivalent, global scoped, and settings not saved as part of the instrument state)
- `SOURce:DC:START` and `SOURce:DC:STOP` (DC Source dialog is the GUI equivalent, channel scoped, and settings saved as part of the instrument state)
- Interface Control dialog (no remote equivalent, channel scoped, and settings saved as part of the instrument state)

To avoid unexpected behavior, choose one method only to set the Power I/O voltages.

**Enable Control** Enables and disables sending data out the Auxiliary I/O connector.

**DAC1, DAC2** Sets voltages on the Aux I/O connector pins 2 (DAC1) and pin 3 (DAC2).

**Dwell After Command** Specifies a wait time, in milliseconds, after all commands to all interfaces are sent. Any positive integer is allowed. This is used to allow all external devices to settle before beginning a measurement. An erratic trace could indicate that more settling time is necessary.

**Reset All** Sets ALL fields on ALL channels to their default values.

**Save and Recall** Saves and recalls the contents of this dialog box. If the Interface Control dialog box is populated with settings during an Instrument State Save, the settings are automatically recalled with the Instrument State settings.

Interface control uses an *.xml file type. An example file is stored on the VNA hard drive. You can recall it into the dialog, or you can open and edit it with a word processor, such as Word Pad.
OK  Applies the settings and closes the dialog box.

Cancel  Does not apply changes that were made, and closes the dialog box.

Z5623A H08 Test Set Commands

The following table lists the commands that are used to control the popular Keysight Z5623A H08 Test Set. These commands can be entered into the GPIB Interface control.

<table>
<thead>
<tr>
<th>Connection Path</th>
<th>Test Set Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reflection to Port 1</td>
<td>refl_01</td>
</tr>
<tr>
<td>Reflection to Port 2</td>
<td>refl_02</td>
</tr>
<tr>
<td>Reflection to Port 3</td>
<td>refl_03</td>
</tr>
<tr>
<td>Reflection to Port 4</td>
<td>refl_04</td>
</tr>
<tr>
<td>Reflection to Port 5</td>
<td>refl_05</td>
</tr>
<tr>
<td>Reflection to Port 6</td>
<td>refl_06</td>
</tr>
<tr>
<td>Reflection to Port 7</td>
<td>refl_07</td>
</tr>
<tr>
<td>Reflection to Port 8</td>
<td>refl_08</td>
</tr>
<tr>
<td>Transmission to Port 1</td>
<td>tran_01</td>
</tr>
<tr>
<td>Transmission to Port 2</td>
<td>tran_02</td>
</tr>
<tr>
<td>Transmission to Port 3</td>
<td>tran_03</td>
</tr>
<tr>
<td>Transmission to Port 4</td>
<td>tran_04</td>
</tr>
<tr>
<td>Transmission to Port 5</td>
<td>tran_05</td>
</tr>
<tr>
<td>Transmission to Port 6</td>
<td>tran_06</td>
</tr>
<tr>
<td>Transmission to Port 7</td>
<td>tran_07</td>
</tr>
<tr>
<td>Transmission to Port 8</td>
<td>tran_08</td>
</tr>
<tr>
<td>Reset</td>
<td>*rst</td>
</tr>
<tr>
<td>Reflection Termination</td>
<td>*r_term</td>
</tr>
<tr>
<td>Transmission Termination</td>
<td>*t_term</td>
</tr>
<tr>
<td>All Termination</td>
<td>*all_term</td>
</tr>
</tbody>
</table>
External Test Set I/O Connector

General Description

This DB-25 female connector is used to control external test sets. The external test set bus consists of 13 multiplexed address and data lines, three control lines, and an open-collector interrupt line. The Test Set IO is not compatible with the 8753 test sets.

You can change the settings on the External Test Set IO connector through SCPI and COM programming commands. The settings are NOT accessible through the front-panel keys or display menu.

Notes:

- The External Test Set pin settings are NOT affected by Instrument State Save/Recall or Instrument Preset.
- At VNA Power Up and return from Hibernation, the External Test Set bus data lines, address lines, and control lines are set HIGH, and no strobe lines are pulsed.

Caution: Do not mistake this connector with a Parallel Printer port. A printer may be damaged if connected to this port.

Other System Configuration Topics

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SEL0</td>
<td>Test set select bit 0; tied to GND</td>
</tr>
<tr>
<td>2</td>
<td>Sweep Holdoff In</td>
<td>TTL input - state may be read with SCPI or COM command</td>
</tr>
<tr>
<td>3</td>
<td>AD12</td>
<td>Address and latched data</td>
</tr>
<tr>
<td>4</td>
<td>AD10</td>
<td>Address and latched data</td>
</tr>
<tr>
<td>5</td>
<td>AD9</td>
<td>Address and latched data</td>
</tr>
<tr>
<td></td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>-------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Address and latched data</strong></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>GND 0V</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>LAS TTL output Low = Address Strobe</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>AD4 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>AD3 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>AD2 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>GND 0V</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Interrupt In TTL input state may be read with a SCPI or COM command</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>No connect CAUTION: Older VNAs have +22v on this line; this will damage a printer.</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>SEL1 Test set select bit 1; tied to GND</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>SEL2 Test set select bit 2; tied to GND</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>AD11 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>SEL3 Test set select bit 3; tied to GND</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>AD7 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>AD6 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>AD5 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>AD0 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>AD1 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>LDS TTL output active low data strobe</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>RLW TTL output high-read, low write</td>
<td></td>
</tr>
</tbody>
</table>

**SEL0-SEL3 (pins 1,15,16,18)**

**Description**

Selects addresses of test sets that are "daisy chained" to this port. The select code is set to zero at the VNA connector and is incremented by one as it goes through each successive external test set. Therefore, the first test set in the chain has address zero and so on, for up to 16 test sets.

**HW Details**
Connected to ground inside the VNA.

**Timing**

None

---

**Sweep Holdoff In (pin 2)**

**Description**

Input line used by the test set for holding off a sweep. Holding off a sweep is one way of introducing a delay that allows an external device to settle before the VNA starts taking data. You must write a program that will query the line and perform the delay. The program needs to query the line and keep VNA from sweeping while the line remains low. When a subsequent query detects that the line went high the program would then trigger the VNA to start the sweep.

Use either Single or External trigger mode to control the VNA sweep.

**HW Details**

This pin has a series 215-ohms resistor followed by 4.7k-ohm pull-up and then an "ABT" TTL buffered register.

**Timing**

This input is not latched by the VNA hardware. Therefore the input level must be held at the desired state by the test set until it's read by your program.

---

**AD0-AD12 (pins 3-6, 9-11, 17, 19-23)**

**Description**

Thirteen lines are used to output data addresses or input / output data. Several SCPI and COM commands are available for reading and writing to these lines. You can choose to use commands where the VNA provides the appropriate timing signals needed for strobing the addresses and data. Or you can choose to control the timing signal directly. The timing signals are RLW, LAS and LDS. If you decide to do direct control refer to the corresponding SCPI and COM command details. Close attention to detail is needed to insure the desired results.

After a write command, lines AD0-AD12 are left in the state they were programmed. Default setting for Mode is Read / Input).

After a read command, lines AD0-AD12 are left in input mode. While in this mode an external test set attached to the IO is free to set the level on each line.

**HW Details**
Each of these I/O pins has a series 215-ohm resistor followed by 4.7k-ohm pull-up resistor.

Write/Read is implemented by an output tri-state TTL buffer / latch for latching and enabling write data in parallel with a TTL input buffer for reading.

**Timing**

Output Address and data setup and hold times are 1us minimum.

---

**Address & Data I/O Write**

---

**Address & Data I/O Read** - Data must be valid for 1us before and after strobe

---

**GND (pins 7, 12)**

**Description**

Two ground pins used as ground references by the test set.

**HW Details**

Connected to digital ground.

**Timing**

None.
LAS (Low Address Strobe) (pin 8)

Description

This line has two behaviors that are command dependent. Refer to the SCPI and COM commands for further details.

In one behavior LAS is one of the lines used by the VNA to provide appropriate timing for writing Address and Data to the Test Set. In this case LAS is controlled automatically by the VNA and is intended to be used as the strobe for the Address. When LAS is low, lines AD0 - AD12 represent the Address. LAS will return to its normally high state when the transaction is finished.

In the second behavior the VNA will NOT provide appropriate timing. In this case LAS is controlled directly by the user through a SCPI or COM command. When the transaction is finished LAS is left set to the state it was programmed to until another command changes it. (Default for LAS is TTL High).

HW Details

This output pin is driven by a TTL latched buffer with a series 215-ohm resistor followed by 2.15k-ohm pull-up.

Timing

Strobe length, setup and hold times are all 1us minimum.

See the description for AD0-AD12 for more timing information.

Interrupt In (pin 13)

Description

Query this line with a SCPI or COM command.

HW Details

This line is a non-latched TTL input, has series 215-ohms followed by 4.64k-ohm pull-up.

Timing

The Test Set must maintain at the desired TTL level until its read.

(pin 14) No Connect (previously +22V)

WARNING: Early versions of the VNA had +22v on this pin. Connecting a printer to this port will usually damage the printer.

Description
+22V, 100mA max. The 25-pin D connector is the same as a computer parallel printer port connector. Pin (14) corresponds to a printer's "autofeed" line. **Connecting a printer to this port will damage the printer if +22v is present** since printers requires less than 5V on all control lines.

**HW Details**

No connect

**Timing**

None

---

**LDS (Low Data Strobe) (pin 24)**

**Description**

This line has two behaviors that are command dependent. Refer to the External Test Set IO SCPI and COM commands for further details. (Default setting for LDS is TTL High)

In one behavior LDS is one of lines used by the VNA to provide appropriate timing for writing Address and Data to the Test Set. In this case LDS is controlled automatically by the VNA and is intended to be used as the strobe for the Data. When LDS is low, lines AD0 - AD12 represents Data. LDS will return to its normally high state when the transaction is finished.

In the second behavior the VNA will NOT provide appropriate timing. In this case LDS is controlled directly by the user through a SCPI or COM command. When the transaction is finished the LDS is left set to the state it was programmed to.

**HW Details**

This output pin is driven by a TTL latched buffer with a series 215-ohm resistor followed by 2.15k-ohm pull-up.

**Timing**

Strobe length, setup and hold times are all 1us minimum.

See the description for **AD0-AD12** for more timing information.

---

**RLW (pin 25)**

**Description**

This line is the output for the Read Write signal. It has two behaviors that are command dependent. Refer to the External Test Set IO SCPI and COM commands for further details. (Default setting for RLW is TTL High)

In one behavior RWL is controlled automatically by the VNA during a Read Write operation. When RLW is low, lines AD0 - AD12 represent output Data. When RLW is high, the lines represent input Data.

In the second behavior the VNA does NOT provide the timing. The user must control it directly through the SCPI or COM command. In this case the line is left set to the state it was programmed to.

**HW Details**

This pin is a TTL latched output with a series 215-ohm resistor followed by 2.15k-ohm pull-up resistor.
Timing

Strobe length, setup and hold times are all 1us minimum.

See the description for AD0-AD12 for more timing information.
Material Handler I/O Connector

This rectangular 36-pin female connector provides communication signals between the VNA and a material parts handler. You can change the settings on the Material Handler IO connector using SCPI and COM commands. The settings are NOT accessible through the front-panel keys or display menu.

- Overview - Controlling a Material Handler
- Pin Assignments
- Pin Descriptions
- Timing Diagrams
- Input Output Electrical Characteristics

Note: On early VNAs this connector is labeled "GPIO". It is covered to indicate that the connector is not functional.

Overview - Controlling a Material Handler

The VNA is capable of interacting with an external material handler or part handler. This allows the VNA to be used in an automated test environment, where devices to be tested are inserted into a test fixture by a part handler, and sorted into pass/fail bins by the handler after testing is complete. By connecting the part handler to the VNA Auxiliary or Material Handler I/O ports, the VNA and part handler can synchronize their activities in a way that makes automated testing possible.

VNA and Part Handler Preparation

1. **Define the measurements** you want to make.

2. **Define limits** for each of the measurements.

3. Configure the VNAs Material Handler port so that it is compatible with your part handler. This usually involves setting the handler logic, pass/fail logic, pass/fail scope, and pass/fail mode. These settings are made remotely using SCPI or COM commands.

4. Use a cable to connect the VNA to your part handler.

5. Put the VNA in **External Trigger** mode.
6. Load parts in handler per manufacturer instructions.

**Note:** The Material Handler configuration settings REMAIN after an Instrument Preset. The settings will revert to their default settings ONLY after the VNA is restarted, or until they are changed by you. Material Handler settings are saved and recalled with Instrument State.

**Flow Diagram**

The following diagram and descriptions summarizes the events that occur during automated testing. 'DUT' refers to Device Under Test.

### Text Descriptions

**0.** (Optional). The VNA sends values out the Material Handler to configure external instruments. The A,B,C, and D ports of the Material Handler can be used to control devices used in testing, such as step attenuators, part handlers, or even the DUT itself. Also, the DAC1 and DAC2 lines on the Power I/O connector can be used to provide bias voltages for devices and instruments. If you wish to use the Material Handler for testing, you will need to write a program to send values out the various lines and ports, as there is no activity on these lines by default.

**1.** The part handler receives a **Ready for Trigger** signal from the VNA. This indicates that the VNA is properly configured and ready to take a measurement.

**2.** The part handler sends an **External Trigger** signal to the VNA. This signals that the part handler has settled, and allows the VNA to begin taking measurements.
3. The VNA takes measurements on all triggerable channels.

4. The Index line on the material handler goes to a Low state (default), which means that all required data has been collected by the VNA.

5. The part handler removes the DUT from the test fixture, and inserts a new DUT into the fixture. This operation is often referred to as part handler indexing. The device just tested is staged (removed from the fixture and prepared for binning), and the next part to be tested is put into the fixture. The removed DUT cannot be assigned to a Pass/Fail bin yet, as the Pass/Fail status is not available.

6. The VNA sends the Pass/Fail Status.

7. The VNA sends the Pass/Fail Strobe meaning that the Pass/Fail status has been determined.

8. The part handler reads the Pass/Fail Status line.

9. The part handler bins the staged part based on the Pass/Fail Status.

10. The test process repeats at step 1, waiting for Ready for Trigger from the VNA.

### Material Handler IO Pin Assignments

<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ground</td>
</tr>
<tr>
<td>2</td>
<td>/ INPUT1</td>
</tr>
<tr>
<td>3</td>
<td>/ OUTPUT1</td>
</tr>
<tr>
<td>4</td>
<td>/ OUTPUT2</td>
</tr>
<tr>
<td>5</td>
<td>/ Output port A0</td>
</tr>
<tr>
<td>6</td>
<td>/ Output port A1</td>
</tr>
<tr>
<td>7</td>
<td>/ Output port A2</td>
</tr>
<tr>
<td>8</td>
<td>/ Output port A3</td>
</tr>
<tr>
<td>9</td>
<td>/ Output port A4</td>
</tr>
<tr>
<td>10</td>
<td>/ Output port A5</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>11</td>
<td>Output port A6</td>
</tr>
<tr>
<td>12</td>
<td>Output port A7</td>
</tr>
<tr>
<td>13</td>
<td>Output port B0</td>
</tr>
<tr>
<td>14</td>
<td>Output port B1</td>
</tr>
<tr>
<td>15</td>
<td>Output port B2</td>
</tr>
<tr>
<td>16</td>
<td>Output port B3</td>
</tr>
<tr>
<td>17</td>
<td>Output port B4</td>
</tr>
<tr>
<td>18</td>
<td>Ext. Trigger</td>
</tr>
<tr>
<td>19</td>
<td>Output port B5</td>
</tr>
<tr>
<td>20</td>
<td>Output port B6 or Index Signal (select by command)</td>
</tr>
<tr>
<td>21</td>
<td>Output port B7 or Ready for Trigger (select by command or dialog box)</td>
</tr>
<tr>
<td>22</td>
<td>In/Out port C0</td>
</tr>
<tr>
<td>23</td>
<td>In/Out port C1</td>
</tr>
<tr>
<td>24</td>
<td>In/Out port C2</td>
</tr>
<tr>
<td>25</td>
<td>In/Out port C3</td>
</tr>
<tr>
<td>26</td>
<td>In/Out port D0</td>
</tr>
<tr>
<td>27</td>
<td>In/Out port D1</td>
</tr>
<tr>
<td>28</td>
<td>In/Out port D2</td>
</tr>
<tr>
<td>29</td>
<td>In/Out port D3</td>
</tr>
<tr>
<td>30</td>
<td>Port C Status</td>
</tr>
<tr>
<td>31</td>
<td>Port D Status</td>
</tr>
<tr>
<td>32</td>
<td>Output Port Write Strobe</td>
</tr>
<tr>
<td>33</td>
<td>Pass/Fail</td>
</tr>
<tr>
<td>34</td>
<td>Sweep End</td>
</tr>
</tbody>
</table>
Pin Descriptions

**Input1**

When this Input line receives a Low pulse from the material handler, data is latched on the OUTPUT1 and OUTPUT2 lines. See OUTPUT1|2 Data Output Write Timing

The Input Line activity can be read:

```
SCPI
CONTrol:HANDler:INPut?
```

**Output1, Output2**

See OUTPUT1|2 Data Output Write Timing

The current state of these latched TTL outputs may be set High or Low (Default setting) using the (non-user) SCPI put_Output (COM) commands.

The next state (following a negative edge on the INPUT1 line) may be pre-loaded to High or Low (Default setting) using the user commands.

For example, on the next negative pulse on the INPUT1 line, you want the OUTPUT1 line to go from 0 to 1. To do this:

```
CONT:HAND:OUTP1:DATA 0  'Force the OUTPUT1 line to 0
CONT:HAND:OUTP1:USER 1  'Set the OUTPUT1:USER buffer to 1, indicating the next state
```

Write User Data
Read last value written
Write non-user data
Read last value written
**Output Ports A and B**

These two general purpose, 8-bit output ports are used to write data to the material handler. When any line changes state, all output lines are latched to the I/O connector as the **Output Write Strobe** goes Low.

The default state for data is Low.

**See Data Output Write Timing Diagram**

**Set Port Logic:**

The logic for the data lines can be set to either: Positive (1 = High) or Negative (1 = Low). This setting affects all data ports. They cannot be set independently.

```
SCPI
CONTrol:HANDler:LOGic

COM
PortLogic Property
```

**Combine to read or write data to Port F:**

Ports A and B can be virtually combined to write data to one 16-bit I/O port **F**.

```
SCPI
CONTrol:HANDler:F <num>

COM
put Port (F)
```

---

**Input/Output Ports C and D**

These two general purpose 4-bit Input/Output ports are used to write data (Output) or read data (Input). These lines could be used to write to an external device such as a step attenuator.

When any line changes state, all output lines are latched to the I/O connector as the **Output Write Strobe** goes Low. **See Data Output Write Timing.**

**Set Input | Output Mode:**

Each port may be independently defined as Output or Input.

```
SCPI
CONTrol:HANDler:C:MODE

COM
PortMode Property
```
Set Port Logic:

The logic for the data lines can be set to either: Positive (1 = High) or Negative (1 = Low). This setting affects all data ports. They cannot be set independently.

Read or write data:

Ports C and D can be virtually combined to read or write data to one 8-bit I/O port \( E \). When combined, both C and D ports must be set to either INPUT or OUTPUT mode.

Port C Status, Port D Status

These two output lines indicate the Read / Write mode of the C and D ports.

- A Low level indicates that the associated port is in \textbf{INPUT} mode (read only).
- A High level indicates that the associated port is in \textbf{OUTPUT} mode (write only).

These logic of these status outputs cannot be changed.

See Input/Output Ports C and D to learn how to set I/O Mode

See Data Output Write Timing
**Output Port Write Strobe**

This Output line goes Low to write data from Ports A and B and Ports C and D when a change is detected on any of the data lines.

These logic of this strobe output cannot be changed.

See Data Output Write Timing

**External Trigger**

When trigger source is set to external, this Input line accepts a trigger signal from the material handler. This usually means that a part is in place and ready to be tested.

See Trigger Timing Diagram

**Index**

A Low signal (default) on this Output line indicates to the material handler that the measurement is complete. This usually means that the handler can connect the next device. However, measurement data is not available until data is calculated. See Trigger Timing Diagram.

**Set Function:**

This line also serves as a data line. Set the function using the following commands:

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTrol:HANDler:INDex:STATe</td>
<td>IndexState</td>
</tr>
</tbody>
</table>
### Ready for Trigger

When this output line goes low (default), it indicates to the material handler that the VNA is ready for a trigger signal.

**See Trigger Timing Diagram**

**See Pass/Fail Timing Diagram**

**Set Function:**

This line also serves as a data line. Set the function using the following commands:

```
SCPI
CONTrol:HANDler:RTTrigger:STATe
```

### Pass/Fail State

This Output line indicates to the handler whether the limit test has passed or failed.

Pass/Fail state is valid only when the limit test function is ON and while Pass/Fail strobe line is Low. **See Pass/Fail Timing Diagram**

**Set Pass / Fail Logic:**

- Positive Logic: High=Pass, Low=Fail. (Default setting)
- Negative Logic: High=Fail, Low=Pass.

```
SCPI
CONTrol:HANDler:PASSfail:LOGic
```

### Set Default Conditions:

- **PASS** - the line stays in PASS state. When a device fails, then the line goes to fail after the Sweep End line is asserted.

- **FAIL** - the line stays in FAIL state. When a device passes, then the line goes to PASS state after the Sweep End line is asserted.

- **No Wait** - the line stays in PASS state. When a device fails, then the line goes to fail IMMEDIATELY. (Default setting)
Set Pass / Fail Scope:

- **Channel scope**: The line resets to the default state after the measurements on a channel have completed.
- **Global scope**: The line resets to the default state after the measurements on all triggerable channels have completed. (Default setting)

### Pass/Fail Write Strobe

A Low pulse indicates that **Pass/Fail** line is valid and the Pass / Fail State is output to the material handler.

The Pass/Fail Strobe is fixed in duration and timing. However, when the strobe occurs depends on the Pass/Fail Mode and Pass/Fail Scope (Channel or Global) settings. [See Pass/Fail State](#)

[See Pass/Fail Timing Diagram](#)

### +5V

+5V nominal output (100mA max).

Protected by self-healing fuse.
**Sweep End**

This output line indicates the status of the VNA sweep. The sweep includes sweeping the source and taking data.

- **Low** (falling edge) indicates that the specified sweep event has finished. This does NOT indicate that all calculations have finished.

- **High** indicates that the specified sweep event is active.

**See Trigger Timing Diagram**

**Set Sweep Event Mode:**

- **Sweep**: indicates that a single source sweep has finished.

- **Channel**: indicates that a single channel has finished.

- **Global**: indicates that all enabled channels have finished. (Default setting)

```
SCPI
 CONTol:HANDler:SWEepend

COM
 SweepEndMode Property
```

**Timing Diagrams**

**Trigger Timing**

![Trigger Timing Diagram](image)

- **T1**: External Trigger (to VNA)
- **T2**: Index (from VNA) meas. complete
- **Sweep End**: (from VNA) sweep complete
- **Ready for Trigger**: (from VNA)
All signals are active low.

**T1 = 1 \mu s**  External Trigger pulse width

**T2 > 10 \mu s** Sweep End pulse width (both High and Low)

---

### Pass / Fail Timing

- **Pass / Fail Status**
- **/ Pass / Fail Write Strobe**
- **/ Ready for Trigger**

---

**T1 = 1 \mu s**  Pulse width and response time of Pass / Fail Strobe

---

### Ports A-F(H) Data Output Write Timing

- **Ports A-F (H) any data line changing level**
- **/ Write Strobe**

---

**T1 = 1 \mu s**  Write Strobe response time

**T2 = 1 \mu s**  Write Strobe pulse width
OUTPUT1|2 Data Output Write Timing

The old state to new state transition can be either low to high (as shown) or high to low.

\[ T_1 = 0.6 \, \mu s \]  Output1|2 response time

\[ T_2 = 1 \, \mu s \]  Input1 Strobe pulse width

Input / Output Electrical Characteristics

All Material Handler I/O Input and Output lines are TTL compatible.

Input and Input/Output lines

Lines carrying information IN (or bidirectional) to the VNA from the material handler.

<table>
<thead>
<tr>
<th>Maximum Input Voltages:</th>
<th>-0.5 V to 5.5 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>TTL High level:</td>
<td>2.0 V to 5.0 V</td>
</tr>
<tr>
<td>TTL Low level:</td>
<td>0 V to 0.5 V</td>
</tr>
</tbody>
</table>

Output Lines

Lines carrying information OUT of the VNA to the material handler.

<table>
<thead>
<tr>
<th>Output Current</th>
<th>Maximum Output Current:</th>
<th>-10 mA to 10 mA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TTL High level:</td>
<td>-5 mA</td>
</tr>
<tr>
<td></td>
<td>TTL Low level:</td>
<td>3 mA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output Voltage</th>
<th>TTL High level:</th>
<th>2.0 V to 3.3 V</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TTL Low level:</td>
<td>0 V to 0.8 V</td>
</tr>
</tbody>
</table>
Pulse I/O Connector

The models offer the 15 pin D connector provides access to Pulse Modulators and Generators.

- See IF Path Configuration Dialog and block diagram, which includes the Pulse Modulators and Generators.
- See the Integrated Pulsed Application

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IFGateAIn</td>
<td>IF pulse gate input A (TTL)</td>
</tr>
<tr>
<td>2</td>
<td>IFGateBIn</td>
<td>IF pulse gate input B (TTL)</td>
</tr>
<tr>
<td>3</td>
<td>IFGateCIn</td>
<td>IF pulse gate input C (TTL)</td>
</tr>
<tr>
<td>4</td>
<td>IFGateDIn</td>
<td>IF pulse gate input D (TTL)</td>
</tr>
<tr>
<td>5</td>
<td>IFGateRIn</td>
<td>IF pulse gate input R (TTL)</td>
</tr>
<tr>
<td>6</td>
<td>DCOM</td>
<td>Ground</td>
</tr>
<tr>
<td>7</td>
<td>PulseSyncIn</td>
<td>Pulse gen. synchronization trigger input (TTL)</td>
</tr>
<tr>
<td>8</td>
<td>RFPulseModIn</td>
<td>RF source pulse modulation drive input (TTL)</td>
</tr>
<tr>
<td>9</td>
<td>DCOM</td>
<td>Ground</td>
</tr>
<tr>
<td>10</td>
<td>Pulse1Out</td>
<td>Hardwired pulse train output #1 (TTL)</td>
</tr>
<tr>
<td>11</td>
<td>Pulse2Out</td>
<td>Hardwired pulse train output #2 (TTL)</td>
</tr>
<tr>
<td>12</td>
<td>Pulse3Out</td>
<td>Hardwired pulse train output #3 (TTL)</td>
</tr>
<tr>
<td>13</td>
<td>Pulse4Out</td>
<td>Hardwired pulse train output #4 (TTL)</td>
</tr>
<tr>
<td>14</td>
<td>N.C.</td>
<td>No connect -- for future use</td>
</tr>
<tr>
<td>15</td>
<td>DCOM</td>
<td>Ground</td>
</tr>
</tbody>
</table>

See Pulse SCPI and COM commands

N1966A Pulse I/O Adapter
This D connector to RF adapter makes accessing the Pulse I/O connector more convenient.
Power I/O Connector

The 9-pin D connector replaces much of the functionality of the AUX I/O connector on older VNA models. See Rear Panel

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+15V</td>
<td>+15V @ 400 mA</td>
</tr>
<tr>
<td>2</td>
<td>-15V</td>
<td>-15V @ 400 mA</td>
</tr>
<tr>
<td>3</td>
<td>AnalogOut1</td>
<td>Controlled from the DC Source Control dialog.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Analog Output Voltage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Programmable +/-10V @100 mA out</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Nominally 0 ohms</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.44mV typical resolution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1MHz BW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Read and write voltage programmatically using:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- CONT:AUX:OUTP1:VOLT (SCPI - read and write)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- get OutputVoltage Method (COM - read)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- put OutputVoltage Method (COM - write)</td>
</tr>
<tr>
<td>4</td>
<td>AnalogOut2</td>
<td>Controlled from the DC Source Control dialog.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Analog Output Voltage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Programmable +/-10V @100 mA out</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Nominally 0 ohms</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.44mV typical resolution</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>1MHz BW</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read and write voltage programmatically using:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CONT:AUX:OUTP2:VOLT (SCPI - read and write)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>get OutputVoltage Method (COM - read)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>put OutputVoltage Method (COM - write)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ACOM</td>
<td>System ground</td>
</tr>
<tr>
<td>6</td>
<td>GndSense</td>
<td>Ground sense for Analog In &amp; Out</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Connected with 51.1-ohm to ACOM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Read voltage programmatically using:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InputVoltageEX Property (COM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CONT:AUX:OUTP3:VOLT (SCPI)</td>
</tr>
<tr>
<td>7</td>
<td>AnalogIn1</td>
<td>Analog input:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+/-10V @ 1.22mV typical resolution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rin &gt;1 M-ohm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BW ~ 1 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ADC conversion time &lt; 1 us typical</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Read voltage programmatically using:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CONT:AUX:INP1:VOLT (SCPI)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InputVoltageEX Property (COM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Voltages on this pin can also be read using ADC receiver measurements. Learn more.</td>
</tr>
<tr>
<td>8</td>
<td>AnalogIn2</td>
<td>Analog input:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+/-10V @ 1.22mV typical resolution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rin &gt;1 M-ohm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BW ~ 1 MHz</td>
</tr>
</tbody>
</table>
ADC conversion time < 1 us typical

Read voltage programatically using:

- `CONT:AUX:INP2:VOLT` (SCPI)
- `InputVoltageEX` Property (COM)

Voltages on this pin can also be read using ADC receiver measurements.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>Power Button</td>
<td>Open collector input</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Active low replicates power button key press.</td>
</tr>
</tbody>
</table>
New Programming Commands

The following are new programming commands for VNA release A.15.40.xx/A.15.50.xx See What's New.

<table>
<thead>
<tr>
<th>X- and Y-Axis Commands</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the current units of the X-axis.</td>
<td>CALCulate:MEASure:X:AXIS:UNIT?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the current units of the Y-axis,</td>
<td>CALCulate:MEASure:Y:AXIS:UNIT?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the Y-axis values for the selected measurement in the current units.</td>
<td>CALCulate:MEASure:Y[:VALues]?</td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Misc. Commands</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the current state of the Modify setting (NONE, NREFlect, or NXTalk).</td>
<td>CALCulate:FSIMulator:CIRCuit:FILE:MODify?</td>
<td>None</td>
</tr>
<tr>
<td>Sets all reflection and crosstalk parameters on the DUT side to zero.</td>
<td>CALCulate:FSIMulator:DRAFt:CIRCuit:FILE:MODify</td>
<td>None</td>
</tr>
<tr>
<td>Returns TRUE if user is only using legacy SCPI fixture commands.</td>
<td>CALCulate:FSIMulator:LEGacy?</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the Primary Clock setting and is controlled by the internal or external pulse generator which is the primary system clock.</td>
<td>SENSE:SWEep:PULSe:PRIMary:CLOCk</td>
<td>None</td>
</tr>
</tbody>
</table>
Returns whether or not the **Enable Remote Drive Access** is checked in the Remote Interface dialog.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:COMMunicate:DRIVe:ENABle?</td>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following are new programming commands for VNA release A.15.30.xx See What's New.

**Fixturing Power Compensation Command**

Adjust the source power compensation for gain/loss through “ALL” fixture components or through “DEEMbed” components only.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:FSIMulator:POWer:COMPensate:MODE</td>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Licensing Command**

Enables/disables showing the licensed features dialog on startup.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:PREFerences:ITEM:DIALog:SHOW:LIcense</td>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Shared Memory Data Transfer Commands**

Closes the specified memory mapped buffer.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:DATA:MEMory:CLOSe</td>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Closes the specified memory file.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:DATA:MEMory:CLOSe:FILE</td>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Creates the memory buffer and saves the buffer to a file.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:DATA:MEMory:COMMit:FILE</td>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This command resets the current repeat index to 0.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:DATA:MEMory:REPeat:RESet</td>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### TOPology Commands

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Saves the current active fixture topology to a file.</td>
<td>CALCulate:FSIMulator:TOPology:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Loads an active fixture topology file.</td>
<td>CALCulate:FSIMulator:TOPology:LOAD</td>
<td>None</td>
</tr>
</tbody>
</table>

### Misc. Commands

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>This query compares the stimulus of &quot;chanNumber&quot; with the stimulus stored in the specified calset.</td>
<td>CSET:VALIDate?</td>
<td>None</td>
</tr>
<tr>
<td>Enable/disable using existing source power calibration array when acquiring calibration standard data:</td>
<td>SENSe:CORRection:COLLect:GUIDed:PCAL:APPLy</td>
<td>None</td>
</tr>
<tr>
<td>Selects between a small- and medium-sized font displayed below the trace area of the display.</td>
<td>SENSe:DISTortion:TABLE:DISPLAY:FONT</td>
<td>None</td>
</tr>
<tr>
<td>Returns the ID string, and optionally visa address, of power meters / sensors that are connected to the VNA via GPIB.</td>
<td>SYStem:COMMunicate:GPIB:PMETer:CATalog?</td>
<td>None</td>
</tr>
</tbody>
</table>

The following are new programming commands for VNA release A.15.05.xx See What's New.
### GCA/GCX Phase Commands
Remaps the source port to Port 1.

<table>
<thead>
<tr>
<th>Command</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:GCSetup:PMAP:SOURce:OVERride</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Misc. Commands
Returns the time the first point of a Time Sweep is measured.

<table>
<thead>
<tr>
<th>Command</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SWEep:TIME:STARt?</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets the time the analyzer takes to complete one sweep.

<table>
<thead>
<tr>
<th>Command</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SWEep:TIME[:STOP]</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

The following are new programming commands for VNA release A.14.80.xx See What's New.

### Automatic Fixture Removal (AFR) Commands
Turns ON or OFF AFR mode conversion.

<table>
<thead>
<tr>
<th>Command</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:ADVanced:MCONversion[:STATe]</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Resets the AFR configuration.

<table>
<thead>
<tr>
<th>Command</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:ADVanced:RESet</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Gets or sets the manual start time in AFR configuration.

<table>
<thead>
<tr>
<th>Command</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:ADVanced:TIME:STARt</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Gets or sets the manual stop time in AFR configuration.

<table>
<thead>
<tr>
<th>Command</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:ADVanced:TIME:STOP</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Gets or sets the manual window type in AFR configuration.

<table>
<thead>
<tr>
<th>Command</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:ADVanced:WINDow:COEFFicient</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Turns ON or OFF the manual window coefficient of the

<table>
<thead>
<tr>
<th>Command</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:ADVanced:WINDow:MANual[:STATe]</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
AFR.

| Sets "System Z0" to Calibration Reference Z0. | AFR:FIXTure:SET:SYSZ[:STATe] | None |
| Specifies whether thru's are used in case of multi-port fixtures. | AFR:FIXTure:USE:THRUs[:STATe] | None |
| Specifies whether the port impedances are normalized in saving the AFR fixture files. | AFR:SAVE:IMPedance:NORMalize[:STATe] | None |
| Measures calibration standard. | AFR:STANdard:MEASure | None |

### GCA/GCX Phase Commands

<p>| Set and read the desired phase to measure compression. | SENSe:GCSetup:COMPression:PHASe:LEVel | None |
| Set and read compression format to be either magnitude, phase, or magnitude and phase. | SENSe:GCSetup:COMPression:PHASe:MODE | None |
| Set and read the state of the mixer reference. | SENSe:GCSetup:MIXer:REFerence | None |
| Set and read the aperture to use when computing the linear input power. | SENSe:GCSetup:POWer:LINear:INPut:COMPute:APERture | None |</p>
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set and read the state of the power smoothing.</td>
<td>SENSE:GCSsetup:SWEep:POWer:SMOoth</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the power smoothing aperture in percent.</td>
<td>SENSE:GCSsetup:SWEep:POWer:SMOoth:APERture</td>
<td>None</td>
</tr>
</tbody>
</table>

### Modulation Distortion Edit Multitone Commands

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set all tone states to on or off.</td>
<td>SOURce:MODulation:FILE:TONE:ALL[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Returns the number of tones.</td>
<td>SOURce:MODulation:FILE:TONE:COUNt?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the tone frequency in Hz relative to the carrier.</td>
<td>SOURce:MODulation:FILE:TONE:FREQuency?</td>
<td>None</td>
</tr>
<tr>
<td>Loads the specified multitone file.</td>
<td>SOURce:MODulation:FILE:TONE:LOAD</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the phase in degrees of the specified tone number.</td>
<td>SOURce:MODulation:FILE:TONE:PHASE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the power in dBm of the specified tone number.</td>
<td>SOURce:MODulation:FILE:TONE:POWer</td>
<td>None</td>
</tr>
<tr>
<td>Saves the specified multitone file.</td>
<td>SOURce:MODulation:FILE:TONE:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Set specified tone state to on or off.</td>
<td>SOURce:MODulation:FILE:TONE[:STATe]</td>
<td>None</td>
</tr>
</tbody>
</table>

### Residual Phase Noise Commands

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the noise type to phase or residual noise.</td>
<td>SENSE:PN:NTYPE</td>
<td>None</td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td>-----------------</td>
<td>------</td>
</tr>
<tr>
<td>Sets and returns receiver at the DUT input for residual phase noise measurements.</td>
<td>SENSE:PN:RESidual:INPut</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns receiver at the DUT output for residual phase noise measurements.</td>
<td>SENSE:PN:RESidual:OUTput</td>
<td>None</td>
</tr>
</tbody>
</table>

**LO Feedthru Monitor Commands**

<table>
<thead>
<tr>
<th>Return the last monitor result of LO feedthru monitor.</th>
<th>SENSE:SA:COHerance:LO:FTHRu:MONitor:LAST:FAILED</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return the last monitor value of LO feedthru monitor.</td>
<td>SENSE:SA:COHerance:LO:FTHRu:MONitor:LAST:VALUE</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the target receiver for LO Feedthru monitor.</td>
<td>SENSE:SA:COHerance:LO:FTHRu:MONitor:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the LO feedthru monitoring state.</td>
<td>SENSE:SA:COHerence:LO:FTHRu:MONitor[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the threshold level for warning.</td>
<td>SENSE:SA:COHerence:LO:FTHRu:MONitor:TOLerance</td>
<td>None</td>
</tr>
</tbody>
</table>
Set and read the monitoring method for LO feedthru monitor.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:LO:FTHRu:MONitor:TYPE</td>
<td>None</td>
</tr>
</tbody>
</table>

**Fixture Simulator Commands**

These commands are used to query and set fixture values on a scratch fixture.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:FSIMulator:DRAFt: Commands</td>
<td>None</td>
</tr>
</tbody>
</table>

These commands are used to query ONLY fixture values on the active fixture.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:FSIMulator: Active Commands</td>
<td>None</td>
</tr>
</tbody>
</table>

**Phase Control Commands**

Returns the available internal ports that the external port can be set to.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:PHASe:EXTernal:CATalog?</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and returns the internal port that the external port is routed through.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:PHASe:EXTernal:PORT</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns the available phase control modes for the specified port.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:PHASe:MODE:CATalog?</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and returns the Phase Control mode.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:PHASe:MODE[:VALue]</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns the available parameters.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:PHASe:PARameter[:CATalog?]</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and returns the ratioed
receivers (parameter) to use for phase control.

| Returns the available ports that can be used as phase control reference ports for the phase controlled port. | SOURce:PHASe:REFerence:CATalog? | None |
| Sets and returns the reference port for the Phase Control measurement. | SOURce:PHASe:REFerence:PORT | None |

**Misc. Commands**

| Sets and returns the IF bandwidth filter shape | SENSE:IF:BANDwidth:FILTer | None |

The following are new programming commands for VNA release A.14.60.xx See What's New.

Dark shaded commands existed before this release. They are provided here for convenience.

**Phase Reference Commands**

<p>| Reads the number of phase reference modules connected. | CONTrol:PHASE:COUNT? | None |
| Reads the calibration data from the phase reference of the specified module (x), and writes the data to the Path | CONTrol:PHASE:MODule[x]:DATA:STORe | None |</p>
<table>
<thead>
<tr>
<th>Command Description</th>
<th>Command Code</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reads the model number of the specified module (x).</td>
<td><strong>CONTrol:PHASe:MODule[x]:MODel?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Reads the serial number of the specified module (x).</td>
<td><strong>CONTrol:PHASe:MODule[x]:SERial?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Turns on the phase reference, and sets the divide number.</td>
<td><strong>CONTrol:PHASe:MODule[x]:SETup</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>USB Noise Source Commands</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets and reads the noise source ON or OFF.</td>
<td><strong>OUTPut:MANual:NOISe[:STATe]?</strong></td>
<td>NoiseSourceState</td>
</tr>
<tr>
<td>Specifies whether to use the ENR file stored internally on the USB sensor or to use specified .enr file.</td>
<td><strong>SENSe:NOISe:ENR</strong></td>
<td>None</td>
</tr>
<tr>
<td>Returns a comma separated list of connected USB Noise Sources.</td>
<td><strong>SENSe:NOISe:USBSourCe:CATalog?</strong></td>
<td>None</td>
</tr>
<tr>
<td>This writes the ENR file stored in USB Noise Sensor to the filename.</td>
<td><strong>SENSe:NOISe:USBSourCe:ENR:SAVE</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and queries the USB noise source to use for calibration.</td>
<td><strong>SENSe:NOISe:USBSourCe[:SELect]</strong></td>
<td>None</td>
</tr>
<tr>
<td>Returns sensor reported temperature in Kelvin.</td>
<td><strong>SENSe:NOISe:USBSourCe:TEMPerature?</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Misc. Commands</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SCPI</th>
<th>BSSMeasurement</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:FSIMulator:BALun:PARameter:BSSended[:DEFine]</td>
<td>BSSMeasurement</td>
</tr>
</tbody>
</table>

This command enables/disables asynchronous sweeping during a calibration acquisition.

<table>
<thead>
<tr>
<th>SCPI</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:CORRection:COLLect:IDLE:TRIGger</td>
<td>None</td>
</tr>
</tbody>
</table>

Limits start/stop times to avoid aliasing.

<table>
<thead>
<tr>
<th>SCPI</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:MEASure:TRANsform:TIME:CLIP</td>
<td>None</td>
</tr>
</tbody>
</table>

The following are new programming commands for VNA release A.14.40.xx See What's New.

Dark shaded commands existed before this release. They are provided here for convenience.

<table>
<thead>
<tr>
<th>Phase Noise Application Commands</th>
<th>SCPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the calculated Allan deviation.</td>
<td>CALCulate:MEASure:PN:AVARiance:DEViation</td>
</tr>
<tr>
<td>Returns the calculated Jitter.</td>
<td>CALCulate:MEASure:PN:AVARiance:JITTer</td>
</tr>
<tr>
<td>Returns the calculated Allan variance.</td>
<td>CALCulate:MEASure:PN:AVARiance:VARiance</td>
</tr>
<tr>
<td>Returns the measured carrier frequency.</td>
<td>CALCulate:MEASure:PN:CARRier:FREQuency?</td>
</tr>
<tr>
<td>Returns the measured carrier power level.</td>
<td>CALCulate:MEASure:PN:CARRier:LEVel?</td>
</tr>
<tr>
<td>Sets and returns the dBc data array for the selected measurement.</td>
<td>CALCulate:MEASure:PN:DATA:PDATa</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Sets and returns the dBc memory data for the selected measurement.</td>
<td><code>CALCulate:MEASure:PN:DATA:PMEMory</code></td>
</tr>
<tr>
<td>Returns the spurious data (0 or 1) for the selected measurement.</td>
<td><code>CALCulate:MEASure:PN:DATA:SPData</code></td>
</tr>
<tr>
<td>Returns the spurious memory data (0 or 1) for the selected measurement.</td>
<td><code>CALCulate:MEASure:PN:DATA:SPMemory</code></td>
</tr>
<tr>
<td>Returns the specified data for the selected range number.</td>
<td><code>CALCulate:MEASure:PN:INTegral:RANGe:DATA</code></td>
</tr>
<tr>
<td>Sets and returns the start frequency of the selected integration range.</td>
<td><code>CALCulate:MEASure:PN:INTegral:RANGe:STARt</code></td>
</tr>
<tr>
<td>Sets and returns the stop frequency of the selected integration range.</td>
<td><code>CALCulate:MEASure:PN:INTegral:RANGe:STOP</code></td>
</tr>
<tr>
<td>Sets and returns the integration range type of the selected integration range.</td>
<td><code>CALCulate:MEASure:PN:INTegral:RANGe:TYPE</code></td>
</tr>
<tr>
<td>Sets and returns the weighting filter of the selected integration range.</td>
<td><code>CALCulate:MEASure:PN:INTegral:RANGe:WERightung</code></td>
</tr>
<tr>
<td>Enables and disables the spot noise calculation on every decade offset frequency.</td>
<td><code>CALCulate:MEASure:PN:SNOise:DECades[:STATe]</code></td>
</tr>
<tr>
<td>Returns the spot noise x-axis array of all decade offset frequencies.</td>
<td><code>CALCulate:MEASure:PN:SNOise:DECades:X</code></td>
</tr>
<tr>
<td>Returns the spot noise y-axis array of all decade offset frequencies.</td>
<td><code>CALCulate:MEASure:PN:SNOise:DECades:Y</code></td>
</tr>
<tr>
<td>Enables and disables spot noise calculation for the specified user-defined offset frequency.</td>
<td>CALCulate:MEASure:PN:SNOise:USER[:STATe]</td>
</tr>
<tr>
<td>Sets and returns the offset frequency on which the spot noise is calculated.</td>
<td>CALCulate:MEASure:PN:SNOise:USER:X</td>
</tr>
<tr>
<td>Returns the spot noise y-axis value of the specified offset frequency.</td>
<td>CALCulate:MEASure:PN:SNOise:USER:Y</td>
</tr>
<tr>
<td>Enables and disables spurious analysis.</td>
<td>CALCulate:MEASure:PN:SPURious:ANALysis[:STATe]</td>
</tr>
<tr>
<td>Returns a list of detected spurs.</td>
<td>CALCulate:MEASure:PN:SPURious:DATA</td>
</tr>
<tr>
<td>Enables and disables spur omission.</td>
<td>CALCulate:MEASure:PN:SPURious:OMISsion[:STATe]</td>
</tr>
<tr>
<td>Sets and returns the User Spur Table data which defines spurs to omit.</td>
<td>CALCulate:MEASure:PN:SPURious:OSSPur:DATA</td>
</tr>
<tr>
<td>Deletes the User Spur Table data which defines spurs to omit.</td>
<td>CALCulate:MEASure:PN:SPURious:OSSPur:DELete</td>
</tr>
<tr>
<td>Enables and disables user specified spur omission.</td>
<td>CALCulate:MEASure:PN:SPURious:OSSPur[:STATe]</td>
</tr>
<tr>
<td>Sets and returns the spurious sensibility number.</td>
<td>CALCulate:MEASure:PN:SPURious:SENSibility</td>
</tr>
<tr>
<td>Sets and returns the spurious table sorting order.</td>
<td>CALCulate:MEASure:PN:SPURious:SORT</td>
</tr>
<tr>
<td>Sets and returns the minimum spurious threshold level.</td>
<td>CALCulate:MEASure:PN:SPURious:THReshold:LEVel:MINimum</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Sets and returns the spurious threshold table data.</td>
<td><code>CALCulate:MEASure:PN:SPURious:THReshold:TABle:DATA</code></td>
</tr>
<tr>
<td>Deletes the spurious threshold table.</td>
<td><code>CALCulate:MEASure:PN:SPURious:THReshold:TABle:DELeTe</code></td>
</tr>
<tr>
<td>Enable or disable displaying the integrated noise table.</td>
<td><code>DISPlay:WINDow:TABLE:INOise:ENABLE</code></td>
</tr>
<tr>
<td>Enable or disable displaying the spot noise table.</td>
<td><code>DISPlay:WINDow:TABLE:SNOise:ENABLE</code></td>
</tr>
<tr>
<td>Enable or disable displaying the spurious table.</td>
<td><code>DISPlay:WINDow:TABLE:SPURious:ENABLe</code></td>
</tr>
<tr>
<td>Load user specified spurious frequency list of the active trace of the active channel.</td>
<td><code>MMEMory:LOAD:PN:SPURious:OSSPur</code></td>
</tr>
<tr>
<td>Load the threshold table of the active trace of the active channel.</td>
<td><code>MMEMory:LOAD:PN:SPURious:THReshold</code></td>
</tr>
<tr>
<td>Enables and disables check for carrier.</td>
<td><code>SENSe:PN:ADJust:CONfigure:FREQuency:CHECk</code></td>
</tr>
<tr>
<td>Sets and returns the high frequency limit to use during a broadband carrier search.</td>
<td><code>SENSe:PN:ADJust:CONfigure:FREQuency:LIMit:HIGH</code></td>
</tr>
<tr>
<td>Sets and returns the low frequency limit to use during a broadband carrier search.</td>
<td><code>SENSe:PN:ADJust:CONfigure:FREQuency:LIMit:LOW</code></td>
</tr>
<tr>
<td>Enables and disables a broadband carrier search.</td>
<td><code>SENSe:PN:ADJust:CONfigure:FREQuency:SEARch[:STATe]</code></td>
</tr>
<tr>
<td>Sets and returns the threshold to use during a carrier search.</td>
<td><code>SENSe:PN:ADJust:CONfigure:LEVel:THReshold</code></td>
</tr>
<tr>
<td>Sets and returns the FFT average factor number.</td>
<td><code>SENSe:PN:FAVerage:FACTor</code></td>
</tr>
</tbody>
</table>
Sets and returns the resolution bandwidth ratio.  
**SENSe:PN:BWIDth[:RESolution]:RATio**

Sets and returns the receiver for the phase noise measurement.  
**SENSe:PN:RECeiver**

Sets and returns the carrier frequency.  
**SENSe:PN:SWEep:CARRier:FREQuency**

Sets and returns the sweep noise mode.  
**SENSe:PN:SWEep:NOISe:MODE**

### Marker Search Commands

#### Spurious Search

CALCulate:MEASure:MARKer:FUNCtion:EXECute SPURious

#### Spurious Right >> Search

CALCulate:MEASure:MARKer:FUNCtion:EXECute RSPurious

#### << Spurious Left Search

CALCulate:MEASure:MARKer:FUNCtion:EXECute LSPurious

#### Multi Spurious Search

CALCulate:MEASure:MARKer:FUNCtion:MULTi:EXECute SPURious

### Global Source Commands

Set and return the frequency of the specified global source.  
**SYSTem:PREFerences:SOURce:GLOBal:FREQuency**

Set and return the output state of the specified global source.  
**SYSTem:PREFerences:SOURce:GLOBal:OUTPut[:STATe]**

Set and return the global sources that ignore the power off setting.  
**SYSTem:PREFerences:SOURce:GLOBal:POFF:IGNore[:STATe]**

Set and return the power of the specified global source.  
**SYSTem:PREFerences:SOURce:GLOBal:POWer**
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:PREFerences:SOURce:GLOBal[:STATe]</td>
<td>Set and return the global state of the specified global source.</td>
<td></td>
</tr>
<tr>
<td><strong>Modulation Distortion Commands</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets the X-axis fixed parameter for a fixed sweep type measurement in a Modulation Distortion channel.</td>
<td>CALCulate:MEASure:X:AXIS:FIXed:PARameter</td>
<td></td>
</tr>
<tr>
<td>Sets the X-axis fixed parameter value for a fixed sweep type measurement in a Modulation Distortion channel.</td>
<td>CALCulate:MEASure:X:AXIS:FIXed:PARameter:VALue</td>
<td></td>
</tr>
<tr>
<td>Enables or disables the external modulated source to pass through the attenuator of the specified PNA source.</td>
<td>SENSE:DISTortion:PATH:SOURce:ATTenuation:INCLude</td>
<td></td>
</tr>
<tr>
<td>Selects between a ramp type power sweep or a list of power values to define a power sweep.</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LEVel:TYPE</td>
<td></td>
</tr>
<tr>
<td>Adds a new row to the power sweep list table.</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LIST:ADD</td>
<td></td>
</tr>
<tr>
<td>Deletes the specified row from the</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LIST:DELete</td>
<td></td>
</tr>
</tbody>
</table>
Sets and returns the power level used for the specified row in the power sweep list table.

**SENSe:DISTortion:SWEep:POWer:CARRier:LIST:LEVel**

Specifies the file path to recall a previous power sweep list file.

**SENSe:DISTortion:SWEep:POWer:CARRier:LIST:LOAD**

Sets and returns the noise bandwidth setting used for the specified row in the power sweep list table.

**SENSe:DISTortion:SWEep:POWer:CARRier:LIST:NBW**

Selects the noise bandwidth mode in the power sweep list table.

**SENSe:DISTortion:SWEep:POWer:CARRier:LIST:NBW:MODE**

Sets and returns the number of power points to measure for the power sweep list measurement.

**SENSe:DISTortion:SWEep:POWer:CARRier:LIST:POINts**

5592
<table>
<thead>
<tr>
<th>Function Description</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and returns the receiver attenuation for the specified row in the power sweep list table.</td>
<td>SET::DISTortion::SWEep::POWer::CARRier::LIST::RECeiver::ATTenuation</td>
</tr>
<tr>
<td>Selects the receiver attenuation mode in the power sweep list table.</td>
<td>SET::DISTortion::SWEep::POWer::CARRier::LIST::RECeiver::ATTenuation::MODE</td>
</tr>
<tr>
<td>Specifies the file path to save a power sweep list file.</td>
<td>SET::DISTortion::SWEep::POWer::CARRier::LIST::SAVE</td>
</tr>
<tr>
<td>Sets and returns the source attenuation for the specified row in the power sweep list table.</td>
<td>SET::DISTortion::SWEep::POWer::CARRier::LIST::SOURce::ATTenuation</td>
</tr>
<tr>
<td>Selects the source attenuation mode in the power sweep list table.</td>
<td>SET::DISTortion::SWEep::POWer::CARRier::LIST::SOURce::ATTenuation::MODE</td>
</tr>
<tr>
<td>Sets and returns the starting power level for the power sweep ramp measurement.</td>
<td>SET::DISTortion::SWEep::POWer::CARRier::RAMP::LEVel::STARt</td>
</tr>
<tr>
<td>Sets and</td>
<td>SET::DISTortion::SWEep::POWer::CARRier::RAMP::LEVel::STOP</td>
</tr>
</tbody>
</table>
returns the stop power level for the power sweep ramp measurement.

Enable or disable increasing noise bandwidth at high powers automatically for faster measurements.

Sets and returns the number of power points to measure for the power sweep ramp measurement.

Selects between a fixed or power sweep type of sweep.

Specifies the file path to save a modulation distortion table file.

Sort distortion table by band or power.

Enable or disable the LO feedthru

<table>
<thead>
<tr>
<th>Specification</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable or disable increasing noise bandwidth at high powers automatically for faster measurements.</td>
<td>SENSE:DISTortion:SWEep:POWER:CARRier:RAMP:NBW:AUTO</td>
</tr>
<tr>
<td>Sets and returns the number of power points to measure for the power sweep ramp measurement.</td>
<td>SENSE:DISTortion:SWEep:POWER:CARRier:RAMP:POINts</td>
</tr>
<tr>
<td>Selects between a fixed or power sweep type of sweep.</td>
<td>SENSE:DISTortion:SWEep:TYPE</td>
</tr>
<tr>
<td>Specifies the file path to save a modulation distortion table file.</td>
<td>SENSE:DISTortion:TABLE:DISPlay:SAVE</td>
</tr>
<tr>
<td>Sort distortion table by band or power.</td>
<td>SENSE:DISTortion:TABLE:DISPlay:SORT</td>
</tr>
<tr>
<td>Enable or disable the LO feedthru</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FTHRu:ENABle</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations used by the calibration routine.</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FTHRu:ITERations</td>
</tr>
<tr>
<td>Sets and reads the receiver for LO feedthru modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FTHRu:RECeiver</td>
</tr>
<tr>
<td>Sets and reads the calibration span for a LO feedthru modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FTHRu:SPAN</td>
</tr>
<tr>
<td>Sets and reads the desired LO feedthru modulation calibration tolerance.</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FTHRu:TOLerance</td>
</tr>
</tbody>
</table>

**SA Commands**

Enables and disables baseband sweep independent of the LO sweep to allow signals down to 1 Hz to be analyzed.

Set and read enable force LO to the specified frequency.

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SENSSe:SA:LO:BASEband[:STATe]</td>
</tr>
<tr>
<td></td>
<td>SENSSe:SA:LO:FORCe:FREQuency</td>
</tr>
</tbody>
</table>
Set and read the divider in the equation for setting LO frequency.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SESe:SA:LO:FORCE:OFFSet:DIVider</td>
<td>Sets and reads the divider in the equation for setting LO frequency.</td>
</tr>
</tbody>
</table>

Set and read the multiplier in the equation for setting LO frequency.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SESe:SA:LO:FORCE:OFFSet:MULtiplier</td>
<td>Sets and reads the multiplier in the equation for setting LO frequency.</td>
</tr>
</tbody>
</table>

Set and read the source name in the equation for setting LO frequency.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SESe:SA:LO:FORCE:OFFSet:SOURce</td>
<td>Sets and reads the source name in the equation for setting LO frequency.</td>
</tr>
</tbody>
</table>

Set and read enable force LO to frequency mode.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SESe:SA:LO:FORCE[:STATe]</td>
<td>Sets and reads the enable force LO to frequency mode.</td>
</tr>
</tbody>
</table>

**Misc. Commands**

Sets and reads the sample rate of the compact modulation file.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

Enables or disables a brick-wall filter for spectral leakage.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

Sets the local/remote state to local.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:LOCal</td>
<td>Sets the local/remote state to local.</td>
</tr>
</tbody>
</table>

Enables/disables changing from local to remote status when a SCPI command is received.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:PREFerences:ITEM:REMote:AUTO[:STATe]</td>
<td>Enables/disables changing from local to remote status when a SCPI command is received.</td>
</tr>
</tbody>
</table>

Sets the raw data for the specified raw parameter buffer.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:MEASure:DATA:RAW</td>
<td>Sets the raw data for the specified raw parameter buffer.</td>
</tr>
</tbody>
</table>

Returns the list of raw parameters associated with the measurement <mnum>.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:MEASure:DATA:RAW:CATalog?</td>
<td>Returns the list of raw parameters associated with the measurement &lt;mnum&gt;.</td>
</tr>
</tbody>
</table>
Enables/disables External Reference settings being affected by Recall/Preset.

<table>
<thead>
<tr>
<th>SYSTem:PREFerences:ITEM:ROSCillator:RECall</th>
</tr>
</thead>
</table>

Sets and reads the True Mode state for a specified balanced port.

<table>
<thead>
<tr>
<th>CALCulate:FSIMulator:BALun:BPORt:STIMulus:TRUe:STATe</th>
</tr>
</thead>
</table>

The following are new programming commands for VNA release A.14.20.xx See What's New.

Dark shaded commands existed before this release. They are provided here for convenience.

<table>
<thead>
<tr>
<th>Receiver Leveling - Point Mode Commands</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets the phase offset between the two balanced stimulus ports.</td>
<td>CALCulate:FSIMulator:BALun:LPORt:OFFSet:PHASE</td>
<td>None</td>
</tr>
<tr>
<td>Sets the power offset between the two balanced stimulus ports.</td>
<td>CALCulate:FSIMulator:BALun:LPORt:OFFSet:POWer</td>
<td>None</td>
</tr>
<tr>
<td>Sets the start value for a phase sweep.</td>
<td>CALCulate:FSIMulator:BALun:LPORt:SWEep:PHASE:STARt</td>
<td>None</td>
</tr>
<tr>
<td>Sets the stop value for a phase sweep.</td>
<td>CALCulate:FSIMulator:BALun:LPORt:SWEep:PHASE:STOP</td>
<td>None</td>
</tr>
<tr>
<td>Specifies which balanced port the phase will be swept.</td>
<td>CALCulate:FSIMulator:BALun:PHASE:SWEep:PORT</td>
<td>None</td>
</tr>
<tr>
<td>Sets all ports to pre-sweep or point leveling mode for the specified channel.</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:ACQuisition:MODE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:ITERation:VALue</td>
<td>IterationNumber</td>
</tr>
</tbody>
</table>
returns the maximum iterations to be used in order to achieve the tolerance setting.

<table>
<thead>
<tr>
<th>Enables or disables the receiver leveling maximum iteration search function.</th>
<th>SOURce:POWer:ALC[:MODE]:RECeiver:ITERation:ENABle</th>
<th>None</th>
</tr>
</thead>
</table>

Sets the specific PULSe4 behavior.

<table>
<thead>
<tr>
<th>Sets the specific PULSe4 behavior.</th>
<th>SENSE:PULSe4:MODE</th>
<th>None</th>
</tr>
</thead>
</table>

**Misc. Commands**

<table>
<thead>
<tr>
<th>Enables/disables selecting between the internal reference or external reference.</th>
<th>SENSE:ROSCillator:CONTrol:AUTO</th>
<th>None</th>
</tr>
</thead>
</table>

Returns the Synthesizer Revision number.

<table>
<thead>
<tr>
<th>Returns the Synthesizer Revision number.</th>
<th>SYSTem:CONFigure:REVision:PNA:SYNthesizer:VERSion?</th>
<th>None</th>
</tr>
</thead>
</table>


<table>
<thead>
<tr>
<th>Sets and returns the measurement for the Balanced - Single-Ended - Single-Ended topology.</th>
<th>CALCulate:MEASure:PARameter</th>
<th>BSSMeasurement</th>
</tr>
</thead>
</table>

Returns the VNA port number that is connected to the Negative side of the DUT's.

| Returns the VNA port number that is connected to the Negative side of the DUT's | CALCulate<cnum>:FSIMulator:BALun:TOPology:BSSended[:PPORts]? BSS_BalPortNeg | None |
| balanced port. Returns the VNA port number that is connected to the Positive side of the DUT's balanced port. | CALCulate\(<\text{cnum}>\)\,:FSIMulator:BALun:TOPology:BSSended[\,:PPORts]? BSS_BalPortPositive |
| Returns the VNA port number that is connected to the Single-ended port 1. | CALCulate\(<\text{cnum}>\)\,:FSIMulator:BALun:TOPology:BSSended[\,:PPORts]? BSS_SEPort_1 |
| Returns the VNA port number that is connected to the Single-ended port 2. | CALCulate\(<\text{cnum}>\)\,:FSIMulator:BALun:TOPology:BSSended[\,:PPORts]? BSS_SEPort_2 |
| For a Balanced-Single-ended - Single-ended device type, maps the VNA ports to the DUT ports. | CALCulate\(<\text{cnum}>\)\,:FSIMulator:BALun:TOPology:BSSended[\,:PPORts] SetBSSPorts |

The following are new programming commands for VNA release A.13.95.xx See What's New.
## Modulation Distortion/Modulation Distortion Converters Commands

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sets the measurement filter to either None (default) or RRC (root-raised-cosine filter) for EVM and ACP measurements only.</strong></td>
<td>SENSE:DISTortion:MEASure:FILTERer</td>
</tr>
<tr>
<td><strong>Sets and returns the Alpha factor of the measurement filter.</strong></td>
<td>SENSE:DISTortion:MEASure:FILTERer:ALPHA</td>
</tr>
<tr>
<td><strong>Sets and returns the Symbol Rate of the filter.</strong></td>
<td>SENSE:DISTortion:MEASure:FILTERer:SRATe</td>
</tr>
<tr>
<td><strong>Enables or disables using the symbol rate from the file.</strong></td>
<td>SENSE:DISTortion:MEASure:FILTERer:SRATe:AUTO[:STATE]</td>
</tr>
<tr>
<td><strong>Resets the LO Frequency Delta and Tuning parameters to their default settings.</strong></td>
<td>SENSE:MIxer:ELO:RESet</td>
</tr>
<tr>
<td><strong>Sets and returns the Noise Bandwidth for Broadband and Precise tuning sweeps.</strong></td>
<td>SENSE:MIxer:ELO:TUNing:NBW</td>
</tr>
<tr>
<td><strong>Enable or disable the distortion calibration state.</strong></td>
<td>SOURce:MODulation:CORRection:COLLection:DISTortion:ENABle</td>
</tr>
<tr>
<td><strong>Sets and reads the maximum number of iterations used by the calibration routine.</strong></td>
<td>SOURce:MODulation:CORRection:COLLection:DISTortion:ITERations</td>
</tr>
<tr>
<td><strong>Sets and reads the receiver for distortion calibration.</strong></td>
<td>SOURce:MODulation:CORRection:COLLection:DISTortion:RECeiver</td>
</tr>
<tr>
<td><strong>Sets and reads the calibration span for a distortion calibration.</strong></td>
<td>SOURce:MODulation:CORRection:COLLection:DISTortion:SPAN</td>
</tr>
<tr>
<td><strong>Sets and reads the desired distortion calibration tolerance.</strong></td>
<td>SOURce:MODulation:CORRection:COLLection:DISTortion:TOLerance</td>
</tr>
<tr>
<td><strong>Enable or disable the equalization calibration state.</strong></td>
<td>SOURce:MODulation:CORRection:COLLection:EQUalization:ENABle</td>
</tr>
<tr>
<td><strong>Sets and reads the maximum number of iterations used by the calibration routine.</strong></td>
<td>SOURce:MODulation:CORRection:COLLection:EQUalization :ITERations</td>
</tr>
<tr>
<td><strong>Sets and reads the receiver for equalization calibration.</strong></td>
<td>SOURce:MODulation:CORRection:COLLection:EQUalization :RECeiver</td>
</tr>
<tr>
<td><strong>Sets and reads the calibration span for a equalization calibration.</strong></td>
<td>SOURce:MODulation:CORRection:COLLection:EQUalization :SPAN</td>
</tr>
<tr>
<td><strong>Sets and reads the desired equalization calibration tolerance.</strong></td>
<td>SOURce:MODulation:CORRection:COLLection:EQUalization :TOLerance</td>
</tr>
<tr>
<td>Command Description</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>Allows setup of multiple carriers when defining a multicarrier signal.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:SUBCarrier:NUMBer</td>
</tr>
<tr>
<td>Sets and reads the offset of the selected subcarrier.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:SUBCarrier:OFFSET</td>
</tr>
<tr>
<td>Sets and reads the span of the selected subcarrier.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:SUBCarrier:SPAN</td>
</tr>
<tr>
<td>Enables or disables receiver timing auto generation from the source pulse timing.</td>
<td>SENSe:DIStortion:PULSe:RECeiver:AUTO</td>
</tr>
<tr>
<td><strong>AM Distortion Commands</strong></td>
<td></td>
</tr>
<tr>
<td>Sets and returns the compression level.</td>
<td>CALCulate:MEASure:DIStortion:BACKoff:COMPression</td>
</tr>
<tr>
<td>Enable/disable compression calculation.</td>
<td>CALCulate:MEASure:DIStortion:BACKoff[:STATe]</td>
</tr>
<tr>
<td>Displays phase or amplitude distortion.</td>
<td>CALCulate:MEASure:DIStortion:MODE</td>
</tr>
<tr>
<td>Sets the aperture value over which the phase or gain slope will be calculated.</td>
<td>CALCulate:MEASure:DIStortion:SLOPe:APERture</td>
</tr>
<tr>
<td>Enables/disables phase slope (AMP) or gain slope (AMAM) over the slope aperture to be displayed.</td>
<td>CALCulate:MEASure:DIStortion:SLOPe[:STATe]</td>
</tr>
<tr>
<td><strong>Trace Deviation Commands</strong></td>
<td></td>
</tr>
<tr>
<td>Calculates the deviation from a least-squares best fit line.</td>
<td>CALCulate:MEASure:COMPutation:DEViation</td>
</tr>
<tr>
<td><strong>Noise Tuner Commands</strong></td>
<td></td>
</tr>
<tr>
<td>Set and read a custom noise tuner file to be used instead of the one generated automatically based on the state.</td>
<td>SENSe:NOISe:TUNer:FILE:NAME</td>
</tr>
<tr>
<td>Sets the state of the custom noise tuner file.</td>
<td>SENSe:NOISe:TUNer:FILE[:STATe]</td>
</tr>
<tr>
<td><strong>SA Commands</strong></td>
<td></td>
</tr>
<tr>
<td>Read the count of ADCs/receivers in overload status for the previous sweep.</td>
<td>SENSe:SA:ADC:OVERload:COUNI?</td>
</tr>
<tr>
<td>Task Description</td>
<td>Command</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Read the list of measured receivers in overload status for the previous sweep.</td>
<td>SENSE:SA:ADC:OVERload:LIST?</td>
</tr>
<tr>
<td>Set and read the ADC overload threshold percentage.</td>
<td>SENSE:SA:ADC:OVERload:PERCent</td>
</tr>
<tr>
<td>Read maximum ADC range of all receivers in use.</td>
<td>SENSE:SA:ADC:RANGe:PERCent[:MAXimum]?</td>
</tr>
<tr>
<td>Read minimum ADC range of all receivers in use.</td>
<td>SENSE:SA:ADC:RANGe:PERCent:MINimum?</td>
</tr>
<tr>
<td>Read ADC range for specified receiver.</td>
<td>SENSE:SA:ADC:RANGe:PERCent:RECeiver?</td>
</tr>
<tr>
<td>Read the current frequencies count.</td>
<td>SENSE:SA:DATA:KEEP:FREQuencies:COUNt?</td>
</tr>
<tr>
<td>Read the number of exported receivers.</td>
<td>SENSE:SA:DATA:KEEP:RECeivers:COUNt?</td>
</tr>
<tr>
<td>Read the number of currently exported receivers.</td>
<td>SENSE:SA:DATA:KEEP:RECeivers[:CURRent]</td>
</tr>
<tr>
<td>Set and read the list of receivers to export.</td>
<td>SENSE:SA:DATA:KEEP:RECeivers:LIST</td>
</tr>
<tr>
<td>Read the scalar data.</td>
<td>SENSE:SA:DATA:KEEP:SCALar:GET?</td>
</tr>
<tr>
<td>Set and read the ON/OFF state of the end of sweep processing.</td>
<td>SENSE:SA:DATA:KEEP[:STATe]</td>
</tr>
<tr>
<td>Read the vector data.</td>
<td>SENSE:SA:DATA:KEEP:VECTor:GET?</td>
</tr>
<tr>
<td>Set and read the maximum pulse search count for the next pulse search execution.</td>
<td>SENSE:SA:COHerence:PULSe:SEARch:COUNt</td>
</tr>
<tr>
<td>Set and read the duty cycle tolerance.</td>
<td>SENSE:SA:COHerence:PULSe:SEARch:DUTY:TOLerance</td>
</tr>
<tr>
<td>Executes the pulse search.</td>
<td>SENSE:SA:COHerence:PULSe:SEARch:EXECute</td>
</tr>
</tbody>
</table>
Sets all the search parameters to their default values.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:INITialize</td>
<td>Sets all the search parameters to their default values.</td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Returns the number of possible pulse configurations.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Returns the pulse period for found pulse configuration number.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:ITEM:PERiod?</td>
<td>Returns the pulse period for found pulse configuration number.</td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Selects the found configuration number, and applies it to the measurements.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:ITEM:SELect</td>
<td>Selects the found configuration number, and applies it to the measurements.</td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Returns the pulse width for found pulse configuration number.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:ITEM:WIDTh?</td>
<td>Returns the pulse width for found pulse configuration number.</td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Set and read the period tolerance.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:PERiod:TOLerance</td>
<td>Set and read the period tolerance.</td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Set and read the period starting point for the next pulse search execution.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:PERiod[:VALue]</td>
<td>Set and read the period starting point for the next pulse search execution.</td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Set and read the priority choice for the search algorithm.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:PRIority</td>
<td>Set and read the priority choice for the search algorithm.</td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Set and read the width tolerance.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:WIDTh:TOLerance</td>
<td>Set and read the width tolerance.</td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Set and read the width starting point for the next pulse search execution.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:PULSe:SEARch:WIDTh[:VALue]</td>
<td>Set and read the width starting point for the next pulse search execution.</td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Preferences Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:PREFerences:ITEM:CORRection:PARallel:PROCess</td>
<td>Enable or disable parallel processing in the CPU which provides higher calculation speeds.</td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Sets the Series-C and Shunt-L components to legacy or theoretical behavior when set to zero.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:PREFerences:ITEM:FIXTure:CIRCuit:DEFAults</td>
<td>Sets the Series-C and Shunt-L components to legacy or theoretical behavior when set to zero.</td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following are new programming commands for VNA release A.13.80.xx See What's New.

Dark shaded commands existed before this release. They are provided here for convenience.
<table>
<thead>
<tr>
<th><strong>SA Modulation Settings Commands</strong></th>
<th><strong>SCPI</strong></th>
<th><strong>COM</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and reads the guard band between the end of the modulation signal and where ACPR is measured.</td>
<td>SOURc:MODulation:AUTO:ACPR:GBANd</td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable ACPR markers.</td>
<td>SOURc:MODulation:AUTO:ACPR[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Adjusts frequencies and markers if modulation settings are changed.</td>
<td>SOURc:MODulation:AUTO:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the guard band on each side of the notch in an NPR measurement.</td>
<td>SOURc:MODulation:AUTO:NPR:GBANd</td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable NPR markers.</td>
<td>SOURc:MODulation:AUTO:NPR[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable automatic updating of SA sweep settings and coherence settings if the modulation settings are changed.</td>
<td>SOURc:MODulation:AUTO:SA[:STATe]</td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Source Modulation Calibration Commands</strong></th>
<th><strong>SCPI</strong></th>
<th><strong>COM</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and reads the ACP lower frequency delta from the edge of the carrier to the beginning of the Cal Span.</td>
<td>SOURc:MODulation:CORRection:COLLection:ACP:LOWer:GBANd</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the ACP upper frequency delta from the edge of the carrier to the beginning of the calibration span.</td>
<td>SOURc:MODulation:CORRection:COLLection:ACP:UPPer:GBANd</td>
<td>None</td>
</tr>
<tr>
<td>Returns a list of the calibrations stored in the .mdx file.</td>
<td>SOURc:MODulation:FILE:CORRection:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Deletes any of the calibration files stored in the .mdx file.</td>
<td>SOURc:MODulation:FILE:CORRection:DELete?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the frequency of the specified source modulation calibration file.</td>
<td>SOURc:MODulation:FILE:CORRection:FREQuency?</td>
<td>None</td>
</tr>
</tbody>
</table>

5604
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the power level of the specified source modulation calibration file.</td>
<td>SOURce:MODulation:FILE:CORRection:POWer?</td>
<td>None</td>
</tr>
<tr>
<td>Append source modulation calibration files stored in the .mdx file.</td>
<td>SOURce:MODulation:CORRection:COLLection:APPend</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads a fixed frequency.</td>
<td>SOURce:MODulation:CORRection:COLLection:FREQuency[:FIXed]</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the number of frequency measurement points.</td>
<td>SOURce:MODulation:CORRection:COLLection:FREQuency:POINts</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the start frequency to use for a swept frequency source modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:FREQuency:STARt</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the stop frequency to use for a swept frequency calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:FREQuency:STOP</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration frequency type to fixed or swept.</td>
<td>SOURce:MODulation:CORRection:COLLection:FREQuency:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads a fixed power level.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWER[:FIXed]</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the number of power measurement points.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWER:POINts</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the start power level to use for a swept calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWER:STARt</td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Units</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Sets and reads the stop power level to use for a swept calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:STOP</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration power type to fixed or swept.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:TYPE</td>
<td>None</td>
</tr>
<tr>
<td><strong>Create/Edit Modulation File</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns the peak-to-average value of the original signal.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:PAVG?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the peak-to-average value of the signal created from the original signal.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:PAVG:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the calculated start time of the signal created from the original signal.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:TIME:STARt:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the calculated peak-to-average value.</td>
<td>SOURce:MODulation:FILE:SIGNal:PAVG:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the calculated signal span.</td>
<td>SOURce:MODulation:FILE:SIGNal:SPAN:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the calculated sampling rate of the signal created from the original signal.</td>
<td>SOURce:MODulation:FILE:SIGNal:SRATe:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the calculated number of tones.</td>
<td>SOURce:MODulation:FILE:SIGNal:TONE:NUMBer:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the calculated spacing between the tones.</td>
<td>SOURce:MODulation:FILE:SIGNal:TONE:SPACing:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td><strong>Misc. Commands</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>This command launches the Receiver IF Cal.</td>
<td>SENS&lt;enumber&gt;:DISTortion:CORRection:COLLect:IF:ACQuire</td>
<td>None</td>
</tr>
<tr>
<td>Enables or disables S-parameter sweep.</td>
<td>SENS&lt;enumber&gt;:DISTortion:SWEep:SPARam[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the IF bandwidth for the linear S21 sweep.</td>
<td>SENS&lt;enumber&gt;:DISTortion:SWEep:SPARam:BWIDth</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the step size for the linear S21 sweep.</td>
<td>SENS&lt;enumber&gt;:DISTortion:SWEep:SPARam:STEP</td>
<td>None</td>
</tr>
</tbody>
</table>
Selects between a chirp signal from the external signal generator or an internal CW signal in the VNA for measuring S-parameters.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SENSE:DISTortion:SWEep:SPARam:TYPE</td>
<td>None</td>
</tr>
</tbody>
</table>

Automatically fills in the measurement settings for all bands from the currently active modulation file loaded in the source.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SENSE:DISTortion:MEASure:BAND:AUTofill</td>
<td>None</td>
</tr>
</tbody>
</table>

Enable/disable an SMU channel.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SYSTem:CONFigure:EDEVice:SMU:CHANnel[1-4]:STATe</td>
<td>ChanActive</td>
</tr>
</tbody>
</table>

The following are new programming commands for VNA release A.13.60.xx See What's New.

<table>
<thead>
<tr>
<th>Modulation Distortion Commands</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and returns the noise bandwidth.</td>
<td>SENSE:SA:BANDwidth:NOISe</td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable setting the noise bandwidth to its maximum possible value.</td>
<td>SENSE:SA:BANDwidth:NOISe:AUTO</td>
<td>None</td>
</tr>
<tr>
<td>Selects between a narrow and wide IF filter anti-aliasing path.</td>
<td>SENSE:DISTortion:ADC:FILTer:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Identifies the normalization used for the EVM measurements.</td>
<td>SENSE:DISTortion:EVM:NORMalize</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the frequency span window used for modeling the DUT's gain and distortion.</td>
<td>SENSE:DISTortion:MEASure:CORRelation:APERture</td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable automatic calculation of the frequency span window used for modeling the DUT's gain and distortion.</td>
<td>SENSE:DISTortion:MEASure:CORRelation:APERture:AUTO[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the integration bandwidth (IBW) of the lower ACP measurement.</td>
<td>SENSE:DISTortion:MEASure:BAND:ACP:LOWer:IBW</td>
<td>None</td>
</tr>
<tr>
<td>Operation Description</td>
<td>Command</td>
<td>Arguments</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------------------</td>
<td>----------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Sets and returns the lower ACP measurement offset.</td>
<td>SENSE:DISTortion:MEASure:BAND:ACP:LOWer:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the integration bandwidth (IBW) of the upper ACP measurement.</td>
<td>SENSE:DISTortion:MEASure:BAND:ACP:UPPer:IBW</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the upper ACP measurement offset.</td>
<td>SENSE:DISTortion:MEASure:BAND:ACP:UPPer:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>Adds a new distortion measurement band.</td>
<td>SENSE:DISTortion:MEASure:BAND:ADD</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the carrier (Signal) integration bandwidth (IBW) for the distortion measurement.</td>
<td>SENSE:DISTortion:MEASure:BAND:CARRier:IBW</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the carrier offset (Signal).</td>
<td>SENSE:DISTortion:MEASure:BAND:CARRier:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>Queries the total number of distortion measurement bands.</td>
<td>SENSE:DISTortion:MEASure:BAND:COUNt?</td>
<td>None</td>
</tr>
<tr>
<td>Deletes the specified distortion measurement band.</td>
<td>SENSE:DISTortion:MEASure:BAND:DELeete</td>
<td>None</td>
</tr>
<tr>
<td>Deletes all existing bands and adds a single default band back into the Measurement Band Table.</td>
<td>SENSE:DISTortion:MEASure:BAND:INITialize</td>
<td>None</td>
</tr>
<tr>
<td>Assigns a name to the specified distortion measurement band.</td>
<td>SENSE:DISTortion:MEASure:BAND:NAME</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the notch integration bandwidth (IBW) for an NPR Notch modulation measurement.</td>
<td>SENSE:DISTortion:MEASure:BAND:NOTCh:IBW</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the notch offset for an NPR Notch modulation measurement.</td>
<td>SENSE:DISTortion:MEASure:BAND:NOTCh:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the modulation distortion type.</td>
<td>SENSE:DISTortion:MEASure:BAND:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the distortion modulation source name.</td>
<td>SENSE:DISTortion:MODulate:SOURce</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the VNA port number connected to the DUT input.</td>
<td>SENSE:DISTorton:PATH:DUT:INPut</td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Default</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Sets and returns the nominal DUT gain.</td>
<td>SENSE:DISTortion:PATH:DUT:NOMinal:GAIN</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the nominal DUT noise figure.</td>
<td>SENSE:DISTortion:PATH:DUT:NOMinal:NF</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the VNA port number connected to the DUT output.</td>
<td>SENSE:DISTortion:PATH:DUT:OUTPUT</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the nominal gain of an external source amplifier.</td>
<td>SENSE:DISTortion:SOURCE:NOMinal:AMPLifier</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the carrier center frequency.</td>
<td>SENSE:DISTortion:Sweep:CARRier:FREQuency</td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable the re-use of the previous linear S-parameter measurements if available and skip sweep.</td>
<td>SENSE:DISTortion:Sweep:SPARam:REUSE</td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable noise sweep used for the distortion test.</td>
<td>SENSE:DISTortion:Sweep:NOISe</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the power level used to measure the DUT gain when operating in its linear region.</td>
<td>SENSE:DISTortion:Sweep:POWer:SPARam:LEVEL</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the power level used for the distortion test at either the input or output of the DUT.</td>
<td>SENSE:DISTortion:Sweep:POWer:CARRier:LEVEL</td>
<td>None</td>
</tr>
<tr>
<td>Set and read where to apply power to the DUT (input or output).</td>
<td>SENSE:DISTortion:Sweep:POWer:CARRier:LEVEL:PORT</td>
<td>None</td>
</tr>
<tr>
<td>Keeps RF power ON at the End of Sweep.</td>
<td>SENSE:DISTortion:Sweep:RETRace:POWer</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the delay in seconds before the test signal to allow the RF source to settle.</td>
<td>SENSE:DISTortion:Sweep:DWELI</td>
<td>None</td>
</tr>
</tbody>
</table>

**Calibrate Source Modulation Commands**

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set and read the ACP modulation calibration state.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:ENABle</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations for the lower ACP modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:LOWer:ITERations</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the receiver for the lower ACP modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:LOWer:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration span for a lower ACP modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:LOWer:SPAN</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired ACP calibration tolerance for the lower ACP modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:LOWer:TOLerance</td>
<td>None</td>
</tr>
<tr>
<td>Enables or disables the upper ACP (ACPUp) modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:UPPer:ENABle</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations for the upper ACP modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:UPPer:ITERations</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the receiver for the upper ACP modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:UPPer:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration span for an upper ACP modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:UPPer:SPAN</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired ACP calibration tolerance for the upper ACP modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:UPPer:TOLerance</td>
<td>None</td>
</tr>
</tbody>
</table>

**Create/Edit Modulation File**

<p>| Sets and reads the source sample rate. | SOURce:MODulation:FILE:INITialize | None |
| Specifies the file path to recall a previous modulation file. | SOURce: MODulation:FILE:LOAD | None |
| Sets and reads the source sample rate. | SOURce:MODulation:FILE:SAVE | None |
| Sets and reads the Carrier offset value. | SOURce:MODulation:FILE:SIGNal:CARRier:OFFSet | None |</p>
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and reads the number of compact modulation files to create.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:FILE:NUMBer</td>
<td>None</td>
</tr>
<tr>
<td>Specifies the file path to recall a previous modulation file from which to create a compact signal.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:OFILe</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the Peak-to-Avg priority for Compact signals.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:PAVG:PRIority</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads where to start the compact signal within the original signal.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:TIME:STARt</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the start time priority for Compact signals.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:TIME:STARt:PRIority</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the DAC scale as a percentage of full scale.</td>
<td>SOURce:MODulation:FILE:SIGNal:DAC:SCALing</td>
<td>None</td>
</tr>
<tr>
<td>Sets the NPR notch location type for the selected NPR Notch modulation type.</td>
<td>SOURce:MODulation:FILE:SIGNal:NPR:NOTCh:LOCation</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the number of NPR notches for NPR Notch modulation type.</td>
<td>SOURce:MODulation:FILE:SIGNal:NPR:NOTCh:NUMBer</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the NPR notch offset frequency of the selected notch.</td>
<td>SOURce:MODulation:FILE:SIGNal:NPR:NOTCh:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the span of the selected notch.</td>
<td>SOURce:MODulation:FILE:SIGNal:NPR:NOTCh:SPAN</td>
<td>None</td>
</tr>
<tr>
<td>Enables or disables signal optimization settings.</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:ENABle</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the number of decimal digits limit for calculated frequencies.</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:FREQuency:LIMit:DDIGits</td>
<td>None</td>
</tr>
<tr>
<td>Enables or disables the number of decimal digits limit for calculated frequencies.</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:FREQuency:LIMit:ENABle</td>
<td>None</td>
</tr>
</tbody>
</table>
Sets and reads the frequency tolerance value (in percent).

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

Sets and reads the number of test signal harmonics you want to be protected against.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:HREJect</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and reads the maximum distance between each tone.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

Sets and reads the minimum number of tones.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

Sets and reads the minimum waveform period.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

Enables or disables the rejection of Nyquist frequencies.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

Sets the optimize signal type.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:TYPE</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and reads the phase when Fixed phase is the Phase Type.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:PHASE:FIXed</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and reads the phase seed when Random phase is the Phase Type.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:PHASE:RANDOM:SEED</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets the phase type.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:PHASE:TYPE</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and reads the signal span.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:SPAN</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and reads the signal span priority.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:SPAN:PRIority</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and reads the source sample rate.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:SRATe</td>
<td>None</td>
</tr>
</tbody>
</table>

Set and read the auto sample rate.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:SRATe:AUTO</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and reads the number of tones.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:TONE:NUMBer</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and reads the tone number priority.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:TONE:NUMBer:PRIority</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and reads the tone spacing.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:TONE:SPACing</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets and reads the tone spacing priority.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:SIGNal:TONE:SPACing:PRIority</td>
<td>None</td>
</tr>
</tbody>
</table>

Sets the modulation type.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation:FILE:TYPE</td>
<td>None</td>
</tr>
</tbody>
</table>

Distortion Table Commands
<table>
<thead>
<tr>
<th>Command</th>
<th>Command</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the currently displayed Distortion Table parameters.</td>
<td>SENSE:DISTortion:TABLE:DISPlay:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Deletes the specified parameter from the Distortion Table.</td>
<td>SENSE:DISTortion:TABLE:DISPlay:DELETE</td>
<td>None</td>
</tr>
<tr>
<td>Adds the specified parameter to the Distortion Table.</td>
<td>SENSE:DISTortion:TABLE:DISPlay:FEED</td>
<td>None</td>
</tr>
<tr>
<td>Returns a list of existing band names from the Measurement Band Table.</td>
<td>SENSE:DISTortion:TABLE:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Returns a list of data parameter names corresponding to the currently specified Measurement Type.</td>
<td>SENSE:DISTortion:TABLE:DATA:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Returns a list of data values from the Distortion Table for the specified band and parameter name.</td>
<td>SENSE:DISTortion:TABLE:DATA:VALUE?</td>
<td>None</td>
</tr>
</tbody>
</table>

### Modulation Distortion Marker Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Command</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set and read the SA marker function type.</td>
<td>CALCulate:MEASure:SA:MARKer:BAND:FUNCTION</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the integration bandwidth marker.</td>
<td>CALCulate:MEASure:SA:MARKer:BAND:IBW</td>
<td>None</td>
</tr>
<tr>
<td>Executes the search ACPR density marker.</td>
<td>CALCulate:MEASure:SA:MARKer:SEARch:ACPR</td>
<td>None</td>
</tr>
<tr>
<td>Executes the search NPR density marker.</td>
<td>CALCulate:MEASure:SA:MARKer:SEARch:NPR</td>
<td>None</td>
</tr>
</tbody>
</table>

### Measurement Data

<table>
<thead>
<tr>
<th>Command</th>
<th>Command</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieves trace data (Y data) from the modulation distortion measurement.</td>
<td>CALCulate:MEASure:DATA:BUFFER:Y?</td>
<td>GetDataBuffer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GetDataBufferCompact</td>
</tr>
</tbody>
</table>
Retrieves frequency tone data from the modulation distortion measurement. | CALCulate:MEASure:DATA:BUFFER:X? | GetXDataBuffer
|---------------------------|-----------------|-------------------|

### Pulse Generator Commands

Sets and reads the device being controlled by the pulse generator output. | SENSE:PULSE:MTiming:DEVICE | PulseTimingDevice
|---------------------------|-----------------|-------------------|

### SA Commands

Set and read the phase display minimum level. | SENSE:SA:COHerence:PHASE:DISPLAY:LEVEL | None
|---------------------------|-----------------|-------------------|

|---------------------------|-----------------|-------------------|

### Misc. Commands

Set and read the name of the external DC device. | SENSE:GCSetup:SAFE:DC:PARAMeter | SafeSweepDCParameter
|---------------------------|-----------------|-------------------|

Set and read the maximum limit of the external DC device. | SENSE:GCSetup:SAFE:DC:MLimit | SafeSweepMaximumDCLimit
|---------------------------|-----------------|-------------------|

Set and read the state of the thru adapter de-embedding. | SENSE:CORRection:COLLect:NOISE:THRU:ADAPter:DEEMbed[:STATE] | None
|---------------------------|-----------------|-------------------|

Sets or returns the noise figure bandwidth. | SENSE:SEGment:NFBW | NoiseFigureBW
|---------------------------|-----------------|-------------------|

Turns ON or OFF the noise figure bandwidth setting. | SENSE:SEGment:NFBW:CONTrol | NoiseFigureBWOption
|---------------------------|-----------------|-------------------|

The following are new programming commands for VNA release A.13.50.xx See What's New.

Dark shaded commands existed before this release. They are provided here for convenience.

### Spectrum Analyzer Commands

Sets and reads the bandwidth of the band density marker. | CALCulate:MEASure:SA:MARKer:BDENsity:BW | BandDensityBW
|---------------------------|-----------------|-------------------|

| CALCulate:MEASure:SA:MARKer:BDENsity:BW |

Returns the band density level in dBm/Hz from the band density marker. | CALCulate:MEASure:SA:MARKer:BDENsity:DATA? | BandDensityValue
|---------------------------|-----------------|-------------------|

| CALCulate:MEASure:SA:MARKer:BDENsity:DATA? |
### Segment Sweep Commands

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and read the phase display minimum level.</td>
<td>SENSE:SA:COHerence:PHASE:DISPLAY:LEVel</td>
<td>PhaseDisplayMinLevel</td>
</tr>
<tr>
<td>Sets and reads the bandwidth of the band power density marker.</td>
<td>CALCulate:MEASure:SA:MARKer:BDENsity:POWer:BW</td>
<td>BandDensityPowerBW</td>
</tr>
<tr>
<td>Sets and reads the state of the band power density marker.</td>
<td>CALCulate:MEASure:SA:MARKer:BDENsity:POWer[:STATE]</td>
<td>BandDensityPowerState</td>
</tr>
<tr>
<td>Sets and reads the bandwidth of the band tone density marker.</td>
<td>CALCulate:MEASure:SA:MARKer:BDENsity:TONE:BW</td>
<td>BandDensityToneBW</td>
</tr>
<tr>
<td>Sets and reads the state of the band tone density marker.</td>
<td>CALCulate:MEASure:SA:MARKer:BDENsity:TONE[:STATE]</td>
<td>BandDensityToneState</td>
</tr>
<tr>
<td>Sets and reads the frequency span used by Power Density to normalize the power.</td>
<td>CALCulate:MEASure:SA:MARKer:BDENsity:EQSPan</td>
<td>BandDensityEQSPan</td>
</tr>
<tr>
<td>Sets and reads the spacing of the band tone density marker.</td>
<td>CALCulate:MEASure:SA:MARKer:BDENsity:TONE:TSPacing</td>
<td>BandDensityToneSpacing</td>
</tr>
<tr>
<td>Sets or returns the SA data threshold.</td>
<td>SENSE:SEGMenet:SA:DTHReshold</td>
<td>SADataThreshold</td>
</tr>
<tr>
<td>Specifies whether SA Data Threshold can be set independently for each segment.</td>
<td>SENSE:SEGMenet:SA:DTHReshold:CONTrol</td>
<td>SADataThresholdOption</td>
</tr>
<tr>
<td>Sets or returns the SA multitone reference.</td>
<td>SENSE:SEGMenet:SA:MTReference</td>
<td>SAMTReference</td>
</tr>
<tr>
<td>Specifies whether SA Reference Tone can be set independently for each segment.</td>
<td>SENSE:SEGMenet:SA:MTReference:CONTrol</td>
<td>SAMTReferenceFreqOption</td>
</tr>
<tr>
<td>Queries the maximum value of the SA Reference Tone, which is the maximum frequency.</td>
<td>SENSE:SEGMenet:SA:MTReference:MAX?</td>
<td>None</td>
</tr>
<tr>
<td>Queries the minimum value of the SA Reference Tone.</td>
<td>SENSE:SEGMenet:SA:MTReference:MIN?</td>
<td>None</td>
</tr>
<tr>
<td>Sets or returns the SA vector average points.</td>
<td>SENSE:SEGMenet:SA:VAVerage</td>
<td>SAVectorAverage</td>
</tr>
</tbody>
</table>
Specifies whether SA Vector Averaging can be set independently for each segment.

<table>
<thead>
<tr>
<th>Source Modulation Commands</th>
<th>SETSe:SEGMent:SA:VAverage:CONTrol</th>
<th>SAVectorAverageOption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets or returns the SA video bandwidth.</td>
<td>SETSe:SEGMent:SA:VIDeobw</td>
<td>SAVideoBandwidth</td>
</tr>
<tr>
<td>Specifies whether SA Video Bandwidth can be set independently for each segment.</td>
<td>SETSe:SEGMent:SA:VIDeobw:CONTrol</td>
<td>SAVideoAverageOption</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Modulation Commands</th>
<th>SETSe:SEGMent:SA:VAverage:CONTrol</th>
<th>SAVectorAverageOption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets or returns the SA video bandwidth.</td>
<td>SETSe:SEGMent:SA:VIDeobw</td>
<td>SAVideoBandwidth</td>
</tr>
<tr>
<td>Specifies whether SA Video Bandwidth can be set independently for each segment.</td>
<td>SETSe:SEGMent:SA:VIDeobw:CONTrol</td>
<td>SAVideoAverageOption</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Modulation Commands</th>
<th>SETSe:SEGMent:SA:VAverage:CONTrol</th>
<th>SAVectorAverageOption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets or returns the SA video bandwidth.</td>
<td>SETSe:SEGMent:SA:VIDeobw</td>
<td>SAVideoBandwidth</td>
</tr>
<tr>
<td>Specifies whether SA Video Bandwidth can be set independently for each segment.</td>
<td>SETSe:SEGMent:SA:VIDeobw:CONTrol</td>
<td>SAVideoAverageOption</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Modulation Commands</th>
<th>SETSe:SEGMent:SA:VAverage:CONTrol</th>
<th>SAVectorAverageOption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets or returns the SA video bandwidth.</td>
<td>SETSe:SEGMent:SA:VIDeobw</td>
<td>SAVideoBandwidth</td>
</tr>
<tr>
<td>Specifies whether SA Video Bandwidth can be set independently for each segment.</td>
<td>SETSe:SEGMent:SA:VIDeobw:CONTrol</td>
<td>SAVideoAverageOption</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Modulation Commands</th>
<th>SETSe:SEGMent:SA:VAverage:CONTrol</th>
<th>SAVectorAverageOption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets or returns the SA video bandwidth.</td>
<td>SETSe:SEGMent:SA:VIDeobw</td>
<td>SAVideoBandwidth</td>
</tr>
<tr>
<td>Specifies whether SA Video Bandwidth can be set independently for each segment.</td>
<td>SETSe:SEGMent:SA:VIDeobw:CONTrol</td>
<td>SAVideoAverageOption</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Modulation Commands</th>
<th>SETSe:SEGMent:SA:VAverage:CONTrol</th>
<th>SAVectorAverageOption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets or returns the SA video bandwidth.</td>
<td>SETSe:SEGMent:SA:VIDeobw</td>
<td>SAVideoBandwidth</td>
</tr>
<tr>
<td>Specifies whether SA Video Bandwidth can be set independently for each segment.</td>
<td>SETSe:SEGMent:SA:VIDeobw:CONTrol</td>
<td>SAVideoAverageOption</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Modulation Commands</th>
<th>SETSe:SEGMent:SA:VAverage:CONTrol</th>
<th>SAVectorAverageOption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets or returns the SA video bandwidth.</td>
<td>SETSe:SEGMent:SA:VIDeobw</td>
<td>SAVideoBandwidth</td>
</tr>
<tr>
<td>Specifies whether SA Video Bandwidth can be set independently for each segment.</td>
<td>SETSe:SEGMent:SA:VIDeobw:CONTrol</td>
<td>SAVideoAverageOption</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Modulation Commands</th>
<th>SETSe:SEGMent:SA:VAverage:CONTrol</th>
<th>SAVectorAverageOption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets or returns the SA video bandwidth.</td>
<td>SETSe:SEGMent:SA:VIDeobw</td>
<td>SAVideoBandwidth</td>
</tr>
<tr>
<td>Specifies whether SA Video Bandwidth can be set independently for each segment.</td>
<td>SETSe:SEGMent:SA:VIDeobw:CONTrol</td>
<td>SAVideoAverageOption</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Modulation Commands</th>
<th>SETSe:SEGMent:SA:VAverage:CONTrol</th>
<th>SAVectorAverageOption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets or returns the SA video bandwidth.</td>
<td>SETSe:SEGMent:SA:VIDeobw</td>
<td>SAVideoBandwidth</td>
</tr>
<tr>
<td>Specifies whether SA Video Bandwidth can be set independently for each segment.</td>
<td>SETSe:SEGMent:SA:VIDeobw:CONTrol</td>
<td>SAVideoAverageOption</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Modulation Commands</th>
<th>SETSe:SEGMent:SA:VAverage:CONTrol</th>
<th>SAVectorAverageOption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets or returns the SA video bandwidth.</td>
<td>SETSe:SEGMent:SA:VIDeobw</td>
<td>SAVideoBandwidth</td>
</tr>
<tr>
<td>Specifies whether SA Video Bandwidth can be set independently for each segment.</td>
<td>SETSe:SEGMent:SA:VIDeobw:CONTrol</td>
<td>SAVideoAverageOption</td>
</tr>
<tr>
<td>Function Description</td>
<td>Command/Function</td>
<td>Default Value</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------------</td>
<td>---------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Sets and reads the desired flatness calibration tolerance for the flatness modulation</td>
<td>SOURce:MODulation:CORRection:COLLection:FLATness:TOLerance</td>
<td>None</td>
</tr>
<tr>
<td>calibration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations to provide the deepest notch.</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:ITERations</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration plane for a notch modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the notch modulation calibration state.</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:ENABle</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired notch calibration tolerance for the notch modulation</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:TOLerance</td>
<td>None</td>
</tr>
<tr>
<td>calibration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations for a power modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:ITERations</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration plane for a power modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the power modulation calibration state.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:ENABle</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired power calibration tolerance for the power modulation</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:TOLerance</td>
<td>None</td>
</tr>
<tr>
<td>calibration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set and read the modulation correction state.</td>
<td>SOURce:MODulation:CORRection[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Returns a list of modulation files (*.mdx).</td>
<td>SOURce:MODulation:FILE</td>
<td>None</td>
</tr>
<tr>
<td>Loads the specified modulation file.</td>
<td>SOURce:MODulation:LOAD</td>
<td>None</td>
</tr>
<tr>
<td>Saves the specified modulation file.</td>
<td>SOURce:MODulation:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Checks if pulse source exists.</td>
<td>SOURce:PULSe:EXISts?</td>
<td>None</td>
</tr>
<tr>
<td>Turns pulse modulation on and off with an external source.</td>
<td>SOURce:PULSe:MODulation[:STATe]</td>
<td>None</td>
</tr>
</tbody>
</table>
Enable fast calibration.

<table>
<thead>
<tr>
<th>Power Meter Uncertainty Commands</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns a list of available power meters that have power uncertainty.</td>
<td>SYStem:CONFigure:EDEVice:PMAR:UNCertainty:CATalog?</td>
</tr>
<tr>
<td>Sets and returns a custom model uncertainty file containing all of the power meter uncertainty properties.</td>
<td>SYStem:CONFigure:EDEVice:PMAR:UNCertainty:FILE</td>
</tr>
<tr>
<td>Sets and returns the name assigned to a specific power meter model among those available for uncertainty.</td>
<td>SYStem:CONFigure:EDEVice:PMAR:UNCertainty:MODel</td>
</tr>
<tr>
<td>Returns the power level associated with the best accuracy for a specific power meter.</td>
<td>SYStem:CONFigure:EDEVice:PMAR:UNCertainty:PLEVel?</td>
</tr>
<tr>
<td>Returns the power uncertainty associated with the specific power meter at the specified frequency and power.</td>
<td>SYStem:CONFigure:EDEVice:PMAR:UNCertainty:READ?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pulse Generator Commands</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the ADC delay.</td>
<td>SENS:PUlSe:HDELay:ADC?</td>
</tr>
<tr>
<td>Sets the time lag between the pulse drive signal and the actual RF output.</td>
<td>SENS:PUlSe:HDELay:MODulator</td>
</tr>
<tr>
<td>Enables / Disables offset delays.</td>
<td>SENS:PUlSe:HDELay[:STATe]</td>
</tr>
<tr>
<td>Enables / Disables pulse4 to use an oscilloscope connected to pin 13 of the PULSE I/O connector on the rear panel of the VNA to display when the ADC is making measurements.</td>
<td>SENS&lt;ch&gt;:PUlSe4:OPTion</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Misc. Commands</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and returns the wave correction method.</td>
<td>SENS:CORRection:METHods:WAVE</td>
</tr>
</tbody>
</table>
The following are new programming commands for VNA release A.13.25.xx See What's New.

### Spectrum Analyzer Commands

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enables/disables data level threshold mode.</td>
<td>SENSe:SA:DATA:THReshold[:STATe]</td>
<td>DataLevelThresholdEnabled</td>
</tr>
<tr>
<td>Sets and returns the threshold value (dBm).</td>
<td>SENSe:SA:DATA:THReshold:VALue</td>
<td>DataLevelThreshold</td>
</tr>
<tr>
<td>Returns the windowing factor for band power computation.</td>
<td>SENSe:SA:DATA:WFACtor?</td>
<td>DataExportWindowingFactor</td>
</tr>
</tbody>
</table>

### Millimeter Wave Commands

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return the calibration date of the active configuration's test set.</td>
<td>SYSTem:CONFigure:MWAVe:CONF:ACTive:CALibration:DATE?</td>
<td>None</td>
</tr>
<tr>
<td>Return the calibration time of the active configuration's test set.</td>
<td>SYSTem:CONFigure:MWAVe:CONF:ACTive:CALibration:TIME?</td>
<td>None</td>
</tr>
<tr>
<td>Return the calibration date of the active configuration's port.</td>
<td>SYSTem:CONFigure:MWAVe:CONF:ACTive:PORT:CALibration:DATE?</td>
<td>None</td>
</tr>
<tr>
<td>Return the calibration time of the active configuration's port.</td>
<td>SYSTem:CONFigure:MWAVe:CONF:ACTive:PORT:CALibration:TIME?</td>
<td>None</td>
</tr>
</tbody>
</table>

### Misc. Commands

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and returns the power level knob resolution.</td>
<td>DISPlay:GUI:POWer:SPIN:RESolution</td>
<td>PowerSpinResolution</td>
</tr>
<tr>
<td>Returns the word size (32 or 64).</td>
<td>SYSTem:CONFigure:BIT?</td>
<td>None</td>
</tr>
</tbody>
</table>
The following are new programming commands for VNA release A.13.20.xx See What's New.

### Enhanced Time Domain Analysis (TDR)

**Setup and execution of TDR measurement Commands**

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets the type of parameter and format allocation for each trace.</td>
<td>CALCulate:TDR:ALLocate</td>
<td>None</td>
</tr>
<tr>
<td>Sets the DUT topology.</td>
<td>CALCulate:TDR:DEVice</td>
<td>None</td>
</tr>
<tr>
<td>Sets the emphasis post1 level.</td>
<td>CALCulate:TDR:EMPHasis:CURSor:POST1</td>
<td>None</td>
</tr>
<tr>
<td>Sets the emphasis post2 level.</td>
<td>CALCulate:TDR:EMPHasis:CURSor:POST2</td>
<td>None</td>
</tr>
<tr>
<td>Sets the emphasis pre1 level.</td>
<td>CALCulate:TDR:EMPHasis:CURSor:PRE1</td>
<td>None</td>
</tr>
<tr>
<td>Turns the emphasis function state ON or OFF.</td>
<td>CALCulate:TDR:EMPHasis:STATe</td>
<td>None</td>
</tr>
<tr>
<td>Sets the equalization CTLE DC gain parameter.</td>
<td>CALCulate:TDR:EQUalization:CTLE:DC</td>
<td>None</td>
</tr>
<tr>
<td>Sets the equalization CTLE Pole1 parameter.</td>
<td>CALCulate:TDR:EQUalization:CTLE:POLE1</td>
<td>None</td>
</tr>
<tr>
<td>Sets the equalization CTLE Pole2 parameter.</td>
<td>CALCulate:TDR:EQUalization:CTLE:POLE2</td>
<td>None</td>
</tr>
<tr>
<td>Sets the equalization CTLE zero parameter.</td>
<td>CALCulate:TDR:EQUalization:CTLE:ZERO1</td>
<td>None</td>
</tr>
<tr>
<td>Sets the filename of the equalization equation user file.</td>
<td>CALCulate:TDR:EQUalization:FILENAME</td>
<td>None</td>
</tr>
<tr>
<td>Turns the equalization function state ON or OFF.</td>
<td>CALCulate:TDR:EQUalization:STATe</td>
<td>None</td>
</tr>
<tr>
<td>Sets the equalization type.</td>
<td>CALCulate:TDR:EQUalization:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Executes the calculation for the simulated eye diagram for the active trace.</td>
<td>CALCulate:TDR:EYE:EXECute</td>
<td>None</td>
</tr>
<tr>
<td>Sets the bits' power of 2 for a PRBS pattern.</td>
<td>CALCulate:TDR:EYE:INPut:BPATtern:LENGth</td>
<td>None</td>
</tr>
<tr>
<td>Sets the bit pattern type for the simulated eye function.</td>
<td>CALCulate:TDR:EYE:INPut:BPATtern:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Sets the bit rate in bits/sec for the simulated eye function.</td>
<td>CALCulate:TDR:EYE:INPut:DRATe</td>
<td>None</td>
</tr>
<tr>
<td>Sets the display limit value.</td>
<td>CALCulate:TDR:EYE:INPut:JITTer:DLIMit</td>
<td>None</td>
</tr>
<tr>
<td>Sets the periodic jitter frequency.</td>
<td>CALCulate:TDR:EYE:INPut:JITTer:PERiodic:FREQuency</td>
<td>None</td>
</tr>
<tr>
<td>Sets the periodic jitter magnitude in rms.</td>
<td>CALCulate:TDR:EYE:INPut:JITTer:PERiodic:MAGNitude</td>
<td>None</td>
</tr>
<tr>
<td>Sets the random jitter magnitude in rms.</td>
<td>CALCulate:TDR:EYE:INPut:JITTer:RANDom:MAGNitude</td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Default</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Turns the jitter function state with simulated eye ON or OFF.</td>
<td><code>CALCulate:TDR:EYE:INPut:JITTer:STATe</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the jitter function type for the simulated eye function.</td>
<td><code>CALCulate:TDR:EYE:INPut:JITTer:TYPE</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the voltage level for bit &quot;1&quot; for the simulated eye function.</td>
<td><code>CALCulate:TDR:EYE:INPut:OLEVel</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the rise time value for the simulated eye.</td>
<td><code>CALCulate:TDR:EYE:INPut:RTIMe:DATA</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the rise time threshold for the simulated eye.</td>
<td><code>CALCulate:TDR:EYE:INPut:RTIMe:THReshold</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the voltage level for bit &quot;0&quot; for the simulated eye function.</td>
<td><code>CALCulate:TDR:EYE:INPut:ZLEVel</code></td>
<td>None</td>
</tr>
<tr>
<td>Returns the mask test result.</td>
<td><code>CALCulate:TDR:EYE:MASK:FAIL?</code></td>
<td>None</td>
</tr>
<tr>
<td>Turns the mask test function state with simulated eye ON or OFF.</td>
<td><code>CALCulate:TDR:EYE:MASK:STATe</code></td>
<td>None</td>
</tr>
<tr>
<td>Returns the results of the eye measurement.</td>
<td><code>CALCulate:TDR:EYE:RESults:DATA?</code></td>
<td>None</td>
</tr>
<tr>
<td>Turns the overlay ON or OFF.</td>
<td><code>CALCulate:TDR:EYE:RESults:DISPlay:STATe</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the rise time threshold level for the results of eye measurement.</td>
<td><code>CALCulate:TDR:EYE:RESults:THReshold</code></td>
<td>None</td>
</tr>
<tr>
<td>Turns the Eye/Mask window ON or OFF.</td>
<td><code>CALCulate:TDR:EYE:STATe</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets active marker number.</td>
<td><code>CALCulate:TDR:MEASure[1-16]:ACTive:MARKer</code></td>
<td>None</td>
</tr>
<tr>
<td>Returns the delta time result value.</td>
<td><code>CALCulate:TDR:MEASure[1-16]:DTIMe:DATA?</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets delta time reference position.</td>
<td><code>CALCulate:TDR:MEASure[1-16]:DTIMe:POsition</code></td>
<td>None</td>
</tr>
<tr>
<td>Turns the delta time marker in the marker search ON or OFF.</td>
<td><code>CALCulate:TDR:MEASure[1-16]:DTIMe:STATe</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the target trace number for the delta time function.</td>
<td><code>CALCulate:TDR:MEASure[1-16]:DTIMe:TARGET</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the trace format.</td>
<td><code>CALCulate:TDR:MEASure[1-16]:FORMat</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the measurement parameter.</td>
<td><code>CALCulate:TDR:MEASure[1-16]:PARameter</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets state for the peeling function.</td>
<td><code>CALCulate:TDR:MEASure[1-16]:PEELing:STATe</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets state for the smoothing function.</td>
<td><code>CALCulate:TDR:MEASure[1-16]:SMOothing:STATe</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the impulse width value for the transform function.</td>
<td><code>CALCulate:TDR:MEASure[1-16]:TIME:IMPulse:WIDTh</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets rise time value for the transform function.</td>
<td><code>CALCulate:TDR:MEASure[1-16]:TIME:STEP:RTIMe</code></td>
<td>None</td>
</tr>
<tr>
<td>Function</td>
<td>Command</td>
<td>None</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
<td>------</td>
</tr>
<tr>
<td>Sets the rise time threshold level for the results of eye measurement.</td>
<td>CALCulate:TDR:MEASure[1-16]:TIME:STEP:RTIMe:THReshold</td>
<td>None</td>
</tr>
<tr>
<td>Sets the stimulus type for the transform function.</td>
<td>CALCulate:TDR:MEASure[1-16]:TIME:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Returns the rise time result value for marker search.</td>
<td>CALCulate:TDR:MEASure[1-16]:TTIMe:DATA?</td>
<td>None</td>
</tr>
<tr>
<td>Displays the rise time marker.</td>
<td>CALCulate:TDR:MEASure[1-16]:TTIMe:STATe</td>
<td>None</td>
</tr>
<tr>
<td>Sets the rise time threshold for the rise time in the marker search function.</td>
<td>CALCulate:TDR:MEASure[1-16]:TTIMe:THReshold</td>
<td>None</td>
</tr>
</tbody>
</table>

### Display TDR Commands

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turns the continuous auto-scale mode for the eye y-axis ON or OFF.</td>
<td>DISPlay:TDR:EYE:Y:SCALe:AUTO:STATe</td>
<td>None</td>
</tr>
<tr>
<td>Sets the value of the y-axis scale per division for eye diagram.</td>
<td>DISPlay:TDR:EYE:Y:SCALe:PDIVision</td>
<td>None</td>
</tr>
<tr>
<td>Sets the value of the eye diagram y-axis reference line.</td>
<td>DISPlay:TDR:EYE:Y:SCALe:RLEVel</td>
<td>None</td>
</tr>
<tr>
<td>Sets the value of the eye diagram y-axis reference position.</td>
<td>DISPlay:TDR:EYE:Y:SCALe:RPOSition</td>
<td>None</td>
</tr>
<tr>
<td>Changes the background color of the screen.</td>
<td>DISPlay:TDR:IMAGe</td>
<td>None</td>
</tr>
<tr>
<td>Sets the display to off, data type, memory type, or data and memory type.</td>
<td>DISPlay:TDR:MEASure[1-16]:DMEMory:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Executes x-axis auto scaling.</td>
<td>DISPlay:TDR:MEASure[1-16]:X:SCALe:AUTO</td>
<td>None</td>
</tr>
<tr>
<td>Sets the value of the x-axis scale per division.</td>
<td>DISPlay:TDR:MEASure[1-16]:X:SCALe:PDIVision</td>
<td>None</td>
</tr>
<tr>
<td>Sets the value of the x-axis reference line.</td>
<td>DISPlay:TDR:MEASure[1-16]:X:SCALe:RLEVel</td>
<td>None</td>
</tr>
<tr>
<td>Executes y-axis auto scaling.</td>
<td>DISPlay:TDR:SCALe:AUTO</td>
<td>None</td>
</tr>
<tr>
<td>Sets the x-axis reference position for the time domain measurement.</td>
<td>DISPlay:TDR:SCALe:RPOSition</td>
<td>None</td>
</tr>
<tr>
<td>Selects the view point for waveform analysis either before or after the DUT.</td>
<td>DISPlay:TDR:VIEW</td>
<td>None</td>
</tr>
<tr>
<td>Sets or gets the minimize state.</td>
<td>DISPlay:TDR:MINimize:STATe</td>
<td>None</td>
</tr>
</tbody>
</table>

### Memory TDR Commands

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loads the specified user bit pattern file.</td>
<td>MMEMory:TDR:LOAD[:EYE]:BPATtern</td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Notes</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>---------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Loads eye-mask file.</td>
<td>MMEMory:TDR:LOAD[:EYE][:MASK]</td>
<td>None</td>
</tr>
<tr>
<td>Stores the user bit pattern file.</td>
<td>MMEMory:TDR:STORe[:EYE]:BPATtern</td>
<td>None</td>
</tr>
<tr>
<td>Stores the eye-mask file.</td>
<td>MMEMory:TDR:STORe[:EYE][:MASK]</td>
<td>None</td>
</tr>
<tr>
<td>Correction TDR Commands</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Executes load measurement.</td>
<td>SENSE:CORRection:TDR:COLLection:DLComp:LOAD</td>
<td>None</td>
</tr>
<tr>
<td>Executes open measurement.</td>
<td>SENSE:CORRection:TDR:COLLection:DLComp:OPEN</td>
<td>None</td>
</tr>
<tr>
<td>Saves the result of the Loss Compensation sequence.</td>
<td>SENSE:CORRection:TDR:COLLection:DLComp:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Executes a thru measurement.</td>
<td>SENSE:CORRection:TDR:COLLection:DLComp:THRU</td>
<td>None</td>
</tr>
<tr>
<td>Executes fixture compensation after ECAL.</td>
<td>SENSE:CORRection:TDR:COLLection:ECAL:FCOMp:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>Executes full calibration using the ECal module.</td>
<td>SENSE:CORRection:TDR:COLLection:ECAL:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>Sets the dielectric constant value.</td>
<td>SENSE:CORRection:TDR:DCONstant</td>
<td>None</td>
</tr>
<tr>
<td>Executes deskew (auto port extension).</td>
<td>SENSE:CORRection:TDR:EXTension:AUTO:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>Sets the standard for auto port extension.</td>
<td>SENSE:CORRection:TDR:EXTension:AUTO:STANdard</td>
<td>None</td>
</tr>
<tr>
<td>Sets the reference impedance value.</td>
<td>SENSE:CORRection:TDR:RIMPedance</td>
<td>None</td>
</tr>
<tr>
<td>Bandwidth, DUT information, avoid spurious function, and sweep TDR Commands</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets the IF bandwidth value.</td>
<td>SENSE:TDR:BWIDth[:RESolution]</td>
<td>None</td>
</tr>
<tr>
<td>Executes auto DUT length setting.</td>
<td>SENSE:TDR:DLENgth:AUTO:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>Sets the DUT length value.</td>
<td>SENSE:TDR:DLENgth:DATA</td>
<td>None</td>
</tr>
<tr>
<td>Executes avoid spurious.</td>
<td>SENSE:TDR:SPURious:AVOid:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>Queries the avoid spurious state.</td>
<td>SENSE:TDR:SPURious:AVOid:STATe?</td>
<td>None</td>
</tr>
<tr>
<td>Sets the value of input bit rate for avoid spurious.</td>
<td>SENSE:TDR:SPURious:INPut:DRATe</td>
<td>None</td>
</tr>
<tr>
<td>Queries the Hot TDR mode status.</td>
<td>SENSE:TDR:SPURious:STATe?</td>
<td>None</td>
</tr>
<tr>
<td>Sets the TDR averaging trigger state on/off.</td>
<td>SENSE:TDR:SWEep:AVERage</td>
<td>None</td>
</tr>
<tr>
<td>Sets trigger mode.</td>
<td>SENSE:TDR:SWEep:MODE</td>
<td>None</td>
</tr>
<tr>
<td>Executes single trigger.</td>
<td>SENSE:TDR:SWEep:SINGle</td>
<td>None</td>
</tr>
<tr>
<td>Enables/disables the Reduce IF BW at Low Frequencies feature in segments with IFBW arbitrary.</td>
<td>SENSE:TDR:BWIDth:TRACK:FORCe</td>
<td>None</td>
</tr>
</tbody>
</table>

Source Power Level TDR Command

Sets the source power level.  SOURce:TDR:POWer[:LEVel][:IMMediate][:AMPLitude]  None
### Preset TDR Command

Executes a TDR preset.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:TDR:PRESet</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

The following are new programming commands for VNA release A.12.90.xx See What's New.

### Multi-Dimensional Sweep

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:DC:DIimension:ORDER</td>
<td>Set and read the order for the specified DC source in the multi-dimensional sweep.</td>
<td>DCOrder</td>
</tr>
<tr>
<td>SOURce:DC:DIimension[:STATE]</td>
<td>Set and read the specified DC source’s ON/OFF state in the multi-dimensional sweep.</td>
<td>DCState</td>
</tr>
<tr>
<td>SOURce:DIimension:CATalog?</td>
<td>Read the names of source domains in the multi-dimensional sweep whose state is ON and whose dimension order is the specified dimension order.</td>
<td>DimensionCatalog</td>
</tr>
<tr>
<td>SOURce:DIimension:COUNT?</td>
<td>Read the highest dimension order in the multi-dimensional sweep.</td>
<td>DimensionCount</td>
</tr>
<tr>
<td>SOURce:DIimension:POINts</td>
<td>Set and read the point count for the specified dimension order in the multi-dimensional sweep.</td>
<td>DimensionPointCount</td>
</tr>
<tr>
<td>SOURce:DIimension:REPeat:COUNt</td>
<td>Set and read the repeat count for the specified dimension order in the multi-dimensional sweep.</td>
<td>DimensionRepeatCount</td>
</tr>
<tr>
<td>SOURce:FREQuency:DIimension:ORDER</td>
<td>Set and read the source frequency domain’s order in the multi-dimensional sweep.</td>
<td>SourcePortFrequencyOrder</td>
</tr>
<tr>
<td>SOURce:FREQuency:DIimension[:STATE]</td>
<td>Set and read the source frequency domain’s ON/OFF state in the multi-dimensional sweep.</td>
<td>SourcePortFrequencyState</td>
</tr>
<tr>
<td>SOURce:FREQuency:FIXed</td>
<td>Set and read the fixed frequency value for a specific port.</td>
<td>SourcePortFixedFrequency</td>
</tr>
<tr>
<td>SOURce:FREQuency:STARt</td>
<td>Set and read the start frequency value for a specific port.</td>
<td>SourcePortStartFrequency</td>
</tr>
<tr>
<td>SOURce:FREQuency:STOP</td>
<td>Set and read the stop frequency value for a specific port.</td>
<td>SourcePortStopFrequency</td>
</tr>
<tr>
<td>SOURce:PHASe:DIimension:ORDER</td>
<td>Set and read the source phase domain’s order in the multi-dimensional sweep.</td>
<td>SourcePortPhaseOrder</td>
</tr>
<tr>
<td>SOURce:PHASe:DIimension[:STATE]</td>
<td>Set and read the source phase domain’s ON/OFF state in the multi-dimensional sweep.</td>
<td>SourcePortPhaseState</td>
</tr>
<tr>
<td>SOURce:POWer:DIimension:ORDER</td>
<td>Set and read the source power domain’s order in the multi-dimensional sweep.</td>
<td>SourcePortPowerOrder</td>
</tr>
</tbody>
</table>
Set and read the source power domain’s ON/OFF state in the multi-dimensional sweep.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:POWer:DIManion[:STATe]</td>
<td>SourcePortPowerState</td>
<td></td>
</tr>
</tbody>
</table>

**Active Hot Parameters**

- Sets whether or not interpolation is on for display.
  - SENS:ACTive:DISP:INTerpolate[:STATe] DisplayInterpolationState
- Set and read a fixed input power level.
  - SENS:ACTive:DISPlay:TRACe:IPWer DisplayInputPower
- Set and read the number of phase points.
  - SENS:ACTive:SWEep:PHASE:POINt PhaseSweepPoints
- Set and read the start power level for a 3D sweep.
  - SENS:ACTive:SWEep:POWer:STARt StartPowerIn3DSweep
- Set and read the number of power steps for a 3D sweep.
  - SENS:ACTive:SWEep:POWer:STEP PowerStepsIn3DSweep
- Set and read the stop power level for a 3D sweep.
  - SENS:ACTive:SWEep:POWer:STOP StopPowerIn3DSweep
- Set and read the sweep type.
  - SENS:ACTive:SWEep:TYPE SweepType
- Set and read the tuning tone mode.
  - SENS:ACTive:TTONe:MODE ExtractionToneMode
- Set and read the absolute tone power level.
  - SENS:ACTive:TTONe:ABSolute AbsoluteExtractionToneLevel
- Set and read the tone power relative to the input power (dBc).
  - SENS:ACTive:TTONe:RELative RelativeExtractionToneLevel
- Set and read the X-axis domain type.
  - CALC:X:AXIS:DOMain DisplayDomain

**Millimeter Wave Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return the model number of the active test set.</td>
<td>SYSTem:CONFigure:MWAve:CONF:ACTive:MODel?</td>
<td>None</td>
</tr>
<tr>
<td>Return the option number of the active test set.</td>
<td>SYSTem:CONFigure:MWAve:CONF:ACTive:OPTion?</td>
<td>None</td>
</tr>
<tr>
<td>Return the model number of the frequency extender module connected to the specified port number.</td>
<td>SYSTem:CONFigure:MWAve:CONF:ACTive:PORT{1:4}:MODel?</td>
<td>None</td>
</tr>
<tr>
<td>Return the option number of the frequency extender module connected to the specified port number.</td>
<td>SYSTem:CONFigure:MWAve:CONF:ACTive:PORT{1:4}:OPTion?</td>
<td>None</td>
</tr>
</tbody>
</table>
Spectrum Analyzer

Set and read the Nyquist protection level.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:MULTitone:NYQReject</td>
<td>MultiToneNyquistProtection</td>
<td></td>
</tr>
</tbody>
</table>

Read the current multitone settings and determine if they are valid or not.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:MULTitone:VALid</td>
<td>MultiToneSettingsValid</td>
<td></td>
</tr>
</tbody>
</table>

Misc. Commands

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:CALibrate:ALL</td>
<td>CalibrateAllChannelsEx</td>
</tr>
<tr>
<td>CSET:FIXTure:ZERO</td>
<td>ZeroTermsInS4PFile</td>
</tr>
</tbody>
</table>

The following are new programming commands for VNA release A.12.85.xx See What's New.

Dark shaded commands existed before this release. They are provided here for convenience.

Automatic Fixture Removal (AFR)

Selects whether the fixture is band limited or not.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:FIXTure:BLIMited[:STATe]</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Selects whether to use DUT correction or not when the characterization fixture is not equal to the DUT measurement fixture.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:FIXTure:CDUT[:STATe]</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Selects Fixture Length A not equal to B correction.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:FIXTure:CLENgth[:STATe]</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Selects Fixture Match A not equal to B correction.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:FIXTure:CMATch[:STATe]</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Describes the fixture inputs (single ended or differential).

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:FIXTure:INPuts</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Selects the number of fixtures to be characterized.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:FIXTure:MEASurement</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Refreshes preview data.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:FIXTure:PREView</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Reads the impedance profile of the calculated fixture model.

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFR:FIXTure:PREView:DATA[:IMPedance]?</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Command/Parameter</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Reads the impedance profile of the calculated fixture model at a specified position.</td>
<td>AFR:FIXTure:PREview:DATA[:IMPedance]:MARKer:Y?</td>
<td>None</td>
</tr>
<tr>
<td>Chooses the calibration reference Z0 after fixture removal.</td>
<td>AFR:FIXTure:REFZ</td>
<td>None</td>
</tr>
<tr>
<td>Restores the default AFR settings.</td>
<td>AFR:INITialize</td>
<td>None</td>
</tr>
<tr>
<td>Specifies the file paths of saved fixture data.</td>
<td>AFR:SAVE:FILENAME</td>
<td>None</td>
</tr>
<tr>
<td>Assigns the ports for saved fixture data in several formats.</td>
<td>AFR:SAVE:PORTS</td>
<td>None</td>
</tr>
<tr>
<td>Sets the file type to save fixture data.</td>
<td>AFR:SAVE:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Selects all OPEN standards.</td>
<td>AFR:STANdard:ALLOpen[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Selects all SHORT standards.</td>
<td>AFR:STANdard:ALLShort[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Reads the impedance profile of the measured standard.</td>
<td>AFR:STANdard:DATA[:IMPedance]?</td>
<td>None</td>
</tr>
<tr>
<td>Reads the impedance of the measured standard at a specified position.</td>
<td>AFR:STANdard:DATA[:IMPedance]:MARKer:Y?</td>
<td>None</td>
</tr>
<tr>
<td>Sets the fixture length for the selected fixture (for 1X AFR only).</td>
<td>AFR:STANdard:EDIT:FLENgth</td>
<td>None</td>
</tr>
<tr>
<td>Sets the gate position for the selected fixture.</td>
<td>AFR:STANdard:EDIT:GATE</td>
<td>None</td>
</tr>
<tr>
<td>Sets the impedance for the selected term.</td>
<td>AFR:STANdard:EDIT:IMPedance</td>
<td>None</td>
</tr>
<tr>
<td>Sets the impedance method.</td>
<td>AFR:STANdard:EDIT:IMPedance:METHod</td>
<td>None</td>
</tr>
<tr>
<td>Loads the calibration standards data from a file.</td>
<td>AFR:STANdard:LOAD</td>
<td>None</td>
</tr>
<tr>
<td>Specifies fixture thru settings.</td>
<td>AFR:STANdard:THRU</td>
<td>None</td>
</tr>
<tr>
<td>Chooses the calibration standards.</td>
<td>AFR:STANdard:USE</td>
<td>None</td>
</tr>
</tbody>
</table>
### Independent Power Calibration Commands

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:CALibrate:ALL:INDependent:SOURce:CALibrate:RANGe:POINts</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>SYSTem:CALibrate:ALL:INDependent:SOURce:CALibrate:RANGe:STARt</td>
<td>StartFrequency</td>
</tr>
</tbody>
</table>

### ISegment3 and ISegments6 Interface Commands

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SEGMenT:SWEep:DELay</td>
<td>DelayOption</td>
</tr>
<tr>
<td>SENSE:SEGMenT:BWIddth</td>
<td>BANDwidth:PORT[:RESolution]:CONTrol</td>
</tr>
<tr>
<td>SENSE:SEGMenT:SHLO</td>
<td>ShiftLOOption</td>
</tr>
<tr>
<td>SENSE:SEGMenT:SWEep:GENeration:CONTrol</td>
<td>SweepModeOption</td>
</tr>
<tr>
<td>ExportCSVfile</td>
<td>ExportCSVfile</td>
</tr>
<tr>
<td>ImportCSVfile</td>
<td>ImportCSVfile</td>
</tr>
</tbody>
</table>

The following are new programming commands for VNA release A.12.80.xx See What's New.
Dark shaded commands existed before this release. They are provided here for convenience.

<table>
<thead>
<tr>
<th>External DC Source/Meter</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set and return the DC Meter/DC Source Abort Sweep command</td>
<td>SYSTem:CONFigure:EDEVice:DC:COM曼:SWEEp:ABORt</td>
<td>AbortSweepCmd</td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source After Sweep command</td>
<td>SYSTem:CONFigure:EDEVice:DC:COM曼:SWEEp:AFTer</td>
<td>AfterSweepCmd</td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source Before Sweep command</td>
<td>SYSTem:CONFigure:EDEVice:DC:COM曼:SWEEp:BEFore</td>
<td>BeforeSweepCmd</td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source Error Query command</td>
<td>SYSTem:CONFigure:EDEVice:DC:QUERy:ERRor</td>
<td>ErrorQuery</td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source Disable I/O command</td>
<td>SYSTem:CONFigure:EDEVice:DC:COMMan:EXIT</td>
<td>ExitCmd</td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source ID Query command</td>
<td>SYSTem:CONFigure:EDEVice:DC:QUERy:ID</td>
<td>IDQuery</td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source Enable I/O command</td>
<td>SYSTem:CONFigure:EDEVice:DC:COMMan:INIT</td>
<td>InitCmd</td>
</tr>
<tr>
<td>Set and return the DC Source maximum output</td>
<td>SYSTem:CONFigure:EDEVice:DC:MAX:VALue</td>
<td>MaxOutput</td>
</tr>
<tr>
<td>Set and return the DC Source maximum output state</td>
<td>SYSTem:CONFigure:EDEVice:DC:MAX[:STATe]</td>
<td>MaxOutputState</td>
</tr>
<tr>
<td>Set and return the DC Source minimum output</td>
<td>SYSTem:CONFigure:EDEVice:DC:MIN:VALue</td>
<td>MinOutput</td>
</tr>
<tr>
<td>Set and return the DC Source minimum output state</td>
<td>SYSTem:CONFigure:EDEVice:DC:MIN[:STATe]</td>
<td>MinOutputState</td>
</tr>
<tr>
<td>Set and return the Point Read commands and Point Set commands</td>
<td>SYSTem:CONFigure:EDEVice:DC:COMMan:POINt:SET</td>
<td>PointCmd</td>
</tr>
<tr>
<td>Set and return the Max DC limit value for a DC source</td>
<td>SOURce:DC:LIMit:MAXimum</td>
<td>LimitMax</td>
</tr>
<tr>
<td>Set and return the Min DC limit value for a DC source</td>
<td>SOURce:DC:LIMit:MINimum</td>
<td>LimitMin</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Spectrum Analyzer Commands</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read current maximum available ADC record size</td>
<td>SENSe:SA:ADC:RECor:SIZE:MAX?</td>
<td>ADCRecordSizeMax</td>
</tr>
<tr>
<td>Read current minimum available ADC record size</td>
<td>SENSe:SA:ADC:RECor:SIZE:MIN?</td>
<td>ADCRecordSizeMin</td>
</tr>
<tr>
<td>Set and read stacking size</td>
<td>SENSe:SA:ADC:STACking:VALue</td>
<td>ADCStacking</td>
</tr>
</tbody>
</table>

5629
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Alias</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enables/disable multitone image rejection</td>
<td>SENSE:SA:COHerence:MULTitone[:STATe]</td>
<td>MultiToneImageRejectEnable</td>
</tr>
<tr>
<td>Set and return the data display mode</td>
<td>SENSE:SA:COHerence:MULTitone:DATA</td>
<td>MultiToneImageRejectDataDisplay</td>
</tr>
<tr>
<td>Set and return the tone spacing of the multitone signal (in Hz)</td>
<td>SENSE:SA:COHerence:MULTitone:SPACing</td>
<td>MultiToneImageRejectSpacing</td>
</tr>
<tr>
<td>Set and return the test signal repetition rate (in seconds)</td>
<td>SENSE:SA:COHerence:Multitone:PERiod</td>
<td>MultiToneImageRejectPeriod</td>
</tr>
<tr>
<td>Set and return the multitone image rejection offset frequency</td>
<td>SENSE:SA:COHerence:MULTitone:REFERENCE</td>
<td>MultiToneImageRejectReference</td>
</tr>
<tr>
<td>Sets and returns the number of test signal harmonics you want to be protected against</td>
<td>SENSE:SA:COHerence:MULTitone:HREject</td>
<td>MultiToneImageRejectHarmonic</td>
</tr>
<tr>
<td>Zooms in on signal within a SA sweep</td>
<td>SENSE:SA:FREQuency:TUNE:IMMediate</td>
<td>FrequencyAutoTune</td>
</tr>
<tr>
<td>Sets and returns the data format.</td>
<td>SENSE:SA:DATA:TYPE</td>
<td>DataFormat</td>
</tr>
<tr>
<td>Returns the frequency of the first RF bin.</td>
<td>SENSE:SA:DATA:START?</td>
<td>DataFirstRFBin</td>
</tr>
<tr>
<td>Returns the currently exported receiver list.</td>
<td>SENSE:SA:DATA:RECeivers?</td>
<td>ExportReceiverList</td>
</tr>
<tr>
<td>Sets and returns the list of exported receivers.</td>
<td>SENSE:SA:DATA:RECeivers:LIST</td>
<td>ExportReceiverSetList</td>
</tr>
<tr>
<td>Returns the number of currently exported receivers.</td>
<td>SENSE:SA:DATA:RECeivers:COUNT?</td>
<td>ExportReceiverCount</td>
</tr>
<tr>
<td>Returns the number of DFT points processed across the total RF span.</td>
<td>SENSE:SA:DATA:BINs:COUNT?</td>
<td>DataBinCount</td>
</tr>
<tr>
<td>Returns the byte size.</td>
<td>SENSE:SA:DATA:SIZE?</td>
<td>DataByteSize</td>
</tr>
<tr>
<td>Returns byte size of one data bin in binary mode.</td>
<td>SENSE:SA:DATA:SIZE:BIN?</td>
<td>DataBytesPerBin</td>
</tr>
<tr>
<td>Returns the least significant bytes.</td>
<td>SENSE:SA:DATA:SIZE:LOW?</td>
<td>DataByteSizeLSB</td>
</tr>
<tr>
<td>Returns the most significant bytes.</td>
<td>SENSE:SA:DATA:SIZE:HIGH?</td>
<td>DataByteSizeMSB</td>
</tr>
<tr>
<td>Enables output data files to be erased after each sweep.</td>
<td>SENSE:SA:DATA:FILE:ERASE[:STATe]</td>
<td>FileEraseEachSweep</td>
</tr>
<tr>
<td>Sets and returns the file name prefix for the data file.</td>
<td>SENSE:SA:DATA:FILE:PREFIX</td>
<td>FilePrefix</td>
</tr>
<tr>
<td>Feature</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>---------</td>
<td>---------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Enables exporting data to the FIFO (First-IN, First-OUT) data buffer.</td>
<td>SENSe:SA:DATA:FIFO[:STATe]</td>
<td>FIFOEnabled</td>
</tr>
<tr>
<td>Enables data to be output to shared memory.</td>
<td>SENSe:SA:DATA:SHARed[:STATe]</td>
<td>MemShareEnabled</td>
</tr>
<tr>
<td>Assigns a specified name to the shared data.</td>
<td>SENSe:SA:DATA:SHARed:NAME</td>
<td>MemShareName</td>
</tr>
<tr>
<td>Read the occupied bandwidth center frequency.</td>
<td>CALC:MEAS:SA:MARK:OCCBand:CENTer?</td>
<td>OccupiedBandCenter</td>
</tr>
<tr>
<td>Sets and reads the percentage of the band span to measure.</td>
<td>CALC:MEAS:SA:MARK:OCCBand:PERCent</td>
<td>OccupiedBandPercent</td>
</tr>
<tr>
<td>Read the occupied bandwidth power.</td>
<td>CALC:MEAS:SA:MARK:OCCBand:POWer?</td>
<td>OccupiedBandPowerdBm</td>
</tr>
<tr>
<td>Read the span of the occupied bandwidth.</td>
<td>CALC:MEAS:SA:MARK:OCCBand:SPAN?</td>
<td>OccupiedBandSpan</td>
</tr>
<tr>
<td>Sets and reads the occupied bandwidth on/off state.</td>
<td>CALC:MEAS:SA:MARK:OCCBand[:STATe]</td>
<td>OccupiedBandState</td>
</tr>
<tr>
<td>Sets and reads the minimum search frequency to use during an Occupied BW search measurement.</td>
<td>SENSe:SA:BANDwidth:SEARch:OCCupied:MIN</td>
<td>SearchOccupiedBWMinFreq</td>
</tr>
<tr>
<td>Set and read the ON/OFF state of the vector averaging.</td>
<td>SENSe:SA:COHerence:VECTor:AVERage[:STATe]</td>
<td>VectorAverageEnable</td>
</tr>
<tr>
<td>Set and read the vector averaging value.</td>
<td>SENSe:SA:COHerence:VECTor:AVERage:VALue</td>
<td>VectorAverageValue VectorAverageMax</td>
</tr>
</tbody>
</table>

### Low Frequency Extension Commands

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:SWEep:LFEXtension:STATe</td>
<td>LowFrequencyExtension</td>
</tr>
<tr>
<td>SYSTem:CAPability:HARDware:LFEXtensions:EXISts?</td>
<td>HasLowFrequencyExtension</td>
</tr>
</tbody>
</table>
## Millimeter Wave Commands

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reads the port count of the test set</td>
<td>SYSTem:CONFigure:MWAVe:TSET:PORT:COUNt?</td>
<td>None</td>
</tr>
<tr>
<td>Reads the serial number of the frequency extender module connected to the specified port number.</td>
<td>SYSTem:CONFigure:MWAVe:CONF:ACTive:PORT{1:4}:SERial?</td>
<td>None</td>
</tr>
<tr>
<td>Reads the serial number of the test set</td>
<td>SYSTem:CONFigure:MWAVe:CONF:ACTive:SERial?</td>
<td>None</td>
</tr>
</tbody>
</table>

## Misc. Commands

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checks if the PNA is ready for a hardware trigger</td>
<td>TRIGger:STATus:READY?</td>
<td>ReadyForTriggerStatus</td>
</tr>
<tr>
<td>Sets and returns ALC mode that the Pulse Setup will use when the pulse is turned off</td>
<td>SENSe:SWEep:PULSe:POFF:ALCmode</td>
<td>PulseOffAlcMode</td>
</tr>
<tr>
<td>Reads the FIFO data byte count.</td>
<td>SYST:FIFO:DATA:BYTe:COUNt?</td>
<td>DataByteCount</td>
</tr>
<tr>
<td>Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of bytes.</td>
<td>SYST:FIFO:DATA:BYTe?</td>
<td>DataAsBytes</td>
</tr>
<tr>
<td>Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of 32-bit floating point (Float32) numbers.</td>
<td></td>
<td>DataAsFloat32</td>
</tr>
<tr>
<td>Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of 16-bit integers.</td>
<td></td>
<td>DataAsInt16</td>
</tr>
<tr>
<td>Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of 32-bit integers.</td>
<td></td>
<td>DataAsInt32</td>
</tr>
<tr>
<td>Returns all cal all guided calibration channels.</td>
<td>SYSTem:CAL:ALL:GUIDed:CHANnel:LIST?</td>
<td>None</td>
</tr>
</tbody>
</table>

The following are new programming commands for VNA release A.12.70.xx See What's New.
## Error Correction

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Argument</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turn error correction ON or OFF</td>
<td><strong>CALCulate:MEASure:CORRection[:STATE]</strong></td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td>Returns the error correction state for the measurement</td>
<td><strong>CALCulate:MEASure:CORRection:INDicator?</strong></td>
<td>ErrorCorrectionIndicator</td>
</tr>
<tr>
<td>Set the Cal Type</td>
<td><strong>CALCulate:MEASure:CORRection:TYPE</strong></td>
<td>meas.CalibrationTypeID</td>
</tr>
<tr>
<td>Execute the Ecal calibration</td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:ECAL:ACQuire</strong></td>
<td>None</td>
</tr>
<tr>
<td>Specifies the Ecal Kit for Ecal Calibration</td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:ECAL:SESelect</strong></td>
<td>None</td>
</tr>
<tr>
<td>Compute Error Terms</td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:ETERms:COMPute</strong></td>
<td>None</td>
</tr>
<tr>
<td>Return Number of Steps in a Cal</td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:LIST:COUNt?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Return number of standards for step[n]</td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:LIST:STEP:COUNt?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Return step description</td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:LIST:STEP:DESCRIPTION?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Return label for complete standard</td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:LIST:STEP:LABEL?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Return number of ports on standard used in the step</td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:LIST:STEP:PORTs?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Return label for one of the standards in the step</td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:LIST:STEP:STANDARD:LABEL?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Return number of ports on one of the standards used in the step</td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:LIST:STEP:STANDARD:PORTs?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Return the enumeration for the type of standard</td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:LIST:STEP:STANDARD:STYPE?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Return list of VNA test ports to which one of the standards is attached</td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:LIST:STEP:STANDARD:TPORTs?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Example</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Return enumeration for the type of standard device used in the step</td>
<td>SENSE:CORRection:COLLect:GUIDed:LIST:STEP:STYPE?</td>
<td>None</td>
</tr>
<tr>
<td>Return list of VNA test ports to which the standard(s) in this step is attached</td>
<td>SENSE:CORRection:COLLect:GUIDed:LIST:STEP:TPORTs?</td>
<td>None</td>
</tr>
<tr>
<td>Get list of name-value pairs from cal set</td>
<td>SENSE:CORRection:CSET:ITEM:CAT?</td>
<td>Item</td>
</tr>
<tr>
<td>Add or change a name-value pair in the Cal Set</td>
<td>SENSE:CORRection:CSET:ITEM[:DATA]?</td>
<td>None</td>
</tr>
<tr>
<td>Quickly test a prototype of automation software</td>
<td>SENSE:CORRection:CSET:CREate:DEFault</td>
<td>None</td>
</tr>
<tr>
<td>Return measurement parameters measured in the specified step number of a guided calibration</td>
<td>SENSE:CORRection:COLLect:GUIDed:DATA:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Set and return measurement data for a specified measurement parameter of a particular step of a guided cal</td>
<td>SENSE:CORRection:COLLect:GUIDed:DATA</td>
<td>None</td>
</tr>
<tr>
<td>Return list of ports being calibrated by an active calibration session</td>
<td>SENSE:CORRection:COLLect:GUIDed:PORTs?</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the selected ports to include in a full NPort correction.</td>
<td>SENSE:CORRection:METHods:PORT:SUBSet:FULL[:VALue]</td>
<td>FullyCorrectedPorts</td>
</tr>
<tr>
<td>Resets the full and response list to their default values.</td>
<td>SENSE:CORRection:METHods:PORT:SUBSet:RESet</td>
<td>ResetPortValues</td>
</tr>
<tr>
<td>Sets and returns the selected ports to be corrected with enhanced response calibration.</td>
<td>SENSE:CORRection:METHods:PORT:SUBSet:RESPonse[:VALue]</td>
<td>ResponseCorrectedPorts</td>
</tr>
<tr>
<td>Set and return the ON/OFF subset correction state.</td>
<td>SENSe:CORRection:METHods:PORT:SUBSet:[.STATE]</td>
<td>CorrectionSubsettingState</td>
</tr>
<tr>
<td>Enable/disable use of error messages during a source calibration if calibration fails to achieve desired power level at the power sensor</td>
<td>SOURce:POWer:CORRection:COLLect:WARN</td>
<td>None</td>
</tr>
</tbody>
</table>

### Read Measurement Data

| Get formatted data array of multiple traces | CALCulate:DATA:MFData? | None |
| Get corrected data array of multiple traces | CALCulate:DATA:MSData? | None |

| Get receiver data | CALCulate:MEASure:RDATA? | IArrayTrans.getComplex  
IArrayTrans.getNAComplex  
IArrayTrans.getPairedData  
IArrayTrans.getScalar  
meas.GetDataByString |

### Topology

| Maps the physical VNA ports to a device of balanced and single-ended logical ports for multi-port systems with greater than 4 ports | CALCulate:DTOPology | BalancedTopology Object |
| For a Balanced device type, maps the VNA ports to the DUT ports | CALCulate:FSIMulator:BAUn:TOPology:BAIanced:PPORts |
| Specifies the VNA port connections for VNAs having greater than 4 ports | CALCulate:FSIMulator:EMBed:TOPology:D:PORTs | Embed4PortList |

### Balanced Measurements and Fixturing
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select measurement parameter for the specified trace for a balanced device type</td>
<td>CALCulate:FSIMulator:BAUn:PARameter:BALanced:DEFine</td>
<td>ChangeParameter</td>
</tr>
<tr>
<td>Returns the list of parameters available for the currently selected topology</td>
<td>CALCulate:FSIMulator:BAUn:PARameter:CATalog?</td>
<td>Parameter</td>
</tr>
<tr>
<td>Defines a balanced measurement parameter corresponding to a custom topology for systems where the port count is expandable beyond 4 ports.</td>
<td>CALCulate:FSIMulator:BAUn:PARameter:CUSTom[:DEFine]</td>
<td>ChangeParameter</td>
</tr>
</tbody>
</table>

**De-embedding**

- Sets and returns the Capacitance value
  - Command: CALCulate:FSIMulator:GLOop:DEEMbed:PARameters:C
  - Type: None
- Sets and returns the Inductance value
  - Command: CALCulate:FSIMulator:GLOop:DEEMbed:PARameters:L
  - Type: None
- Sets and returns the Resistance value
  - Type: None
- Turns ON or OFF De-embedding
  - Command: CALCulate:FSIMulator:GLOop:DEEMbed:STATe
  - Type: None
- Specifies the circuit model type
  - Command: CALCulate:FSIMulator:GLOop:DEEMbed:TYPE
  - Type: None
- Specifies the filename of the s1p file to load
  - Type: None

**Embedding**

- Sets and returns the Capacitance value
  - Type: None
- Sets and returns the Inductance value
  - Type: None
- Sets and returns the Resistance value
  - Type: None
- Turns ON or OFF Embedding
  - Type: None
- Specifies the circuit model type
  - Type: None
- Specifies the filename of the s1p file to load
  - Type: None
| Limits |
|-----------------|-----------------|
| Read the bandwidth test results for the active trace of selected channel | CALCulate:MEASure:LImit:REPort:ALL |
| | CALCulate:LImit:REPort:ALL |
| Read stimulus values at all the measurement points that failed the limit test | CALCulate:MEASure:LImit:REPort:DATA |
| | CALCulate:LImit:REPort:DATA |
| Read number of measurement points that failed the limit test | CALCulate:MEASure:LImit:REPort:POINts |
| | CALCulate:LImit:REPort:POINts |
| Set / Read entire Limit Line | CALCulate:MEASure:LImit:DATA |
| Delete all limit line data | CALCulate:MEASure:LImit:DATA:DELeate |
| Display Lines ON/OFF | CALCulate:MEASure:LImit:DISPlay[:STATe] |
| | LineDisplay |
| Limit Test Failed | CALCulate:MEASure:LImit:FAIL? |
| | meas.LimitTestFailed |
| Begin Response | CALCulate:MEASure:LImit:SEGMent1:AMPLitude:STARt |
| | limtseg.BeginResponse |
| End Response | CALCulate:MEASure:LImit:SEGMent1:AMPLitude:STOP |
| | limtseg.EndResponse |
| Begin Stimulus | CALCulate:MEASure:LImit:SEGMent:STIMulus:STARt |
| | limtseg.BeginStimulus |
| End Stimulus | CALCulate:MEASure:LImit:SEGMent:STIMulus:STOP |
| | limtseg.EndStimulus |
| Limit Line Type (Max|Min) | CALCulate:MEASure:LImit:SEGMent:TYPE |
| | limts.Type |
| Fail Sound ON/OFF | CALCulate:MEASure:LImit:SOUNd[:STATe] |
| | SoundOnFail |
| Testing ON/OFF | CALCulate:MEASure:LImit[:STATe] |
| | Trans.State |

**Bandwidth Tests**

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Set bandwidth threshold value of bandwidth test</td>
<td>CALCulate:MEASure:BLIMit:BWIDth:THReshold</td>
</tr>
<tr>
<td>Turn ON/OFF the bandwidth value display of the bandwidth test</td>
<td>CALCulate:MEASure:BLIMit:BWIDth:DISPlay:MARker:STATe</td>
</tr>
<tr>
<td>Get the bandwidth limit test results</td>
<td>CALCulate:MEASure:BLIMit:FAIL</td>
</tr>
<tr>
<td>Set/get the upper limit value of the bandwidth test</td>
<td>CALCulate:MEASure:BLIMit:MAXimum</td>
</tr>
<tr>
<td>Set/get the lower limit value of the bandwidth test</td>
<td>CALCulate:MEASure:BLIMit:MINimum</td>
</tr>
<tr>
<td>Get the bandwidth value of the bandwidth test</td>
<td>CALCulate:MEASure:BLIMit:REPort:DATA</td>
</tr>
<tr>
<td>Turn ON/OFF the bandwidth test function</td>
<td>CALCulate:MEASure:BLIMit:STATe</td>
</tr>
</tbody>
</table>

### Conversions

| Set/get parameter conversion function | CALCulate:MEASure:CONVersion:FUN Ctiohn | None |

### Electrical Delay

| Delay in distance | CALCulate:MEASure:CORRection:EDELay:DISTance | ElecDistanceDelay |
| Media | CALCulate:MEASure:CORRection:EDELay:MEDium | PortMedium |
| Electrical Delay | CALCulate:MEASure:CORRection:EDELay:TIME | meas.ElectricalDelay |
| Set units for distance | CALCulate:MEASure:CORRection:EDELay:UNIT | ElecDistanceDelayUnit |

### Wavegd Cutoff

| CALCulate:MEASure:CORRection:EDELay:WGCutoff | PortWGCutoffFreq |

### Save/Recall

| Set/get formatted measurement data | CALCulate:MEASure:DATA:FDATa | getData |
| Set/get formatted memory data | CALCulate:MEASure:DATA:FMEMory | getData |
| Set/get complex measurement data | CALCulate:MEASure:DATA:SDATa | getData |
| Set/get complex memory data | CALCulate:MEASure:DATA:SMEMory | getData |
| Reads SnP data from the selected measurement | CALCulate:MEASure:DATA:SNP? | GetSnPData |
| Reads SNP data for the specified ports | CALCulate:MEASure:DATA:SNP:PORTs? | GetSnPDataWithSpecifiedPorts |
| Saves SNP data for the specified ports | CALCulate:MEASure:DATA:SNP:PORTs:SAVE | WriteSnPDataWithSpecifiedPorts |
| Recall Files | MMEMory:LOAD | app.recall |
| Save Instrument States (*.csa, *.cst, *.sta, *.cal) and type of file | MMEMory:STORe | app.Save |

### Measurements
<table>
<thead>
<tr>
<th>Create Measurement</th>
<th>CALCulate:MEASure::DEFine</th>
<th>app.CreateMeasurement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete a measurement</td>
<td>CALCulate:MEASure::DELeTe</td>
<td>Measurements.Remove</td>
</tr>
<tr>
<td>Delete ALL measurements</td>
<td>CALCulate:MEASure::DELeTe:ALL</td>
<td>None</td>
</tr>
<tr>
<td>Deletes the trace associated with the specified measurement number</td>
<td>DISPlay:MEASure:DELeTe</td>
<td>None</td>
</tr>
<tr>
<td>Create a new trace in the specified window</td>
<td>DISPlay:MEASure:FEED</td>
<td>None</td>
</tr>
<tr>
<td>Turn the memory trace ON or OFF for the specified measurement</td>
<td>DISPlay:MEASure:MEMory[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Move a trace associated with measurement number to the specified window</td>
<td>DISPlay:MEASure:MOVE</td>
<td>None</td>
</tr>
<tr>
<td>Activate the specified measurement to be selected</td>
<td>DISPlay:MEASure:SELeCT</td>
<td>None</td>
</tr>
<tr>
<td>Turn trace display associated with the specified measurement ON or OFF</td>
<td>DISPlay:MEASure[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Set or return the title for the specified measurement</td>
<td>DISPlay:MEASure:TITLe:DATA</td>
<td>None</td>
</tr>
<tr>
<td>Turn the measurement title ON or OFF</td>
<td>DISPlay:MEASure:TITLe[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Autoscale the specified trace in the specified measurement</td>
<td>DISPlay:MEASure:Y[:SCALe]:AUTO</td>
<td>None</td>
</tr>
<tr>
<td>Set the Y axis Scale Per Division value of the specified trace associated with the specified</td>
<td>DISPlay:MEASure:Y[:SCALe]:PDIvision</td>
<td>None</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td>Command Line</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td><strong>Set the Y axis Reference Level of the specified trace associated with the specified measurement</strong></td>
<td>DISPlay:MEASure:Y[:SCALe]:RLEVel</td>
<td>None</td>
</tr>
<tr>
<td><strong>Set the Reference Position of the specified trace associated with the specified measurement</strong></td>
<td>DISPlay:MEASure:Y[:SCALe]:RPOSITION</td>
<td>None</td>
</tr>
<tr>
<td><strong>Return active sheet number</strong></td>
<td>SYSTem:ACTive:SHEet</td>
<td>None</td>
</tr>
<tr>
<td><strong>Read Active Measurement Number</strong></td>
<td>SYSTem:ACTive:MEASurement:NUMBer?</td>
<td>meas.Number</td>
</tr>
</tbody>
</table>

**Equation Editor**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command Line</th>
<th>Argument</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Turn ON / OFF equation</strong></td>
<td>CALCulate:MEASure:EQUation[:STATe]</td>
<td>State</td>
<td></td>
</tr>
<tr>
<td><strong>Set equation</strong></td>
<td>CALCulate:MEASure:EQUation:TEXT</td>
<td>Text</td>
<td></td>
</tr>
<tr>
<td><strong>Return validity of equation</strong></td>
<td>CALCulate:MEASure:EQUation:VALid?</td>
<td>Valid</td>
<td></td>
</tr>
</tbody>
</table>

**Gating**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command Line</th>
<th>Argument</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Set gate coupling parameters</strong></td>
<td>CALCulate:MEASure:FILTer[:GATE]:COUPle:PARameters</td>
<td>gate.CoupledParameters</td>
<td></td>
</tr>
<tr>
<td>Center</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:CENTer</td>
<td>gate.Center</td>
<td></td>
</tr>
<tr>
<td>Shape</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:SHApe</td>
<td>gate.Shape</td>
<td></td>
</tr>
<tr>
<td>Span</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:SPAN</td>
<td>gate.Span</td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STARTt</td>
<td>gate.Start</td>
<td></td>
</tr>
<tr>
<td>ON/OFF</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STATe</td>
<td>gate.State</td>
<td></td>
</tr>
<tr>
<td>Stop</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STOP</td>
<td>gate.Stop</td>
<td></td>
</tr>
<tr>
<td>Type (BandPass, Notch)</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME[:TYPE]</td>
<td>gate.Type</td>
<td></td>
</tr>
</tbody>
</table>

**Format**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Command Line</th>
<th>Argument</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Set or return the measurement format</strong></td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td><strong>Set or return the units for the specified data format</strong></td>
<td>CALCulate:MEASure:FORMat:UNIT</td>
<td>FormatUnit</td>
<td></td>
</tr>
<tr>
<td><strong>Group Delay Aperture Frequency</strong></td>
<td>CALCulate:MEASure:GDELay:FREQuency</td>
<td>Frequency</td>
<td></td>
</tr>
<tr>
<td>Group Delay Aperture Percent of Span</td>
<td>CALCulate:MEASure:GDELay:PERCent</td>
<td>Percent</td>
<td></td>
</tr>
<tr>
<td>Group Delay Aperture Points</td>
<td>CALCulate:MEASure:GDELay:POINts</td>
<td>Points</td>
<td></td>
</tr>
</tbody>
</table>

**Statistics**

| Get trace statistic data            | CALCulate:MEASure:FUNCtion:DATA? |
|                                    | meas.GetFilterStatistics         |
|                                    | meas.StandardDeviation           |
|                                    | meas.Mean                       |
|                                    | meas.PeakToPeak                  |

|                                    | meas.StatisticsRange             |

| Domain Range Start                  | CALCulate:MEASure:FUNCtion:DOMain:USER:STARt |
|                                    | UserRangeMin                     |

|                                    | UserRangeMax                      |

| Executes the statistical analysis   | CALCulate:MEASure:FUNCtion:EXECute    |
|                                    | None                                |

| Statistics ON|OFF            | CALCulate:MEASure:FUNCtion:STATistics[:STATe] |
|              |                | meas.ShowStatistics                 |

| Set Type (Pk-Pk|StdDev|Mean) | CALCulate:MEASure:FUNCtion:TYPE     |
|            |                | Set individually                     |

**Gain Compression**

| Read all GCA data                   | CALCulate:MEASure:GCDdata:DATA?   |
|                                    | GetRaw2DData                      |

| Read imaginary GCA data             | CALCulate:MEASure:GCDdata:IMAG?   |
|                                    | GetDataIm                          |

| Read number of iterations           | CALCulate:MEASure:GCDdata:ITERations? |
|                                    | TotalIterations                    |

| Read real GCA data                  | CALCulate:MEASure:GCDdata:REAL?   |
|                                    | GetDataRe                          |

| Set CW frequency                    | CALCulate:MEASure:GCMeas:ANALysis:CWFrequency |
|                                    | AnalysisCWFreq                     |

| Set to discrete or interpolated CW frequencies | CALCulate:MEASure:GCMeas:ANALysis:DISCrete |
|                                               | AnalysisIsDiscreteFreq            |

| Enable a compression analysis trace    | CALCulate:MEASure:GCMeas:ANALysis:ENABle |
|                                       | AnalysisEnable                     |

| Sets X-axis display                   | CALCulate:MEASure:GCMeas:ANALysis:XAXis |
|                                       | AnalysisXAxis                       |

| Interpolation                         | SENSE:GCSetup:COMPression:INTerpolate[:STATe] |
|                                       | CompressionInterpolation            |

<p>| Set and read the DC readings at the compression point in the last iteration of a smart sweep | SENSE:GCSetup:SMARt:CDC |
|                                                                                             | ReadDCAtCompression |</p>
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset currently-stored data points to the live data trace</td>
<td>CALCulate:MEASure:HOLD:CLEar</td>
<td>TraceHoldClear</td>
</tr>
<tr>
<td>Set type of trace hold to perform</td>
<td>CALCulate:MEASure:HOLD:TYPE</td>
<td>TraceHoldType</td>
</tr>
<tr>
<td>Measure</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
</tr>
<tr>
<td>Set/get the number of traces of selected channel</td>
<td>CALCulate:PARameter:COUNT</td>
<td>None</td>
</tr>
</tbody>
</table>

**Marker Functions**

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete All Markers</td>
<td>CALCulate:MEASure:MARKer:AOFF</td>
<td>meas.DeleteAllMarkers</td>
</tr>
<tr>
<td>Read/Set Data Point number</td>
<td>CALCulate:MEASure:MARKer:BUCKet</td>
<td>mark.BucketNumber</td>
</tr>
<tr>
<td>Read bandwidth data</td>
<td>CALCulate:MEASure:MARKer:BWIDth:DATA?</td>
<td>meas.BW, meas.BWCF, meas.BWLoss, meas.BWQ</td>
</tr>
<tr>
<td>Set bandwidth marker function reference to either MARKer or PEAK</td>
<td>CALCulate:MEASure:MARKer:BWIDth:REF</td>
<td>None</td>
</tr>
<tr>
<td>Bandwidth (Target)</td>
<td>CALCulate:MEASure:MARKer:BWIDth:THReshold</td>
<td>meas.BandwidthTarget</td>
</tr>
<tr>
<td>Coupled Markers Method</td>
<td>CALCulate:MEASure:MARKer:COUPling:METHod</td>
<td>CoupledMarkersMethod</td>
</tr>
<tr>
<td>Delta Marker</td>
<td>CALCulate:MEASure:MARKer:DELTa</td>
<td>mark.DeltaMarker</td>
</tr>
<tr>
<td>Interpolate Individ. Marker (Discrete)</td>
<td>CALCulate:MEASure:MARKer:DISCrete</td>
<td>mark.Interpolated</td>
</tr>
<tr>
<td>Set and return marker distance value</td>
<td>CALCulate:MEASure:MARKer:DISTance</td>
<td>mark.Distance</td>
</tr>
<tr>
<td>Format All Markers</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>mark.Format</td>
</tr>
<tr>
<td>Compression Marker level found</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:COMPres:LEVEL</td>
<td>CompressionLevel</td>
</tr>
<tr>
<td>Read Compression Marker Input power</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:COMPres:PIN</td>
<td>CompressionPin</td>
</tr>
<tr>
<td>Read Compression Marker Output power</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:COMPres:POUT</td>
<td>CompressionPout</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Default Value</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Turn ON/OFF the compression state</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:COMPression[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Execute Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:EXECute</td>
<td>SearchFunction</td>
</tr>
<tr>
<td>Multi Peak Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:MULTi:EXECute</td>
<td>None</td>
</tr>
<tr>
<td>Peak Excursion</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:MULTi:PEAK:EXCursion</td>
<td>None</td>
</tr>
<tr>
<td>Peak Polarity</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:MULTi:PEAK:POLarity</td>
<td>None</td>
</tr>
<tr>
<td>Peak Threshold</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:MULTi:PEAK:THReshold</td>
<td>None</td>
</tr>
<tr>
<td>Set or return search type of the multi search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:MULTi:SESelect</td>
<td>None</td>
</tr>
<tr>
<td>Transition</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:MULTi:TARGet:TRANSition</td>
<td>None</td>
</tr>
<tr>
<td>Target Value</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:MULTi:TARGet[:VALue]</td>
<td>None</td>
</tr>
<tr>
<td>Turn ON/OFF search tracking</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:MULTi:TRACking</td>
<td>None</td>
</tr>
<tr>
<td>Excursion Value</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:PEAK:EXCursion</td>
<td>mark. PeakExcursion</td>
</tr>
<tr>
<td>Set or return polarity of the peak search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:PEAK:POLarity</td>
<td>None</td>
</tr>
<tr>
<td>Threshold Value</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:PEAK:THReshold</td>
<td>mark. PeakThreshold</td>
</tr>
<tr>
<td>Select Search Function</td>
<td>CALCulate:MEASure:MARKer:FUNCtion[:SESelect]</td>
<td>None</td>
</tr>
<tr>
<td>Select transition type</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TARGet:TRANSition</td>
<td>None</td>
</tr>
<tr>
<td>Target (Value)</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TARGet[:VALue]</td>
<td>mark. TargetValue</td>
</tr>
<tr>
<td>Tracking</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TRACKing</td>
<td>mark. Tracking</td>
</tr>
<tr>
<td>Read notch search result</td>
<td>CALCulate:MEASure:MARKer:NOTCh:DATA?</td>
<td>None</td>
</tr>
<tr>
<td>Set bandwidth marker function reference to either MARKer or PEAK</td>
<td>CALCulate:MEASure:MARKer:NOTCh:REF</td>
<td>None</td>
</tr>
<tr>
<td>Notch Search</td>
<td>CALCulate:MEASure:MARKer:NOTCh[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Notch Level</td>
<td>CALCulate:MEASure:MARKer:NOTCh:THReshold</td>
<td>None</td>
</tr>
<tr>
<td>PNOP backoff</td>
<td>CALCulate:MEASure:MARKer:PNOP:BACKoff</td>
<td>BackOff</td>
</tr>
<tr>
<td>Read PBO In</td>
<td>CALCulate:MEASure:MARKer:PNOP:BACKoff:PIN?</td>
<td>BackOffPin</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td>Result</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>Read Pnop Comp</td>
<td>CALCulate:MEASure:MARKer:PNOP:COMPression?</td>
<td>Compression</td>
</tr>
<tr>
<td>Read Comp Max</td>
<td>CALCulate:MEASure:MARKer:PNOP:COMPression:MAXimum?</td>
<td>CompressionMax</td>
</tr>
<tr>
<td>Read Pnop Gain</td>
<td>CALCulate:MEASure:MARKer:PNOP:GAIN?</td>
<td>Gain</td>
</tr>
<tr>
<td>Read Pnop in</td>
<td>CALCulate:MEASure:MARKer:PNOP:PIN?</td>
<td>Pin</td>
</tr>
<tr>
<td>Read PMax In</td>
<td>CALCulate:MEASure:MARKer:PNOP:PIN:MAXimum?</td>
<td>PMaxIn</td>
</tr>
<tr>
<td>PNOP Power Offset</td>
<td>CALCulate:MEASure:MARKer:PNOP:POFFset</td>
<td>PinOffset</td>
</tr>
<tr>
<td>Read Pnop Out</td>
<td>CALCulate:MEASure:MARKer:PNOP:POUT?</td>
<td>POut</td>
</tr>
<tr>
<td>Read PMax Out</td>
<td>CALCulate:MEASure:MARKer:PNOP:POUT:MAXimum?</td>
<td>PMaxOut</td>
</tr>
<tr>
<td>Turn ON</td>
<td>OFF PNOP marker search</td>
<td>CALCulate:MEASure:MARKer:PNOP[:STATe]</td>
</tr>
<tr>
<td>Set and read PSAT backoff</td>
<td>CALCulate:MEASure:MARKer:PSATuration:BACKoff</td>
<td>PMaxBackOff</td>
</tr>
<tr>
<td>Read Comp Max</td>
<td>CALCulate:MEASure:MARKer:PSATuration:COMPression:MAXimum?</td>
<td>CompressionMax</td>
</tr>
<tr>
<td>Read Comp Sat</td>
<td>CALCulate:MEASure:MARKer:PSATuration:COMPression:SATuration?</td>
<td>CompressionSaturation</td>
</tr>
<tr>
<td>Read Gain Sat</td>
<td>CALCulate:MEASure:MARKer:PSATuration:GAIN?</td>
<td>GainSaturation</td>
</tr>
<tr>
<td>Read Gain Linear</td>
<td>CALCulate:MEASure:MARKer:PSATuration:GAIN:LINEar?</td>
<td>GainLinear</td>
</tr>
<tr>
<td>Read Gain Max</td>
<td>CALCulate:MEASure:MARKer:PSATuration:GAIN:MAXimum?</td>
<td>GainMax</td>
</tr>
<tr>
<td>Read PSat In</td>
<td>CALCulate:MEASure:MARKer:PSATuration:PIN?</td>
<td>Pin</td>
</tr>
<tr>
<td>Read PMax In</td>
<td>CALCulate:MEASure:MARKer:PSATuration:PIN:MAXimum?</td>
<td>PMaxIn</td>
</tr>
<tr>
<td>Read PSat Out</td>
<td>CALCulate:MEASure:MARKer:PSATuration:POUT?</td>
<td>POut</td>
</tr>
<tr>
<td>Read PMax Out</td>
<td>CALCulate:MEASure:MARKer:PSATuration:POUT:MAXimum?</td>
<td>PMaxOut</td>
</tr>
<tr>
<td>Turn ON</td>
<td>OFF PSAT marker search</td>
<td>CALCulate:MEASure:MARKer:PSATuration[:STATe]</td>
</tr>
<tr>
<td>Marker=&gt; SA</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>toSA</td>
</tr>
<tr>
<td>Marker=&gt; Span</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Marker=&gt; Center (Freq)</td>
<td></td>
<td>mark.SetCenter</td>
</tr>
<tr>
<td>Marker=&gt; CW Freq and change sweep type</td>
<td></td>
<td>mark.SetCW</td>
</tr>
<tr>
<td>Marker=&gt; Start (Freq)</td>
<td></td>
<td>mark.SetStart</td>
</tr>
<tr>
<td>Marker=&gt; Stop (Freq)</td>
<td></td>
<td>mark.SetStop</td>
</tr>
<tr>
<td>Function</td>
<td>Command</td>
<td>Parameter</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td><strong>Marker</strong></td>
<td><strong>Elect. Delay</strong></td>
<td>mark.SetElectricalDelay</td>
</tr>
<tr>
<td><strong>Marker</strong></td>
<td><strong>Ref. Level</strong></td>
<td>mark.SetReferenceLevel</td>
</tr>
<tr>
<td><strong>Marker</strong></td>
<td><strong>CW Freq - No sweep type change</strong></td>
<td>SetCWFreq</td>
</tr>
<tr>
<td><strong>ON/OFF</strong></td>
<td><strong>CALCulate:MEASure:MARKer[:STATe]</strong></td>
<td>meas.MarkerState</td>
</tr>
<tr>
<td>**Type (Normal</td>
<td>Fixed)**</td>
<td><strong>CALCulate:MEASure:MARKer:TYPE</strong></td>
</tr>
<tr>
<td><strong>Read/Set X-axis value</strong></td>
<td><strong>CALCulate:MEASure:MARKer:X</strong></td>
<td>mark.Stimulus</td>
</tr>
<tr>
<td><strong>Read/Set Y-axis value</strong></td>
<td><strong>CALCulate:MEASure:MARKer:Y</strong></td>
<td>mark.Value</td>
</tr>
<tr>
<td><strong>Read Band Power</strong></td>
<td><strong>CALCulate:MEASure:SA:MARKer:BPOWer:DATA?</strong></td>
<td>BandpowerData</td>
</tr>
<tr>
<td><strong>Read/Set Band Power Span</strong></td>
<td><strong>CALCulate:MEASure:SA:MARKer:BPOWer:SPAN</strong></td>
<td>BandpowerSpan</td>
</tr>
<tr>
<td><strong>Read/Set Band Power State</strong></td>
<td><strong>CALCulate:MEASure:SA:MARKer:BPOWer:STATe</strong></td>
<td>BandpowerState</td>
</tr>
<tr>
<td><strong>Math</strong></td>
<td><strong>CALCulate:MEASure:MATH:FUNCtion</strong></td>
<td>TraceMath</td>
</tr>
<tr>
<td><strong>Data =&gt;Memory</strong></td>
<td><strong>CALCulate:MEASure:MATH:MEMorize</strong></td>
<td>meas.DataToMemory</td>
</tr>
<tr>
<td><strong>Mixer</strong></td>
<td><strong>CALCulate:MEASure:MIXer:XAXis</strong></td>
<td>ActiveXAxisRange</td>
</tr>
<tr>
<td><strong>Constants</strong></td>
<td><strong>CALCulate:MEASure:OFFSet:MAGNitude</strong></td>
<td>MagnitudeOffset</td>
</tr>
<tr>
<td><strong>Mag Slope</strong></td>
<td><strong>CALCulate:MEASure:OFFSet:MAGNitude:SOPe</strong></td>
<td>MagnitudeSlopeOffset</td>
</tr>
<tr>
<td><strong>Phase Offset</strong></td>
<td><strong>CALCulate:MEASure:OFFSet:PHASe</strong></td>
<td>PhaseOffset</td>
</tr>
<tr>
<td><strong>Ripple Tests</strong></td>
<td><strong>CALCulate:MEASure:RLIMit:DATA</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Turn ON/OFF the ripple limit line display</strong></td>
<td><strong>CALCulate:MEASure:RLIMit:DISPlay::LINE:STATe</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Set/get the ripple limit band</strong></td>
<td><strong>CALCulate:MEASure:RLIMit:DISPlay:SElect</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Set/get the display type of ripple value</strong></td>
<td><strong>CALCulate:MEASure:RLIMit:DISPlay:TYPE</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Read the ripple test result</strong></td>
<td><strong>CALCulate:MEASure:RLIMit:FAIL</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Read the ripple value</strong></td>
<td><strong>CALCulate:MEASure:RLIMit:REPort:DATA</strong></td>
<td>None</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Turn ON/OFF the ripple test function</td>
<td>CALCulate:MEASure:RLIMit:STATe</td>
<td>None</td>
</tr>
<tr>
<td><strong>Smoothing</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smoothing Percent</td>
<td>CALCulate:MEASure:SMOothing:APERture</td>
<td>meas.SmoothingAperture</td>
</tr>
<tr>
<td>Smoothing Points</td>
<td>CALCulate:MEASure:SMOothing:POINts</td>
<td>None</td>
</tr>
<tr>
<td>Smoothing ON/OFF</td>
<td>CALCulate:MEASure:SMOothing[:STATe]</td>
<td>meas.Smoothing</td>
</tr>
<tr>
<td><strong>Transform</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets the alignment of the time domain measurement.</td>
<td>CALCulate:MEASure:TRANsform:TIME:ALIGNment</td>
<td>Alignment</td>
</tr>
<tr>
<td>Center</td>
<td>CALCulate:MEASure:TRANsform:TIME:CENTer</td>
<td>trans.Center</td>
</tr>
<tr>
<td>Specify measurement type for distance markers</td>
<td>CALCulate:MEASure:TRANsform:TIME:MARKer:MODE</td>
<td>trans.DistanceMarkerMode</td>
</tr>
<tr>
<td>Specify units for distance markers</td>
<td>CALCulate:MEASure:TRANsform:TIME:MARKer:UNIT</td>
<td>trans.DistanceMarkerUnit</td>
</tr>
<tr>
<td><strong>Uncertainty</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trace</td>
<td>CALCulate:MEASure:UNCertainty:DISPlay:CFACtor</td>
<td>CoverageFactor</td>
</tr>
<tr>
<td>Trace Type</td>
<td>CALCulate:MEASure:UNCertainty:DISPlay:TYPE</td>
<td>DisplayType</td>
</tr>
<tr>
<td>Repeatability</td>
<td>CALCulate:MEASure:UNCertainty:MODE:CABLE:REPeat</td>
<td>CableRepeatabilityUncertain</td>
</tr>
<tr>
<td>Calibration</td>
<td>CALCulate:MEASure:UNCertainty:MODE:ETERm</td>
<td>ErrorTermUncertainty</td>
</tr>
<tr>
<td>Save uncertainty data</td>
<td>CALCulate:MEASure:UNCertainty:SAVE</td>
<td>WriteUncertaintyFile</td>
</tr>
<tr>
<td><strong>X-Axis</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set/get X-axis for trace</td>
<td>CALCulate:MEASure:X:AXIS</td>
<td>None</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Set/get the X-Axis domain</td>
<td>CALCulate:MEASure:X:AXIS:DOMain</td>
<td>None</td>
</tr>
<tr>
<td>Get X-Axis values (variant)</td>
<td>CALCulate:MEASure:X:VALues?</td>
<td>Get X-axis Values</td>
</tr>
<tr>
<td><strong>Display Setup</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sheet Layout</td>
<td>DISPlay:SHet:ARRange</td>
<td>None</td>
</tr>
<tr>
<td>Get list of window numbers which the sheet contains</td>
<td>DISPlay:SHet:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Add/Delete Sheet</td>
<td>DISPlay:SHet:STATe</td>
<td>None</td>
</tr>
<tr>
<td>Sheet Title</td>
<td>DISPlay:SHet:TITLE:DATA</td>
<td>None</td>
</tr>
<tr>
<td>Creates N windows</td>
<td>DISPlay:SPLit</td>
<td>None</td>
</tr>
<tr>
<td>Toolbar Cal Set Viewer</td>
<td>DISPlay:TOOЛbar:CSET[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Display Update</td>
<td>DISPlay:UPDate:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>Enable/disable display updates</td>
<td>DISPlay:UPDate[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Sets the marker readouts to coupled (one combination annotation) or not</td>
<td>DISPlay:WINDow:ANNotation:MARKer:COUPle</td>
<td>None</td>
</tr>
<tr>
<td>coupled (one annotation per trace).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shows the marker readouts only for active trace or for all traces.</td>
<td>DISPlay:WINDow:ANNotation:MARKer:VISible</td>
<td>None</td>
</tr>
<tr>
<td>Y-axis Labels</td>
<td>DISPlay:WINDow:ANNotation:Y[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Feed specified window to a sheet</td>
<td>DISPlay:WINDow:FEED</td>
<td>None</td>
</tr>
<tr>
<td>Feed specified measurement to specified window</td>
<td>DISPlay:WINDow:TRACE:FEED:MNUMber</td>
<td>None</td>
</tr>
<tr>
<td>Returns the next unused trace number</td>
<td>DISPlay:WINDow:TRACE:NEXT[:NUMBer]</td>
<td>None</td>
</tr>
<tr>
<td>Set graph divisions</td>
<td>DISPlay:WINDow:Y[:SCALE]:DIVisions</td>
<td>None</td>
</tr>
<tr>
<td><strong>Pulse</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set pulse start time</td>
<td>SENSE:SWEep:PULSe:PROFile:STARt</td>
<td>PulseProfileStart</td>
</tr>
<tr>
<td>Set pulse stop time</td>
<td>SENSE:SWEep:PULSe:PROFile:STOP</td>
<td>PulseProfileStop</td>
</tr>
<tr>
<td>Enable pulse4 to monitor ADC activity</td>
<td>SENSE:PULSe4:OPTion</td>
<td>None</td>
</tr>
<tr>
<td><strong>Sweep</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>None</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>IF Bandwidth resolution</td>
<td>SENSE:SEGMENT:BWIDth:PORT[RESolution]</td>
<td></td>
</tr>
<tr>
<td>IF Bandwidth Per Port</td>
<td>SENSE:SEGMENT:SWEep:DElay:CONTrol</td>
<td></td>
</tr>
<tr>
<td>Sweep Delay Time</td>
<td>SENSE:SEGMENT:SWEep:DElay</td>
<td></td>
</tr>
<tr>
<td>Sweep delay ON/OFF</td>
<td>SENSE:SEGMENT:SWEep:DElay:CONTrol</td>
<td></td>
</tr>
<tr>
<td>Sweep Dwell</td>
<td>SENSE:SEGMENT:SWEep:DWEL1</td>
<td></td>
</tr>
<tr>
<td>Sweep dwell ON/OFF</td>
<td>SENSE:SEGMENT:SWEep:DWEL1:CONTrol</td>
<td></td>
</tr>
<tr>
<td>Sweep Mode</td>
<td>SENSE:SEGMENT:SWEep:GENeration</td>
<td></td>
</tr>
<tr>
<td>Sweep mode ON/OFF</td>
<td>SENSE:SEGMENT:SWEep:GENeration:CONTrol</td>
<td></td>
</tr>
<tr>
<td>Total Sweep Points</td>
<td>SENSE:SEGMENT:SWEep:POINts:TOTal?</td>
<td></td>
</tr>
<tr>
<td>Total Sweep Time</td>
<td>SENSE:SEGMENT:SWEep:TIME:TOTal?</td>
<td></td>
</tr>
<tr>
<td>Set shift LO maximum frequency</td>
<td>SENSE:SWEep:SLOCAL:MAXimum</td>
<td></td>
</tr>
<tr>
<td>Turn shift LO on or off</td>
<td>SENSE:SWEep:SLOCAL:STATE</td>
<td></td>
</tr>
</tbody>
</table>

**Power**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receiver Reference Attenuation</td>
<td>SOURce:POWER:ATTenuation:RECeiver:REFERENCE</td>
<td></td>
</tr>
<tr>
<td>Receiver Test Attenuation</td>
<td>SOURce:POWER:ATTenuation:RECeiver:TEST</td>
<td></td>
</tr>
</tbody>
</table>

**System Settings**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Sound</td>
<td>SYSTem:BEEPer:STATE</td>
<td></td>
</tr>
<tr>
<td>Set limit test warning sound</td>
<td>SYSTem:BEEPer:WARNING:IMMediate</td>
<td></td>
</tr>
<tr>
<td>Return the maximum trigger delay</td>
<td>SYSTem:CAPability:DELay:TRIGger:MAX?</td>
<td></td>
</tr>
<tr>
<td>Return the minimum trigger delay</td>
<td>SYSTem:CAPability:DELay:TRIGger:MIN?</td>
<td></td>
</tr>
<tr>
<td>Return the maximum IF frequency</td>
<td>SYSTem:CAPability:HARDware:IF:MAXimum?</td>
<td></td>
</tr>
<tr>
<td>Return the minimum IF frequency</td>
<td>SYSTem:CAPability:HARDware:IF:MINimum?</td>
<td></td>
</tr>
<tr>
<td>Set or return list of discrete frequencies</td>
<td>SYSTem:CAPability:HARDware:POWer:DISCrete:FREQuency:LIST</td>
<td>DiscreteFrequencies</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>----------------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Return a single minimum leveled power value</td>
<td>SYSTem:CAPability:HARDware:POWer:DISCrete:MINimum?</td>
<td>DiscreteGetMinPower</td>
</tr>
<tr>
<td>Returns an array with the names of all RF path elements that may be configured</td>
<td>SYSTem:CAPability:HARDware:POWer:PATH:CONFig:ELEMent:CATalogue?</td>
<td>PathElements</td>
</tr>
<tr>
<td>Return the name of the value for the given path element name or sets the value of a path element</td>
<td>SYSTem:CAPability:HARDware:POWer:PATH:CONFig:ELEMent[:STATe]</td>
<td>PathElement</td>
</tr>
<tr>
<td>Return all valid values for the given path configuration element</td>
<td>SYSTem:CAPability:HARDware:POWer:PATH:CONFig:ELEMent:VALue:CATalogue?</td>
<td>None</td>
</tr>
<tr>
<td>Set or return the port number for power data</td>
<td>SYSTem:CAPability:HARDware:POWer:PORT</td>
<td>PortNumber</td>
</tr>
<tr>
<td>Set or return the lower bound of the frequency range</td>
<td>SYSTem:CAPability:HARDware:POWer:RANGe:FREQuency:STARt</td>
<td>RangeStartFrequency</td>
</tr>
<tr>
<td>Set or return the upper bound of the frequency range</td>
<td>SYSTem:CAPability:HARDware:POWer:RANGe:FREQuency:STOP</td>
<td>RangeStopFrequency</td>
</tr>
<tr>
<td>Return the minimum of all max leveled power values</td>
<td>SYSTem:CAPability:HARDware:POWer:RANGe:MAXimum?</td>
<td>RangeGetMaxPower</td>
</tr>
<tr>
<td>Return the maximum of all min power values</td>
<td>SYSTem:CAPability:HARDware:POWer:RANGe:MINimum?</td>
<td>RangeGetMinPower</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Reset all Power Range properties to default values</td>
<td>SYSTem:CAPability:HARDware:POWer:RESet</td>
<td>Reset (Power Range)</td>
</tr>
<tr>
<td>Set and return the type of power range data to be returned</td>
<td>SYSTem:CAPability:HARDware:POWer:TYPE</td>
<td>PowerRangeType</td>
</tr>
<tr>
<td>Return the maximum IFBW for the standard IF filter</td>
<td>SYSTem:CAPability:IFBW:MAXimum?</td>
<td>None</td>
</tr>
<tr>
<td>Return the minimum IFBW for the standard IF filter</td>
<td>SYSTem:CAPability:IFBW:MINimum?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the list of licenses</td>
<td>SYSTem:CAPability:LICenses:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Set and return the group of channels</td>
<td>SYSTem:CHANnels:COUPle:GROup</td>
<td>None</td>
</tr>
<tr>
<td>Set and return the Multi DUT parallel measurement state</td>
<td>SYSTem:CHANnels:COUPle:PARallel[:ENABle]</td>
<td>None</td>
</tr>
<tr>
<td>Get the information if the parallel measurement is executed in the last sweep</td>
<td>SYSTem:CHANnels:COUPle:PARallel:STATe</td>
<td>None</td>
</tr>
<tr>
<td>Set up multiple channels for manual trigger</td>
<td>SYSTem:CHANnels:SINGle</td>
<td>None</td>
</tr>
<tr>
<td>Return the ID string of ECals</td>
<td>SYSTem:COMMunicate:ECAL:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Return a list of characterizations</td>
<td>SYSTem:COMMunicate:ECAL:CLISt?</td>
<td>None</td>
</tr>
<tr>
<td>Return the number of installed cal kits</td>
<td>SYSTem:COMMunicate:ECAL:COUNt?</td>
<td>None</td>
</tr>
<tr>
<td>Delete user characterizations from VNA disk memory</td>
<td>SYSTem:COMMunicate:ECAL:DMEMory:CLEar</td>
<td>None</td>
</tr>
<tr>
<td>Import file into VNA disk memory</td>
<td>SYSTem:COMMunicate:ECAL:DMEMory:IMPort</td>
<td>None</td>
</tr>
<tr>
<td>Save existing ECal characterization to a file</td>
<td>SYSTem:COMMunicate:ECAL:EXPort</td>
<td>None</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Default</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Read identification and characterization information for ECal module</td>
<td>SYSTem:COMMunicate:ECAL:INFormation?</td>
<td>None</td>
</tr>
<tr>
<td>Read identification and characterization information from ECal module or VNA disk memory</td>
<td>SYSTem:COMMunicate:ECAL:KNAME:INFormation?</td>
<td>None</td>
</tr>
<tr>
<td>Return list of index numbers for ECal modules</td>
<td>SYSTem:COMMunicate:ECAL:LIST?</td>
<td>None</td>
</tr>
<tr>
<td>Return number of unique states for specified path name on selected ECal module</td>
<td>SYSTem:COMMunicate:ECAL:PATH:COUNt?</td>
<td>None</td>
</tr>
<tr>
<td>Set and return the maximum output current value of the external DC Source</td>
<td>SYSTem:CONFigure:EDEVice:DC:LIMit:CURRent</td>
<td>CurrentLimit</td>
</tr>
<tr>
<td>Set and return the maximum output voltage value of the external DC Source</td>
<td>SYSTem:CONFigure:EDEVice:DC:LIMit:VOLTage</td>
<td>VoltageLimit</td>
</tr>
<tr>
<td>Set Modulation Control</td>
<td>SYSTem:CONFigure:EDEVice:SOURce:MODulation:CONTrol</td>
<td>None</td>
</tr>
<tr>
<td>Disk Drive Version</td>
<td>SYSTem:DISK:REVision?</td>
<td>None</td>
</tr>
<tr>
<td>Set and return the coupler state</td>
<td>SYSTem:FCORrection:CHANnel:COUPler[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Set and return Source Port Control</td>
<td>SYSTem:ISPControl[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Modify the manufacturer name</td>
<td>SYSTem:PERSona:MANufacturer</td>
<td>None</td>
</tr>
<tr>
<td>Reset to original manufacturer identification</td>
<td>SYSTem:PERSona:MANufacturer:DEFault</td>
<td>None</td>
</tr>
<tr>
<td>Modify the product model</td>
<td>SYSTem:PERSona:MODEl</td>
<td>None</td>
</tr>
<tr>
<td>Reset to original product model name</td>
<td>SYSTem:PERSona:MODEl:DEFault</td>
<td>None</td>
</tr>
<tr>
<td>Shutdown or Restart System</td>
<td>SYSTem:POFF</td>
<td>None</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Use only ramp sweeps for Auto Sweep Mode</td>
<td>SYSTem:PREFERences:ITEM:ASMRamp</td>
<td>None</td>
</tr>
<tr>
<td>On Power-on show Keys toolbar</td>
<td>SYSTem:PREFERences:ITEM:Keys</td>
<td>ShowkeysToolbarAtPowerOn</td>
</tr>
<tr>
<td>Sets the bandwidth search preference</td>
<td>SYSTem:PREFERences:ITEM:MARKer:BANDwidth:SEARch</td>
<td>BandwidthSearch</td>
</tr>
<tr>
<td>Use single marker for marker search</td>
<td>SYSTem:PREFERences:ITEM:MARKer:SINGle</td>
<td>SingleMarkerSearch</td>
</tr>
<tr>
<td>Confirm preset</td>
<td>SYSTem:PREFERences:ITEM:PRESet:CONFirm</td>
<td>ConfirmPreset</td>
</tr>
<tr>
<td>Controls the on/off state of the preference, &quot;Use keyboard to navigate softkeys&quot;</td>
<td>SYSTem:PREFERences:ITEM:SOFTkeys:NAVigation</td>
<td>None</td>
</tr>
<tr>
<td>Return list of visible sheets</td>
<td>SYSTem:SHEets:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Specifies if the source power cal in the calset linked to a measurement cal should be enabled or disabled with that cal</td>
<td>None</td>
<td>PreferSourcePowerCalFromCal</td>
</tr>
</tbody>
</table>

**Spectrum Analysis**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set and read the FIR filter for 25 MHz decimation</td>
<td>SENSE:SA:ADC:SAMPle:DECimation:FIR</td>
<td>ADCEnableFIRFor25Mhz</td>
</tr>
<tr>
<td>Set and read the default values for DFT bandwidth</td>
<td>SENSE:SA:DFT:BANDwidth:AUTO</td>
<td>AutoBandwidth</td>
</tr>
<tr>
<td>Set and read the maximum value for narrow DFT bandwidth</td>
<td>SENSE:SA:DFT:BANDwidth:NARrow:MAX</td>
<td>BandwidthNarrowMax</td>
</tr>
<tr>
<td>Set and read the minimum value for narrow DFT bandwidth</td>
<td>SENSE:SA:DFT:BANDwidth:NARrow:MIN</td>
<td>BandwidthNarrowMin</td>
</tr>
<tr>
<td>Set and read the maximum value for wide DFT bandwidth</td>
<td>SENSE:SA:DFT:BANDwidth:WIDE:MAX</td>
<td>BandwidthWideMax</td>
</tr>
<tr>
<td>Set and read the minimum value for wide DFT bandwidth</td>
<td>SENSE:SA:DFT:BANDwidth:WIDE:MIN</td>
<td>BandwidthWideMin</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>-----------------------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Read the current DFT record size</td>
<td>SENSe:SA:DFT:RECORD:SIZE?</td>
<td>RecordSize</td>
</tr>
<tr>
<td>Read the current DFT resolution</td>
<td>SENSe:SA:DFT:RESolution?</td>
<td>Resolution</td>
</tr>
<tr>
<td>Set and read the DFT record size type</td>
<td>SENSe:SA:DFT:TYPE</td>
<td>Type</td>
</tr>
<tr>
<td>Set and read the image rejection strength</td>
<td>SENSe:SA:IMAGe:STRENgth</td>
<td>ImageRejectStrength</td>
</tr>
<tr>
<td>Read the current span DFT bin count</td>
<td>SENSe:SA:SPAN:BINS:COUNT?</td>
<td>SpanBinsCount</td>
</tr>
</tbody>
</table>

The following are new programming commands for release A.12.50.01.

**Memory Interpolation**
- Generate a new ENR file by embedding an adapter to an existing ENR file.
  - CSET:FIXTure:ENR:EMBed
  - ENREmbedAdapter
- Read S parameter of ECal Thru from the ECal memory and save it as s2p file
  - SYSTem:COMMunicate:ECAL:EXPort:SNP
  - OutputSNPFromECal Method

The following are new programming commands for PNA release A.10.49.05.xx See What's New.

**Memory Interpolation**
- Sets and reads the state of the memory data interpolation
  - CALC:MATH:INTerpolate
  - InterpolateMemory
- Sets and reads the state of the memory data interpolation default preference
  - SYST:PREF:ITEM:MINTerpolate
  - InterpolateMemoryIsDefault

**External DC Device**
- Sets and returns the current limit value
  - SYST:CONF:EDEV:DC:LIMit:CURRent
  - CurrentLimit
- Sets and returns the voltage limit value
  - SYST:CONF:EDEV:DC:LIMit:VOLTage
  - VoltageLimit
Error Correction Terms
Computes the error correction terms, turns Correction ON, and saves the SENS:CORR:COLL:GUID:ETER:COMPute calibration to an existing, specified Cal Set

The following are new programming commands for PNA release A.10.49 See What's New.

Dark shaded commands existed before this release. They are provided here for convenience.

Source Setup tab

The following are new programming commands for PNA release A.10.45 See What's New.

Dark shaded commands existed before this release. They are provided here for convenience.
SA Advanced Settings tab

Millimeter-Wave Commands
The following are new programming commands for PNA release A.10.40. See What's New.

Dark shaded commands existed before this release. They are provided here for convenience.
Properties

Number of Points
RBW Shape
Detector Type
Averaging Type
ADC Filter
ADC Filter Auto
Resolution Bandwidth
Resolution Bandwidth Auto
Minimum Resolution Bandwidth
Maximum Resolution Bandwidth
Video Bandwidth
Video Bandwidth Auto
Minimum Video Bandwidth
Maximum Video Bandwidth
RBW/VBW
Span/RBW
Video Averaging Count (no UI)

Advanced Settings tab
Source Setup tab

SA Marker Settings

Marker to SA  CALC:MARKer:SET SA  toSA
Read Band Power Span  CALC:SA:MARK:BPoweR:SPAN  BandpowerSpan
Dynamic Uncertainty

Setup Options

Uncertainty Options

- **Calibration Options**
  - **Noise**
  - **Repeatability**
  - **Standard Definition**

- **Max Uncertainty Points**: 500

Service Mode

[OK] [Cancel]

- **SYST:UNC:ETER:NOIS:ENAB** PortNoiseEnabled
- **SYST:UNC:ETER:CABL:REP** CableRepeatabilityEnabled
- **SYST:UNC:ETER:SDEF** StandardDefinitionsEnabled
- **SYST:UNC:POIN:MAX** MaximumUncertaintyPoints

Noise Characterization

- Clear noise data on specified port: **SYST:UNC:PORT<p>:NOIS:RESet** ResetNoise
- Clear noise data on all ports: **SYST:UNC:PORT:NOIS:RESet** ResetNoiseForAllPorts
- Copy noise from a port to all ports: **SYST:UNC:PORT:NOIS:ALL:COPY** CopyNoiseToAllPorts

Cables Characterization

- List cables: **SYST:UNC:CABL:CATalog?** Cables Collection
- Assign Cable to all ports: **SYST:UNC:PORT:CABL:ALL** SelectCableForAllPorts
- Assign Cable to specified port: **SYST:UNC:PORT<p>:CABL:e** Cable
- Reset repeatability: **SYST:UNC:CABL:REP:RES** ResetRepeatability

Uncertainty workspace

- Load workspace: **SYST:UNC:LOAD** Recall
- Save workspace: **SYST:UNC:STORe** Save

Enabling a Guided Calibration to include Uncertainties

- Checkbox on Guided Cal Select Ports page: **SENS:CORR:COLL:GUID:UNC** UncertaintyEnabled
### Trace Properties

#### Uncertainty Trace Property

<table>
<thead>
<tr>
<th>Trace Type</th>
<th>CALC:UNC:DISP:TYPE</th>
<th>DisplayType</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>CALC:UNC:DISP:CFAC</td>
<td>CoverageFactor</td>
</tr>
<tr>
<td>UMax</td>
<td>CALC:UNC:MOD:NOIS</td>
<td>MeasurementNoiseUncertainty</td>
</tr>
<tr>
<td>UMin</td>
<td>CALC:UNC:MODE:CABL:REP</td>
<td>CableRepeatabilityUncertainty</td>
</tr>
<tr>
<td>UBars</td>
<td>CALC:UNC:MODE:ETER</td>
<td>ErrorTermUncertainty</td>
</tr>
<tr>
<td>Shade</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ellipses</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Apply to all traces**: None
- **Add Trace**: None
- **Save uncertainty data**: CALC:UNC:SAVE

### Preferences

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TreatMkr10AsReference</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MarkCoupControlsMkrState</td>
<td>SYST:PREF:ITEM:MCControl</td>
<td>COM:SYST:PREF:ITEM:MCControl</td>
</tr>
</tbody>
</table>

###Capabilities commands

<table>
<thead>
<tr>
<th>Returns the maximum leveled source power.</th>
<th>SYST:CAP:ALC:POWer:MAXimum[:LEVel]?</th>
<th>MaximumSourceALCPower</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the minimum leveled source power.</td>
<td>SYST:CAP:ALC:POWer:MINimum[:LEVel]?</td>
<td>MinimumSourceALCPower</td>
</tr>
<tr>
<td>Returns the maximum possible number of channels.</td>
<td>SYST:CAP:CHANnels:MAXimum[:COUNT]?</td>
<td>MaximumNumberOfChannels</td>
</tr>
<tr>
<td>Returns whether or not the analyzer has FOM.</td>
<td>SYST:CAP:FOM:EXISts?</td>
<td>IsFrequencyOffsetPresent</td>
</tr>
<tr>
<td>Returns the maximum frequency of the analyzer, including any over-sweep</td>
<td>SYST:CAP:FREQuency:MAXimum?</td>
<td>MaximumFrequency</td>
</tr>
<tr>
<td>Returns the minimum frequency of the analyzer, including any under-sweep</td>
<td>SYST:CAP:FREQuency:MINimum?</td>
<td>MinimumFrequency</td>
</tr>
<tr>
<td>Returns whether or not there is a receiver attenuator on the specified port.</td>
<td>SYST:CAP:HARD:ATT:REC:EXISts?</td>
<td>IsReceiverStepAttenuatorPresent</td>
</tr>
<tr>
<td>Returns a list of test port names including external testset ports.</td>
<td>SYST:CAP:HARD:PORT:CAT?</td>
<td>TestPortNames</td>
</tr>
<tr>
<td>Returns the number of test ports including external testset ports.</td>
<td>SYST:CAP:HARD:PORTs:COUNt?</td>
<td>NumberOfPorts</td>
</tr>
<tr>
<td>Returns a list of internal test port names.</td>
<td>SYST:CAP:HARD:PORTs:INT:CAT?</td>
<td>InternalTestsetPortNames</td>
</tr>
<tr>
<td>Returns the number of internal test ports.</td>
<td>SYST:CAP:HARD:PORTs:INT:COUNt?</td>
<td>InternalTestsetPortCount</td>
</tr>
<tr>
<td>Returns the port number associated with the specified port name.</td>
<td>SYST:CAP:HARD:PORTs:PNUMber?</td>
<td>GetPortNumber</td>
</tr>
<tr>
<td>Returns a catalog (comma-separated string list) of internal source port names.</td>
<td>SYST:CAP:HARD:PORTs:INT:CAT?</td>
<td>InternalSourcePortNames</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Result</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>Returns the number of internal source ports.</td>
<td>SYST:CAP:HARD:PORTs:INT:COUNt?</td>
<td>InternalSourcePortCount</td>
</tr>
<tr>
<td>Returns whether or not the specified port number has a reference bypass switch.</td>
<td>SYST:CAP:HARD:RBSWitch:EXISts?</td>
<td>IsReferenceBypassSwitchPresent</td>
</tr>
<tr>
<td>Returns the number of receivers in the analyzer.</td>
<td>SYST:CAP:HARD:REC:COUNT?</td>
<td>ReceiverCount</td>
</tr>
<tr>
<td>Returns whether or not the analyzer has direct receiver access (front-panel jumpers).</td>
<td>SYST:CAP:HARD:REC:DACCess</td>
<td>HasDirectReceiverAccess</td>
</tr>
<tr>
<td>Returns the number of sources in the analyzer.</td>
<td>SYST:CAP:HARD:SOURce:COUNt?</td>
<td>SourceCount</td>
</tr>
<tr>
<td>Returns the list of supported IFBW values.</td>
<td>SYST:CAP:IFBW:CAT?</td>
<td>IFBWList</td>
</tr>
<tr>
<td>Returns the list of supported Noise BW when using a noise receiver (option 029).</td>
<td>SYST:CAP:NBW:NOISe:CAT?</td>
<td>NoiseReceiverNoiseBWList</td>
</tr>
<tr>
<td>Returns the list of supported Noise BW values when using the NA receiver for noise measurements (option 028).</td>
<td>SYST:CAP:NBW:STD:CAT?</td>
<td>StandardReceiverNoiseBWList</td>
</tr>
<tr>
<td>Returns the maximum number of points.</td>
<td>SYST:CAPability:POINts:MAXimum?</td>
<td>MaximumNumberOfPoints</td>
</tr>
<tr>
<td>Returns the minimum number of points.</td>
<td>SYST:CAPability:POINts:MINimum?</td>
<td>MinimumNumberOfPoints</td>
</tr>
<tr>
<td>Returns the maximum specified frequency of the analyzer, not including any over-sweep.</td>
<td>SYST:CAP:PRESet:FREQ:MAXimum?</td>
<td>PresetMaxFrequency</td>
</tr>
<tr>
<td>Returns the minimum specified frequency of the analyzer, not including any under-sweep.</td>
<td>SYST:CAP:PRESet:FREQ:MINimum?</td>
<td>PresetMinFrequency</td>
</tr>
<tr>
<td>Returns the list of supported Resolution BW values for the IMS channel.</td>
<td>SYST:CAP:RBW:IMS:CATalog?</td>
<td>ResBWList</td>
</tr>
<tr>
<td>Returns the list of supported Resolution BW values for the SA channel.</td>
<td>SYST:CAP:RBW:SA:CATalog?</td>
<td>SAResBWList</td>
</tr>
<tr>
<td>Returns the maximum number of windows possible.</td>
<td>SYST:CAP:WIND:MAX[:COUNt]?</td>
<td>MaximumNumberOfWindows</td>
</tr>
<tr>
<td>Returns the maximum number of traces per window.</td>
<td>SYST:CAP:WIND:TRAC:MAX[:COUNt]?</td>
<td>MaximumNumberOfTracesPerWindow</td>
</tr>
<tr>
<td><strong>Miscellaneous</strong></td>
<td><strong>SCPI</strong></td>
<td><strong>COM</strong></td>
</tr>
<tr>
<td>-------------------</td>
<td>----------</td>
<td>---------</td>
</tr>
<tr>
<td>Frequency Step</td>
<td>SENS:SWEep:STEP</td>
<td>FrequencyStep Property</td>
</tr>
<tr>
<td>Size</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Marker noise</td>
<td>CALC:MARK:FORMat NOISe</td>
<td>Format (Marker) Property</td>
</tr>
<tr>
<td>format</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display marker</td>
<td>DISP:WIND:ANN:MARK:SYMB:ABOV</td>
<td>MarkerSymbolsAboveTrace Property</td>
</tr>
<tr>
<td>symbols above the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>trace</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Limit Pass/Fail X-</td>
<td>DISP:WIND:ANN:LIIM:XPOS</td>
<td>LimitTestXPosition</td>
</tr>
<tr>
<td>axis Position</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Limit Pass/Fail Y-</td>
<td>DISP:WIND:ANN:LIIM:YPOS</td>
<td>LimitTestYPosition</td>
</tr>
<tr>
<td>axis Position</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scope of marker</td>
<td>CALC:MARK:COUP:METH</td>
<td>CoupledMarkersMethod</td>
</tr>
<tr>
<td>coupling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DIQ Load</td>
<td>SENS:DIQ:LOAD</td>
<td>Load Method</td>
</tr>
<tr>
<td>DIQ Save</td>
<td>SENS:DIQ:SAVE</td>
<td>Save Method</td>
</tr>
<tr>
<td>Trace Hold</td>
<td>CALC:HOLD:TYPE</td>
<td>TraceHoldType</td>
</tr>
<tr>
<td>Trace Hold Clear</td>
<td>CALC:HOLD:CLEar</td>
<td>TraceHoldClear</td>
</tr>
</tbody>
</table>

The following are new programming commands for PNA release A.10.25 See What's New.

**Diff IQ Setup Dialog**
### DIQ Range Settings Dialog

#### Frequency
- **Start/Stop**
  - **Start:** 1.500000000 GHz
  - **Stop:** 2.500000000 GHz
  - **IFBW:** 100.000 kHz

#### Coupling
- **Couple to:** F1
- **Offset:** F2
- **Multiplier:** 1
- **Divisor:** 1
- **Output = Frequency * Multiplier / Divisor + Offset**

### SCPI
- **SENSe:DIQ:FREQ:RANGe:STARt** RangeStartF0
- **SENSe:DIQ:FREQ:RANGe:STOP** RangeStopF0
- **SENSe:DIQ:FREQ:RANGe:IFBW** RangeIFBW Property
- **SENSe:DIQ:FREQ:RANGe:COUPle:STATe** RangeCouple State Property
- **SENSe:DIQ:FREQ:RANGe:COUPle:ID** RangeCouple Id Property
- **SENSe:DIQ:FREQ:RANGe:COUPle:OFFSet** RangeOffset Property
- **SENSe:DIQ:FREQ:RANGe:COUPle:UCONvert** RangeOffset Up Property
- **SENSe:DIQ:FREQ:RANGe:COUPle:MULTiplier** RangeMultiplier Property
- **SENSe:DIQ:FREQ:RANGe:COUPle:DIVisor** RangeDivisor Property
DIQ Source Configuration Dialog

**SCPI**

SENSe:DIQ:PORT:STA
SENSe:DIQ:PORT:RAN

**Power**

SENSe:DIQ:PORT:POV

**Phase**

SENSe:DIQ:PORT:PHA

**Match Correction**

SENSe:DIQ:PORT:MA
### DIQ Edit Parameters Dialog

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:DIQ:PARameter:DEFine</td>
<td>DefineParameter</td>
</tr>
<tr>
<td>SENSE:DIQ:PARameter:DELETE</td>
<td>DeleteParameter</td>
</tr>
<tr>
<td>SENSE:DIQ:PARameter:CATalog?</td>
<td>ParameterList</td>
</tr>
<tr>
<td>SENSE:DIQ:LOAD</td>
<td>LoadMethod</td>
</tr>
<tr>
<td>SENSE:DIQ:SAVE</td>
<td>SaveMethod</td>
</tr>
</tbody>
</table>

### DIQ Select X-Axis Dialog

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:DIQ:XAXis</td>
<td>ActiveXAxis</td>
</tr>
<tr>
<td>CALC:DIQ:XAXis:DOMain?</td>
<td>XAxisDomain</td>
</tr>
<tr>
<td>CALC:DIQ:XAXis:SOURce?</td>
<td>XAxisSource</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release A.10.20.03**  See What's New .

**Narrowband Compensation**

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable</td>
<td>SENS:NOIS:NARRowband:STATe</td>
</tr>
</tbody>
</table>

**Cal Plane Manager**

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Characterize a fixture</td>
<td>CSET:FIXTure:CHARacterize</td>
</tr>
<tr>
<td>Creates a single S2P file from two existing files.</td>
<td>CSET:FIXTure:CASCADE</td>
</tr>
</tbody>
</table>

**Note:** There are currently NO programming commands for Automatic Fixture Removal.

The following are new programming commands for **PNA release A.10.15**  See What's New .
Cal Sets

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the Cal Types from the calset.</td>
<td>None</td>
</tr>
<tr>
<td>Returns the properties of the calset.</td>
<td>None</td>
</tr>
<tr>
<td>Returns the numbers of the channels using the calset.</td>
<td>None</td>
</tr>
<tr>
<td>Unselect Cal Set</td>
<td>SENS:CORR:CSET:DEAC</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release A.10.00**  See What's New.

### Toolbars ON|OFF

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show virtual hardkeys</td>
<td>DISPlay:TOOLbar:KEYS:[STATe]</td>
</tr>
<tr>
<td>Show port extensions</td>
<td>DISPlay:TOOLbar:EXT:[STATe]</td>
</tr>
<tr>
<td>Shown Time Domain Transform</td>
<td>DISPlay:TOOLbar:TRAN:[STATe]</td>
</tr>
</tbody>
</table>

### Miscellaneous

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set and read marker bucket.</td>
<td>CALC:MARK:BUCKet</td>
</tr>
<tr>
<td>Read the directory path</td>
<td>SYST:CONF:DIR</td>
</tr>
<tr>
<td>Read supported parameters for a Meas Class</td>
<td>SYST:MCL:PAR:CAT?</td>
</tr>
<tr>
<td>System clock state</td>
<td>SYST:CLOCk</td>
</tr>
<tr>
<td>IOConfiguration</td>
<td>None</td>
</tr>
<tr>
<td>Read model number</td>
<td>None</td>
</tr>
<tr>
<td>Read serial number</td>
<td>None</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release A.09.90**  See What's New.
SMC Phase Reference Cal

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unknown Mixer calibration state</td>
<td>SYST:CAL:PHAS:UNKN:INClude</td>
<td>IncludeUnknownMixer</td>
</tr>
<tr>
<td>Unknown mixer LO frequency</td>
<td>SYST:CAL:PHAS:UNKN:LO:FREQuency</td>
<td>UnknownMixerLOFrequency</td>
</tr>
<tr>
<td>Unknown mixer LO power</td>
<td>SYST:CAL:PHAS:UNKN:LO:POWer</td>
<td>UnknownMixerLOPower</td>
</tr>
<tr>
<td>Input power level to the unknown mixer</td>
<td>SYST:CAL:PHAS:UNKN:INPut:POWer</td>
<td>UnknownMixerInputPower</td>
</tr>
<tr>
<td>Connector type and gender of your Cal Kit.</td>
<td>SYST:CAL:PHAS:CONNector</td>
<td>ConnectorType PhaseRef</td>
</tr>
<tr>
<td>Cal Kit to be used to perform the S-parameter Cal</td>
<td>SYST:CAL:PHAS:CKIT</td>
<td>CalKitType PhaseRef</td>
</tr>
<tr>
<td>Set the ports to be calibrated.</td>
<td>SYST:CAL:PHAS:PORT[X]</td>
<td>IncludePort</td>
</tr>
<tr>
<td>Reverse Port2 Coupler</td>
<td>SYST:CAL:PHAS:DEEMbed</td>
<td>DeembedCoupler</td>
</tr>
</tbody>
</table>

CalPod as ECaI

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns whether the specified device is a CalPod module</td>
<td>SENS:CORR:CKIT:ECAL:CHAR:INSitu:ENABle?</td>
<td>SupportsInSituCharacterization</td>
</tr>
<tr>
<td>Sets or returns whether the specified device will be characterized as an in situ device.</td>
<td>SENS:CORR:CKIT:ECAL:CHAR:INSitu:STATe</td>
<td>InSituCharacterization</td>
</tr>
</tbody>
</table>

Read Date and Time

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read the last modified date of a Cal Set</td>
<td>CSET DATE?</td>
<td>None</td>
</tr>
<tr>
<td>Read the last modified time of a Cal Set</td>
<td>CSET:TIME?</td>
<td>None</td>
</tr>
<tr>
<td>Read the last modified date of a file</td>
<td>MMEM:DATE?</td>
<td>None</td>
</tr>
<tr>
<td>Read the last modified time of a file</td>
<td>MMEM:TIME?</td>
<td>None</td>
</tr>
</tbody>
</table>

Miscellaneous

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns whether the specified external device responds.</td>
<td>SYSTem:CONFigure:EDEVice:EXISits?</td>
<td>None</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release A.09.85**  See What's New.
### Get/Save Noise Parameters SNP Data

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:NOIsE:SNP?</td>
<td>GetSnPData</td>
</tr>
<tr>
<td>SENS:NOIsE:SNP:SAVE</td>
<td>WriteSnPData</td>
</tr>
</tbody>
</table>

### Modify TRL Cal Kit

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:COLL:CKIT:TRL:RPLane</td>
<td>None</td>
</tr>
<tr>
<td>SENS:CORR:COLL:CKIT:TRL:IMPedance</td>
<td>None</td>
</tr>
<tr>
<td>SENS:CORR:COLL:CKIT:TRL:LRLChar</td>
<td>None</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release A.09.80** See What's New.

### Calibrate All Channels

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:CAL:ALL:CHAN:SEL</td>
<td>Channels</td>
</tr>
<tr>
<td>SYST:CAL:ALL:IFBW</td>
<td>IFBW</td>
</tr>
<tr>
<td>SYST:CAL:ALL:PORT:SOUR:POW</td>
<td>PowerLevel</td>
</tr>
<tr>
<td>SYST:CAL:ALL:CSET:PREFix</td>
<td>UserCalsetPrefix</td>
</tr>
<tr>
<td>SYST:CAL:ALL:PATH:CONF:ELEM</td>
<td>PathConfigurationElement</td>
</tr>
<tr>
<td>SYST:CAL:ALL:MCL:PROP:VAL</td>
<td>PropertyValue</td>
</tr>
<tr>
<td>SYST:CAL:ALL:GUID:CHAN?</td>
<td>N/A</td>
</tr>
<tr>
<td>SYST:CAL:ALL:CHAN:PORT</td>
<td>CalibrationPorts</td>
</tr>
<tr>
<td>SYST:CAL:ALL:GUID:PORT?</td>
<td>SParameterCalPorts</td>
</tr>
<tr>
<td>SYST:CAL:ALL:CSET:CATalog?</td>
<td>GeneratedCalsets</td>
</tr>
</tbody>
</table>
### SMC Phase Reference Cal

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>SYST:CAL:ALL:PHAS:RES</td>
<td>Reset</td>
</tr>
<tr>
<td>Start Freq</td>
<td>SYST:CAL:PHAS:FREQ:STAR</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop Freq</td>
<td>SYST:CAL:PHAS:FREQ:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>Read channel number</td>
<td>SYST:CAL:PHAS:GUID:CHAN?</td>
<td>N/A</td>
</tr>
<tr>
<td>Read all Phase Reference IDs</td>
<td>SYST:CAL:PHAS:REF:CAT?</td>
<td>GetConnectedPhaseReferences</td>
</tr>
<tr>
<td>Set Phase Reference</td>
<td>SYST:CAL:PHAS:REF</td>
<td>PhaseReference</td>
</tr>
<tr>
<td>Set Source Attenuator</td>
<td>SYST:CAL:PHAS:POW:ATT</td>
<td>SourceAttenuator</td>
</tr>
<tr>
<td>Set Cal Set name</td>
<td>N/A</td>
<td>CalSet</td>
</tr>
<tr>
<td>Perform Cal</td>
<td>Guided Cal commands</td>
<td>Guided Cal commands</td>
</tr>
</tbody>
</table>

### Noise Cal using Power Meter

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select Receiver Cal method</td>
<td>SENS:NOIS:CAL:RMET</td>
<td>RcvCharMethod</td>
</tr>
</tbody>
</table>

### PMAR Cal and Zero

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Zero method</td>
<td>SYST:CONF:EDEV:PMAR:ZERO</td>
<td>None</td>
</tr>
<tr>
<td>Perform Cal</td>
<td>SYST:CONF:EDEV:PMAR:CAL</td>
<td>None</td>
</tr>
</tbody>
</table>

### Bal - Single-ended topology

<table>
<thead>
<tr>
<th>Command</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Bal-SE measurement</td>
<td>CALC:FSIM:BAL:PAR:BALS</td>
<td>BalSMeasurement</td>
</tr>
<tr>
<td>Set Bal-SE topology (modified)</td>
<td>CALC:FSIM:BAL:DEvice</td>
<td>DUTTopology</td>
</tr>
<tr>
<td>Set port mapping</td>
<td>CALC:FSIM:BAL:TOP:BALS:PPORts</td>
<td>SetBSPorts</td>
</tr>
<tr>
<td>Read se port</td>
<td>CALC:FSIM:BAL:TOP:BALS:PPORts</td>
<td>BS_SEPort</td>
</tr>
<tr>
<td>Read Neg Bal port</td>
<td>CALC:FSIM:BAL:TOP:BALS:PPORts</td>
<td>BS_BalPortNegative</td>
</tr>
<tr>
<td>Read Pos Bal port</td>
<td>CALC:FSIM:BAL:TOP:BALS:PPORts</td>
<td>BS_BalPortPositive</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release A.09.50**  See What's New
<table>
<thead>
<tr>
<th><strong>External DC Analyzer Control</strong></th>
<th><strong>SCPI</strong></th>
<th><strong>COM</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Correction ON/OFF</td>
<td>SYST:CONF:EDEVice:DC:CORR</td>
<td>DCCorrection</td>
</tr>
<tr>
<td>Offset correction value.</td>
<td>SYST:CONF:EDEVice:DC:OFFS</td>
<td>DCOffset</td>
</tr>
<tr>
<td>Scale correction value.</td>
<td>SYST:CONF:EDEVice:DC:SCALe</td>
<td>DCScale</td>
</tr>
<tr>
<td>DC Type (Units).</td>
<td>SYST:CONF:EDEVice:DC:TYPE</td>
<td>DCType</td>
</tr>
<tr>
<td>Dwell Before/After Point</td>
<td>SYST:CONF:EDEVice:DC:DPOint</td>
<td>DwellPerPoint</td>
</tr>
<tr>
<td>Dwell Before Sweep value</td>
<td>SYST:CONF:EDEVice:DC:DSWEEP</td>
<td>SweepDwell</td>
</tr>
</tbody>
</table>

### DC Source Sweep

<table>
<thead>
<tr>
<th><strong>Source names catalog</strong></th>
<th>SOUR:DC:Names</th>
<th>Sources</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable source outputs</td>
<td>SOUR:DC:Enable</td>
<td>EnableAllOutput</td>
</tr>
<tr>
<td>Source state</td>
<td>SOUR:DC:State</td>
<td>State</td>
</tr>
<tr>
<td>Start DC</td>
<td>SOUR:DC:Start</td>
<td>Start</td>
</tr>
<tr>
<td>Stop DC</td>
<td>SOUR:DC:Stop</td>
<td>Stop</td>
</tr>
<tr>
<td>Data</td>
<td>SOUR:DC:Data</td>
<td>ListData</td>
</tr>
</tbody>
</table>

### External Pulse Generator Control

<table>
<thead>
<tr>
<th><strong>PG Names catalog</strong></th>
<th>SENS:PULSE:CAT?</th>
<th>PulseGeneratorNames</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read the integer of the name</td>
<td>Not applicable</td>
<td>PulseGeneratorID</td>
</tr>
<tr>
<td>Set output channel</td>
<td>SYST:CONF:EDEVice:PULSE:CHAN</td>
<td>OutputChannel</td>
</tr>
<tr>
<td>Set output Hi amplitude (volts)</td>
<td>SYST:CONF:EDEVice:PULSE:HAMP</td>
<td>HighAmplitude</td>
</tr>
<tr>
<td>Set output Low amplitude (volts)</td>
<td>SYST:CONF:EDEVice:PULSE:LAMP</td>
<td>LowAmplitude</td>
</tr>
<tr>
<td>Set load impedance</td>
<td>SYST:CONF:EDEVice:PULSE:LIMP</td>
<td>LoadImpedance</td>
</tr>
<tr>
<td>Set source impedance</td>
<td>SYST:CONF:EDEVice:PULSE:SIMP</td>
<td>SourceImpedance</td>
</tr>
<tr>
<td>Master Mode</td>
<td>SYST:CONF:EDEVice:PULSE:MMODE</td>
<td>MasterMode</td>
</tr>
</tbody>
</table>

Optional Name/ID argument added to some Pulse gen commands.
### Miscellaneous

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read configured device names</td>
<td>DeviceNames</td>
</tr>
<tr>
<td>Is a device name present?</td>
<td>IsDevicePresent</td>
</tr>
<tr>
<td>Move a trace to another window</td>
<td>Trace.Move</td>
</tr>
<tr>
<td>Recall softkey list sort preference</td>
<td>RecallSoftkeysMostRecent</td>
</tr>
<tr>
<td>Default to &quot;honor the channel &lt;ch&gt; argument in guided calibration SCPI commands.&quot;</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** This MAY impact your legacy programs.

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receiver Leveling frequency range</td>
<td>FrequencyType</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release A.09.42**  See What's New

### Misc Commands

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read CalSet stimulus values</td>
<td>GetErrorTermStimulus</td>
</tr>
<tr>
<td>Write CalSet stimulus values</td>
<td>PutErrorTermStimulus</td>
</tr>
<tr>
<td>Copy mechanical settings from specified channel</td>
<td>CopyFrom</td>
</tr>
<tr>
<td>Return a measurement handle of the trace object</td>
<td>Measurement</td>
</tr>
<tr>
<td>Read the PNA port which is connected to the DUT input.</td>
<td>DeviceInputPort</td>
</tr>
<tr>
<td>Read the PNA port which is connected to the DUT output.</td>
<td>DeviceOutputPort</td>
</tr>
<tr>
<td>Read the window number of the selected trace.</td>
<td>WindowNumber</td>
</tr>
<tr>
<td>Read the trace number of the selected trace.</td>
<td>N/A</td>
</tr>
</tbody>
</table>

### Source / Role Commands (for apps)

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the roles for which sources can be used for the channel.</td>
<td>DefinedRoles</td>
</tr>
<tr>
<td>Set and returns the source to be used in the specified role.</td>
<td>RoleDevice</td>
</tr>
</tbody>
</table>

---

5672
### IMD and IM Spectrum Tone Power settings

<table>
<thead>
<tr>
<th>IMD and IM Spectrum Tone Power settings</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swept IMD Equal tone power</td>
<td>SENSe:IMD:TPOwer:EQualize:STATe</td>
<td>EqualTonePower</td>
</tr>
<tr>
<td>IM Spectrum Equal tone power</td>
<td>SENSe:IMD:TPOwer:SET</td>
<td></td>
</tr>
<tr>
<td>Swept IMD Set power at DUT input or output</td>
<td>SENSe:IMS:TPOwer:EQualize:STATe</td>
<td>TonePowerSetAt</td>
</tr>
<tr>
<td>IM Spectrum set power at DUT input or output</td>
<td>SENSe:IMS:TPOwer:SET</td>
<td></td>
</tr>
</tbody>
</table>

### Pulse Commands

<table>
<thead>
<tr>
<th>Pulse Commands</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set master pulse frequency</td>
<td>SENs:SWE:PULSe:MASTer:FREQ</td>
<td>MasterFrequency</td>
</tr>
<tr>
<td>Set master pulse period</td>
<td>SENs:SWE:PULSe:MASTer:PERiod</td>
<td>MasterPeriod</td>
</tr>
<tr>
<td>Set master pulse width</td>
<td>SENs:SWE:PULSe:MASTer:WIDTh</td>
<td>MasterWidth</td>
</tr>
<tr>
<td>Autoselect CW Sweep Time</td>
<td>SENs:SWE:PULSe:CWTIme</td>
<td>AutoCW Sweep Time</td>
</tr>
</tbody>
</table>

### Capabilities Commands

<table>
<thead>
<tr>
<th>Capabilities Commands</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return parameters supported by the specified Measurement Class</td>
<td>None</td>
<td>SupportedParameters</td>
</tr>
<tr>
<td>Return list of supported IFBWs</td>
<td>None</td>
<td>IFBWList</td>
</tr>
<tr>
<td>Return list of ResBW's supported by IMSpectrum Apps</td>
<td>None</td>
<td>ResBWList</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release A.09.33**  See What's New

### FCA Commands

<table>
<thead>
<tr>
<th>FCA Commands</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replaced SESSION commands.</td>
<td>SENs:Correction:Guided</td>
<td>Not Applicable</td>
</tr>
<tr>
<td></td>
<td>SENs:Correction:Collect:Guided:SMC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENs:Correction:Collect:Guided:VMC</td>
<td></td>
</tr>
</tbody>
</table>
### Multiple Power Sensors

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable multiple sensors</td>
<td>SENS:CORR:COLL:GUID:PSEN:MULT</td>
<td>UseMultipleSensors</td>
</tr>
<tr>
<td>Assign power sensor name</td>
<td>SENS:CORR:COLL:GUID:PSEN:MULT:NAME</td>
<td>Name</td>
</tr>
<tr>
<td>Read the number of configured sensors</td>
<td>SENS:CORR:COLL:GUID:PSEN:MULT:COUNt?</td>
<td>Count</td>
</tr>
<tr>
<td>Set connector type</td>
<td>SENS:CORR:COLL:GUID:PSEN:MULT:CONN</td>
<td>PowerSensorConnectorType</td>
</tr>
<tr>
<td>Set Cal Kit</td>
<td>SENS:CORR:COLL:GUID:PSEN:MULT:CKIT</td>
<td>PowerSensorCalKitType</td>
</tr>
</tbody>
</table>

### Phase Control

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase Sweep type</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Set fixed phase value</td>
<td>SOURce:PHAS: [:FIXed]</td>
<td>FixedPhase</td>
</tr>
<tr>
<td>Phase sweep start value</td>
<td>SOURce:PHAS:STARt</td>
<td>StartPhase</td>
</tr>
<tr>
<td>Phase sweep stop value</td>
<td>SOURce:PHAS:STOP</td>
<td>StopPhase</td>
</tr>
<tr>
<td>Phase parameter</td>
<td>SOURce:PHAS:PARameter</td>
<td>PhaseParameter</td>
</tr>
<tr>
<td>Set Phase control mode</td>
<td>SOURce:PHAS:PARameter:MODE</td>
<td>PhaseControlMode</td>
</tr>
<tr>
<td>Set reference port</td>
<td>SOURce:PHAS:PARameter:PORT</td>
<td>PhaseReferencePort</td>
</tr>
<tr>
<td>Read available phase control modes for the port</td>
<td>SOURce:PHAS:PARameter:MODE:CAT?</td>
<td>PhaseParameterModes</td>
</tr>
<tr>
<td>Couple sweep settings</td>
<td>SOURce:PHAS:CONTrol:COUPle</td>
<td>CouplePhasePortSettings</td>
</tr>
<tr>
<td>Set number of sweep iterations</td>
<td>SOURce:PHAS:CONTrol:ITERation</td>
<td>PhaseIterationNumber</td>
</tr>
<tr>
<td>Set sweep tolerance</td>
<td>SOURce:PHAS:CONTrol:TOLerance</td>
<td>PhaseTolerance</td>
</tr>
<tr>
<td>Set and read an array of phase offsets.</td>
<td>SOURce:PHAS:CORRection:DATA</td>
<td>PhaseCorrectionData</td>
</tr>
<tr>
<td>Use phase offset array.</td>
<td>SOURce:PHAS:CORRection:STATe</td>
<td>PhaseCorrectionEnabled</td>
</tr>
<tr>
<td>Set and read an array of ratioed power offsets.</td>
<td>SOURce:PHAS:POFFset:CORR:DATA</td>
<td>RatioedPowerCorrectionData</td>
</tr>
<tr>
<td>Use power offset array.</td>
<td>SOURce:PHAS:POFFset:CORR:STATe</td>
<td>RatioedPowerCorrectionEnabled</td>
</tr>
<tr>
<td>Set the fixed power ratioed value</td>
<td>SOUR:PHAS:POFFset:FIXed</td>
<td>FixedRatioedPower</td>
</tr>
<tr>
<td>Set the start power ratioed value</td>
<td>SOUR:PHAS:POFFset:STARt</td>
<td>StartRatioedPower</td>
</tr>
</tbody>
</table>
Set the stop power ratioed value.

2 and 4-port Fixture De-embed
- 2-port Reverse
- 4-port remap
- Extrapolate
- Extrapolate added to Cal Set De-embedding
- Extrapolate added to Cal Set Embedding

Mixer Segment Sweep
- Recalculate
- Segment Calculate
- Query Count
- Add Segments
- Delete Segments
- Remove All Segments
- State
- Number Of Points
- IFBW
- Input Fixed Freq
- Input Start Freq
- Input Stop Freq
- Input Fixed/Swept
- Input Power
- Output Fixed Freq
- Output Start Freq
- Output Stop Freq
- Output Fixed/Swept
- Output (+/-)
- Output Power
- LO Fixed Freq
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LO Start Freq</td>
<td>SENS:MIX:SEGM&lt;\n&gt;:LO&lt;x&gt;:FREQ:STARt SegmentStartFrequency</td>
</tr>
<tr>
<td>LO Stop Freq</td>
<td>SENS:MIX:SEGM&lt;\n&gt;:LO&lt;x&gt;:FREQ:STOP SegmentStopFrequency</td>
</tr>
<tr>
<td>LO Fixed/Swept</td>
<td>SENS:MIX:SEGM&lt;\n&gt;:LO&lt;x&gt;:FREQ:MODE SegmentRangeMode</td>
</tr>
<tr>
<td>Input &gt;LO</td>
<td>SENS:MIX:SEGM&lt;\n&gt;:LO&lt;x&gt;:FREQ:ILTI SegmentIsInputGreaterThanLO</td>
</tr>
<tr>
<td>LO Power</td>
<td>SENS:MIX:SEGM&lt;\n&gt;:LO&lt;x&gt;:POWer SegmentFixedPower</td>
</tr>
<tr>
<td>IF (+/-)</td>
<td>SENS:MIX:SEGM&lt;\n&gt;:LO&lt;x&gt;:IF:FREQ:SIdeband SegmentMixingMode</td>
</tr>
</tbody>
</table>

### Miscellaneous

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read available Meas Classes</td>
<td>SYST:MCLass:CAT? AvailableMeasurementClasses</td>
</tr>
<tr>
<td>Set receiver ratio to be used with receiver leveling.</td>
<td>SOURce:POWer:ALC:MODE:REC:RA\Ti o ReceiverRatio</td>
</tr>
<tr>
<td>Removes adapters</td>
<td>SENS:CORR:COLL:GUID:ADAP:COUNt:ZERO None</td>
</tr>
<tr>
<td>Perform Linear Interpolation</td>
<td>SYST:CORR:INT:LINear None</td>
</tr>
<tr>
<td>Return if a CalSet exists</td>
<td>CSET:EXISts? Exists</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release A.09.30** See What's New

### Marker Display

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Readouts Per Trace</td>
<td>DISP:WIND:ANN:MARK:NUMB MarkerReadoutsPerTrace</td>
</tr>
<tr>
<td>Marker symbol</td>
<td>DISP:WIND:ANN:MARK:SYMB MarkerSymbol</td>
</tr>
</tbody>
</table>

**GCX - No new commands. Learn more.**

### Overload Preferences

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turn source power OFF when a receiver is overloaded?</td>
<td>SYST:PREF:ITEM:REC:OVER:POW RFOffOnReceiverOverload</td>
</tr>
</tbody>
</table>
### Miscellaneous

<table>
<thead>
<tr>
<th>Use Last Result for Source Power Cal</th>
<th>SOUR:POW:ALC:MODE:REC:LSPC</th>
<th>LastLevelingAsSPC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:CHANnels:DELe</td>
<td>Remove Method</td>
<td>RemoveChannelNumber</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Selectively delete a channel</th>
</tr>
</thead>
</table>

### Guided Power Cal

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Match-correction ON</td>
<td>OFF</td>
<td>SENS:CORR:METHods:MATCh</td>
</tr>
</tbody>
</table>

### Cal Set Items

<table>
<thead>
<tr>
<th>Set or get name-value pair from calset</th>
<th>None</th>
<th>Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remove name-value pair from calset</td>
<td>None</td>
<td>RemoveItem</td>
</tr>
<tr>
<td>Enumerate name-value pair items in the calset.</td>
<td>None</td>
<td>EnumerateCalSets</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release A.09.22**  See What's New

<table>
<thead>
<tr>
<th>Returns the error correction state for the measurement.</th>
<th>CAL:CORR:INDicator?</th>
<th>ErrorCorrectionIndicator</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Calibrating specific channels</th>
<th>SENS:CORR:COLL:GUID:CHAN:MODE</th>
<th>N/A</th>
</tr>
</thead>
</table>

### Port Mapping - Noise Figure Opt 028

<table>
<thead>
<tr>
<th>Write port mapping</th>
<th>SENSE:NOISe:PMAP</th>
<th>SetPortMap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read input port mapping</td>
<td>SENSE:NOISe:PMAP:INPut?</td>
<td>DeviceInputPort</td>
</tr>
<tr>
<td>Read output port mapping</td>
<td>SENSE:NOISe:PMAP:OUTPut?</td>
<td>DeviceOutputPort</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release A.09.20**  See What's New
Configure Pulse Measurements

- **Pulse Meas Mode**  
  SENS:SWE:PULS:MODE  
  PulseMeasMode

- **Autodetect Pulse mode**  
  SENS:SWE:PULS:DETECTmode  
  AutoDetection

- **Set Pulse Mode (Narrow | Wide)**  
  SENS:SWE:PULS:WIDEband  
  WideBandDetectionState

- **Autoselect IFBW**  
  SENS:SWE:PULS:IFBW  
  AutoIFBandWidth

- **Autoselect IF Gain**  
  SENS:SWE:PULS:IFGain  
  AutoIFGain

- **Autoselect Pulse clock period**  
  SENS:SWE:PULS:PRF  
  AutoOptimizePRF

- **Autoselect Width and Delay**  
  SENS:SWE:PULS:TIMing  
  AutoPulseTiming

- **Autoselect Pulse Gens**  
  SENS:SWE:PULS:DRIVE  
  AutoSelectPulseGen

External Pulse Configuration

- **PulseSyncIn Trigger Polarity**  
  Sense:Pulse:TPolarity  
  TriggerInPolarity

- **PulseSyncIn Trigger Type**  
  Sense:Pulse:TType  
  TriggerInType

PSAT Marker Search

- **Initiate a PSAT search**  
  CALC:MARK:PSAT:BACKoff  
  SearchPowerSaturation

- **Set and read PSAT backoff**  
  CALC:MARK:PSAT:BACKoff  
  PMaxBackOff

- **Read PSat Out**  
  CALC:MARK:PSAT:POUT?  
  POut

- **Read PSat In**  
  CALC:MARK:PSAT:PIN?  
  Pin

- **Read PMax Out**  
  CALC:MARK:PSAT:POUT:MAXimum?  
  PMaxOut

- **Read PMax In**  
  CALC:MARK:PSAT:PIN:MAXimum?  
  PMaxIn

- **Read Gain Sat**  
  CALC:MARK:PSAT:GAIN?  
  GainSaturation

- **Read Gain Max**  
  CALC:MARK:PSAT:GAIN:MAXimum?  
  GainMax

- **Read Gain Linear**  
  CALC:MARK:PSAT:GAIN:LINEar?  
  GainLinear

- **Read Comp Sat**  
  CALC:MARK:PSAT:COMP:SAT?  
  CompressionSaturation

- **Read Comp Max**  
  CALC:MARK:PSAT:COMP:MAX?  
  CompressionMax
## PNOP Marker Search

<table>
<thead>
<tr>
<th>Operation</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiate a PNOP search</td>
<td>CALC:MARK:PNOP:BACKoff</td>
<td>SearchPNOP marker Normal Operating Point</td>
</tr>
<tr>
<td>Set and read PNOP backoff</td>
<td>CALC:MARK:PNOP:BACKoff</td>
<td>BackOff</td>
</tr>
<tr>
<td>Set and read PNOP Power Offset</td>
<td>CALC:MARK:PNOP:POFFset</td>
<td>PinOffset</td>
</tr>
<tr>
<td>Read Pnop Out</td>
<td>CALC:MARK:PNOP:POUT?</td>
<td>POut</td>
</tr>
<tr>
<td>Read Pnop in</td>
<td>CALC:MARK:PNOP:PIN?</td>
<td>Pin</td>
</tr>
<tr>
<td>Read Pnop Gain</td>
<td>CALC:MARK:PNOP:GAIN?</td>
<td>Gain</td>
</tr>
<tr>
<td>Read Pnop Comp</td>
<td>CALC:MARK:PNOP:COMPression?</td>
<td>Compression</td>
</tr>
<tr>
<td>Read PMax Out</td>
<td>CALC:MARK:PNOP:POUT:MAXimum?</td>
<td>PMaxOut</td>
</tr>
<tr>
<td>Read PMax In</td>
<td>CALC:MARK:PNOP:PIN:MAXimum?</td>
<td>PMaxIn</td>
</tr>
<tr>
<td>Read Gain Max</td>
<td>CALC:MARK:PNOP:GAIN:MAXimum?</td>
<td>GainMax</td>
</tr>
<tr>
<td>Read Comp Max</td>
<td>CALC:MARK:PNOP:COMPression:MAXimum?</td>
<td>CompressionMax</td>
</tr>
<tr>
<td>Read PBO In</td>
<td>CALC:MARK:PNOP:BACKoff:PIN?</td>
<td>BackOffPin</td>
</tr>
</tbody>
</table>

## Include Phase in SMC measurements

<table>
<thead>
<tr>
<th>Operation</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Phase</td>
<td>SENS:MIX:PHASE</td>
<td>EnablePhase</td>
</tr>
<tr>
<td>Set normalize point</td>
<td>SENS:MIX:NORMalize</td>
<td>NormalizePoint</td>
</tr>
<tr>
<td>Choose known delay or S2P</td>
<td>SENS:CORR:COLL:SESS:SMC:PHAse:METHod</td>
<td>DelayCalculationMethod</td>
</tr>
<tr>
<td>filename</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Group Delay Aperture

<table>
<thead>
<tr>
<th>Operation</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set aperture using frequency</td>
<td>CALC:GDElay:FREQ</td>
<td>Frequency</td>
</tr>
<tr>
<td>Set aperture using percent of span</td>
<td>CALC:GDElay:PERCent</td>
<td>Percent</td>
</tr>
<tr>
<td>Set aperture using fixed number of points</td>
<td>CALC:GDElay:POINts</td>
<td>Points</td>
</tr>
<tr>
<td>Set Preference to 2 points</td>
<td>SYST:PREF:ITEM:GDElay:TWOP</td>
<td>TwoPointGroupDelayAperture</td>
</tr>
</tbody>
</table>
## Calibrations

Returns the names of the mechanical cal kits for unguided calibrations.

Set/get by name which cal kit is currently selected for use by unguided cal.

Read pass fail status of tolerance limits on the target cal power

Gets the power correction dB values from the prior iteration of the source power cal.

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:CORR:COLL:CKIT:CAT?</td>
</tr>
<tr>
<td>SENSe:CORR:COLL:CKIT:PORT</td>
</tr>
<tr>
<td>Not needed - read from SCPI error queue</td>
</tr>
<tr>
<td>SOURce:POWer:CORR:DATA:PRIor</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set/get</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalKitTypes</td>
</tr>
<tr>
<td>CalKitType</td>
</tr>
<tr>
<td>LastCalPassedTolerance</td>
</tr>
<tr>
<td>getSourcePowerCalDataEx</td>
</tr>
<tr>
<td>getSourcePowerCalDataScalarEx</td>
</tr>
</tbody>
</table>

## Fixturing

Compensate source power

Create Cal Set with De-embed fixture removed.

Create Cal Set with Matching Network included.

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:SEND:POW:PORT:COMP</td>
</tr>
<tr>
<td>CSET:FIXTure:DEEMbed</td>
</tr>
<tr>
<td>CSET:FIXTure:EMBed</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Enable</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnablePowerCompensation</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

## Noise Receiver (Opt 028)

Select Noise Receiver

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:NOISe:RECeiver</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set/get</th>
</tr>
</thead>
<tbody>
<tr>
<td>NoiseReceiver</td>
</tr>
</tbody>
</table>
### Miscellaneous

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include SC12 Sweep</td>
<td>Sens:Mixer:Reverse</td>
</tr>
<tr>
<td>Active Window Background</td>
<td>Disp:Color:ABACKground</td>
</tr>
<tr>
<td>Sweep Delay</td>
<td>SENS:SWEep:DWELl:SDElay</td>
</tr>
<tr>
<td><strong>Preference:</strong> On PRESET always turn power ON</td>
<td>SYST:PREF:ITEM:PRES:POWer</td>
</tr>
<tr>
<td>Send and return Instrument state file to remote computer</td>
<td>SYST:SET</td>
</tr>
<tr>
<td>Read the display image into remote computer</td>
<td>HCOPY:SDUMp:DATA?</td>
</tr>
<tr>
<td>Set format of display image</td>
<td>HCOPY:SDUMp:DATA:FORM</td>
</tr>
<tr>
<td>Set and return LXI status</td>
<td>LXI:IDEN</td>
</tr>
<tr>
<td>GCA Safe mode - max power</td>
<td>Sens:GCS:SAFE:MLIM</td>
</tr>
<tr>
<td>Set Line type of the window grid</td>
<td>DISP:WIND:TRAC:GRAT:GRID:LTYPE</td>
</tr>
<tr>
<td>Save Data</td>
<td>MMEM:STOR:DATA</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release 9.10**  See What's New

### NFX

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets the power sensor connector</td>
<td>SENS:CORR:COLL:GUID:PSEN:CONN</td>
</tr>
<tr>
<td>Sets the power sensor calkit</td>
<td>SENS:CORR:COLL:GUID:PSEN:CKIT</td>
</tr>
<tr>
<td>Sets power level for source power cal</td>
<td>SENS:CORR:COLL:GUID:PSEN:POW:LEV</td>
</tr>
<tr>
<td>Sets auto orientation state for noise tuner</td>
<td>SENS:NOISE:TUNer:ORIent</td>
</tr>
<tr>
<td>Sets LO power calibration state</td>
<td>SENS:CORR:COLL:NOIS:LO:PCAL:STATe</td>
</tr>
<tr>
<td>Sets the source pull technique to compute DUT S-parameters</td>
<td>SENS:NOISE:PULL</td>
</tr>
<tr>
<td>Sets the state of ENR adapter de-embedding.</td>
<td>SENS:CORR:COLL:NOIS:ENR:ADAP:DEEM</td>
</tr>
<tr>
<td>Sets the state of Power Sensor adapter de-embedding.</td>
<td>SENS:CORR:COLL:NOIS:PSEN:ADAP:DEEM</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PowerSensorConnectorType</td>
<td>PowerSensorConnectorType</td>
</tr>
<tr>
<td>PowerSensorCalKitType</td>
<td>PowerSensorCalKitType</td>
</tr>
<tr>
<td>PowerCalibrationLevel</td>
<td>PowerCalibrationLevel</td>
</tr>
<tr>
<td>AutoOrientTuner</td>
<td>AutoOrientTuner</td>
</tr>
<tr>
<td>EnableLOPowerCal</td>
<td>EnableLOPowerCal</td>
</tr>
<tr>
<td>SourcePullForSParameters</td>
<td>SourcePullForSParameters</td>
</tr>
<tr>
<td>ForceDeEmbedENRAdapter</td>
<td>ForceDeEmbedENRAdapter</td>
</tr>
<tr>
<td>ForceDeEmbedSensorAdapter</td>
<td>ForceDeEmbedSensorAdapter</td>
</tr>
</tbody>
</table>
### Help About commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Command String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSP Revision</td>
<td>SYST:CONF:REVision:DSP?</td>
<td>DspRevision</td>
</tr>
<tr>
<td>DSP FPGA</td>
<td>SYST:CONF:REVision:DSPFpga?</td>
<td>DspFpgaRevision</td>
</tr>
<tr>
<td>CPU Speed</td>
<td>SYST:CONF:REVision:CPU?</td>
<td>CpuRevision</td>
</tr>
<tr>
<td>Hostname</td>
<td>SYST:COMM:LAN:HOSTname?</td>
<td>GetIPConfigurationStruct</td>
</tr>
</tbody>
</table>

### Miscellaneous

<table>
<thead>
<tr>
<th>Command</th>
<th>Command String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perform and apply Response (Normalization) cal</td>
<td>SENS:CORR:COLL:METHod</td>
<td>DoResponseCal</td>
</tr>
<tr>
<td>PulseSyncIn Trigger Polarity</td>
<td>Sense: Pulse: TPolarity</td>
<td>TriggerInPolarity</td>
</tr>
<tr>
<td></td>
<td>Sense: Pulse: TType</td>
<td>TriggerInType</td>
</tr>
<tr>
<td>PulseSyncIn Trigger Type</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release 9.0**  See What's New

### Configure External Devices

<table>
<thead>
<tr>
<th>Command</th>
<th>Command String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adds an external device to the system.</td>
<td>SYST:CONF:EDEV:ADD</td>
<td>Add (External Device)</td>
</tr>
<tr>
<td>Returns names of all configured devices</td>
<td>SYST:CONF:EDEV:CAT?</td>
<td>Items</td>
</tr>
<tr>
<td>Set driver for the external device.</td>
<td>SYST:CONF:EDEV:DRIV</td>
<td>Driver</td>
</tr>
<tr>
<td>Set type of device.</td>
<td>SYST:CONF:EDEV:DTYP</td>
<td>DeviceType</td>
</tr>
<tr>
<td>Configuration path for external device.</td>
<td>SYST:CONF:EDEV:IOC</td>
<td>IOConfiguration</td>
</tr>
<tr>
<td>Enable or disable communication with device.</td>
<td>SYST:CONF:EDEV:IOEN</td>
<td>IOEnable</td>
</tr>
<tr>
<td>Activation state of the device.</td>
<td>SYST:CONF:EDEV:STAT</td>
<td>Active</td>
</tr>
<tr>
<td>Time out value for external device.</td>
<td>SYST:CONF:EDEV:TOT</td>
<td>TimeOut</td>
</tr>
</tbody>
</table>

### External Source Config

<table>
<thead>
<tr>
<th>Command</th>
<th>Command String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Dwell per Point</td>
<td>SYST:CONF:EDEV:SOUR:DPP</td>
<td>DwellPerPoint</td>
</tr>
<tr>
<td>Set Trigger Mode</td>
<td>SYST:CONF:EDEV:SOUR:TMOD</td>
<td>Trigger Mode</td>
</tr>
<tr>
<td>Set Trigger Port</td>
<td>SYST:CONF:EDEV:SOUR:TPORt</td>
<td>TriggerPort</td>
</tr>
</tbody>
</table>
## Power Meter As Receiver (PMAR) Config

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable min and max freqs</td>
<td>SYST:CONF:EDEV:PMAR:FLIM</td>
<td>LimitFrequency</td>
</tr>
<tr>
<td>Set Max freq</td>
<td>SYST:CONF:EDEV:PMAR:FMAX</td>
<td>MaximumFrequency</td>
</tr>
<tr>
<td>Set Min freq</td>
<td>SYST:CONF:EDEV:PMAR:FMIN</td>
<td>MinimumFrequency</td>
</tr>
<tr>
<td>Set max number of PM readings</td>
<td>SYST:CONF:EDEV:PMAR:READ:COUN</td>
<td>ReadingsPerPoint</td>
</tr>
<tr>
<td>Set tolerance level</td>
<td>SYST:CONF:EDEV:PMAR:READ:NTOL</td>
<td>ReadingsTolerance</td>
</tr>
<tr>
<td>Select sensor</td>
<td>SYST:CONF:EDEV:PMAR:SSENS</td>
<td>SensorIndex</td>
</tr>
</tbody>
</table>

## Power Limit

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set power limit</td>
<td>SYST:POWer:LIMit</td>
<td>Limit</td>
</tr>
<tr>
<td>Power limit ON/OFF</td>
<td>SYST:POWer:LIMit:STATe</td>
<td>State</td>
</tr>
<tr>
<td>Power limit UI lock</td>
<td>SYST:POWer:LIMit:LOCK</td>
<td>Lock</td>
</tr>
</tbody>
</table>

## Scale Coupling

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set method</td>
<td>DISP:WIND:TRAC:Y:COUP:METH</td>
<td>ScaleCouplingMethod</td>
</tr>
<tr>
<td>Enable window</td>
<td>DISP:WIND:TRAC:Y:COUP</td>
<td>ScaleCouplingState</td>
</tr>
</tbody>
</table>
Display and Print Colors

Set background color
Set labels and grid frame colors
Set inner lines of all grids in all windows colors
Set Inactive window label colors
Set limit line colors
Set trace data and Limit Line colors
Set data trace marker colors
Set memory trace colors
Set memory trace marker colors
Load a color theme
Saves the current color theme.
Resets to the default PNA colors.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:COL:BACK</td>
<td>Background</td>
</tr>
<tr>
<td>DISP:COL:GRAT1</td>
<td>ActiveLabels</td>
</tr>
<tr>
<td>DISP:COL:GRAT2</td>
<td>Grid</td>
</tr>
<tr>
<td>DISP:COL:ILAB</td>
<td>InactiveLabels</td>
</tr>
<tr>
<td>DISP:COL:LIM1</td>
<td>FailedTraces</td>
</tr>
<tr>
<td>DISP:COL:TRAC:DATA</td>
<td>DataAndLimits</td>
</tr>
<tr>
<td>DISP:COL:TRAC:MARK</td>
<td>Markers</td>
</tr>
<tr>
<td>DISP:COL:TRAC:MEM</td>
<td>Memory</td>
</tr>
<tr>
<td>DISP:COL:TRAC:MMAR</td>
<td>MemoryMarkers</td>
</tr>
<tr>
<td>DISP:COL:LOAD</td>
<td>LoadTheme</td>
</tr>
<tr>
<td>DISP:COL:STOR</td>
<td>StoreTheme</td>
</tr>
<tr>
<td>DISP:COL:RES</td>
<td>ResetTheme</td>
</tr>
</tbody>
</table>

IMD and IMS Limited Port Mapping

Set port map
Read Input
Read Output

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:IMD:PMAP</td>
<td>SetPortMap</td>
</tr>
<tr>
<td>SENS:IMS:PMAP</td>
<td>DeviceInputPort</td>
</tr>
<tr>
<td>SENS:IMD:PMAP:INP?</td>
<td>DeviceInputPort</td>
</tr>
<tr>
<td>SENS:IMS:PMAP:INP?</td>
<td>DeviceInputPort</td>
</tr>
<tr>
<td>SENS:IMD:PMAP:OUTP?</td>
<td>DeviceOutputPort</td>
</tr>
<tr>
<td>SENS:IMS:PMAP:OUTP?</td>
<td>DeviceOutputPort</td>
</tr>
</tbody>
</table>
**ECal User Char to Disk (new and modified commands)**

Returns the factory defined and user-defined connectors.

None

Initiate a User Char - optionally check module memory.

*SENSe:CORR:CKIT:ECAL:CHAR:INIT*  
InitializeEx

Saves a new characterization to disk memory

*SENSe:CORR:CKIT:ECAL:CHAR:DMEM:SAVE*  
SaveToDiskMemory

Delete disk memory characterizations.

*SENSe:CORR:CKIT:ECAL:DMEMory:CLEar*  
None

Saves a disk memory characterization to an archive file.

*SENSe:CORR:CKIT:ECAL:EXPort*  
None

Imports the ECal characterization from the specified archive file.

*SENSe:CORR:CKIT:ECAL:DMEMory:IMPort*  
None

Reads the user-characterization info from ECal module or PNA disk memory.

*SENSe:CORR:CKIT:ECAL:KNAM:INF?*  
None

**Gain Compression Analysis**

Enable a compression analysis trace

*CALC:GCM:ANAL:ENAB*  
AnalysisEnable

Set CW frequency

*CALC:GCM:ANAL:CWFR*  
AnalysisCWFreq

Set to discrete or interpolated CW frequencies

*CALC:GCM:ANAL:ISD*  
AnalysisIsDiscreteFreq

Sets X-axis display

*CALC:GCM:ANAL:XAX*  
AnalysisXAxis

**Miscellaneous**

Red Segment on Limit Line failures.

*SYST:PREF:ITEM:RTOF*  
RedTraceOnFail

Returns the X-axis values for the selected trace.

*CALC:X?*

Saves a Cal Kit to a file.

*SENSe:CORR:CKIT:EXP*  
GetCompatibleCalKits

Returns the list of cal kits that use the specified connector.

*SENSe:CORR:COLL:GUID:CKIT:CAT?*  
GetCompatibleCalKits

Gain Compression Saturation level

*SENSe:GCS:COMP:SAT:LEV*  
SaturationLevel

Set Cal FOM Range Preference

*SENSe:CORR:PREF:CAL:RANG*  
FrequencyOffsetRangeForCalComputations
Return the Calset X-axis FOM frequency range

SMC - perform separate power cals

The following are new programming commands for **PNA release 8.60**  See What's New

<table>
<thead>
<tr>
<th><strong>Miscellaneous</strong></th>
<th><strong>SCPI</strong></th>
<th><strong>COM</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Reads or writes the segment sweep table.</td>
<td><strong>SENSe:SEGM:LIST</strong></td>
<td><strong>GetAllSegments</strong></td>
</tr>
<tr>
<td>Optional arguments for...</td>
<td><strong>Sens:Corr:Ckit:CLEar</strong></td>
<td><strong>Sens:Corr:Ckit:Init</strong></td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release 8.55**  See What's New

<table>
<thead>
<tr>
<th><strong>Miscellaneous</strong></th>
<th><strong>SCPI</strong></th>
<th><strong>COM</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable subpoint triggering</td>
<td><strong>Sens:pulse:SUBPointTrig</strong></td>
<td><strong>SubPointTrigger</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th><strong>SCPI</strong></th>
<th><strong>COM</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IMD-X for Converters</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create or Change a Custom (Application) Measurement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create a custom measurement</td>
<td><strong>CALC:CUST:DEF</strong></td>
<td><strong>App.CreateCustomMeasurementEx</strong></td>
</tr>
<tr>
<td>Change a custom measurement</td>
<td><strong>CALC:CUST:MOD</strong></td>
<td></td>
</tr>
<tr>
<td>Return handle to a converter object</td>
<td><strong>None</strong></td>
<td><strong>GetConverter</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Configure a Mixer</strong></th>
<th><strong>SCPI</strong></th>
<th><strong>COM</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Calculate Input and Output frequencies</td>
<td><strong>SENS:MIX:CALC</strong></td>
<td></td>
</tr>
<tr>
<td>X-axis display</td>
<td><strong>CALC:MIX:XAXis</strong></td>
<td><strong>ActiveXAxisRange</strong></td>
</tr>
<tr>
<td>Discard Changes</td>
<td><strong>SENS:MIX:DISCard</strong></td>
<td><strong>DiscardChanges</strong></td>
</tr>
<tr>
<td>Load a mixer setup</td>
<td><strong>SENS:MIX:LOAD</strong></td>
<td><strong>LoadFile</strong></td>
</tr>
<tr>
<td>Save a mixer setup</td>
<td><strong>SENS:MIX:SAVE</strong></td>
<td><strong>SaveFile</strong></td>
</tr>
<tr>
<td>Apply mixer settings</td>
<td><strong>SENS:MIX:APPLY</strong></td>
<td><strong>Apply</strong></td>
</tr>
<tr>
<td>Function</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>Assign a source to mixer input or LO.</td>
<td>SENS:MIX:ROLE:DEVice</td>
<td>AssignSourceToRole</td>
</tr>
<tr>
<td>Read all assigned roles</td>
<td>SENS:MIX:ROLE:CATalog?</td>
<td>GetSourceByRole</td>
</tr>
<tr>
<td>Read the source assigned to a role.</td>
<td>SENS:MIX:ROLE:DEVice</td>
<td>GetSourceRoles</td>
</tr>
</tbody>
</table>

### Set Input

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input start frequency</td>
<td>SENS:MIX:INP:FREQ:STAR</td>
<td>InputStartFrequency</td>
</tr>
<tr>
<td>Input stop frequency</td>
<td>SENS:MIX:INP:FREQ:STOP</td>
<td>InputStopFrequency</td>
</tr>
<tr>
<td>Input power level</td>
<td>SENS:MIX:INP:POW</td>
<td>InputPower</td>
</tr>
<tr>
<td>Input to Swept or fixed</td>
<td>SENS:MIX:INP:FREQ:MODE</td>
<td>InputRangeMode</td>
</tr>
<tr>
<td>Input fixed frequency</td>
<td>SENS:MIX:INP:FREQ:FIXed</td>
<td>InputFixedFrequency</td>
</tr>
</tbody>
</table>

### Set LO

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recall a previously-configured external source. This is the only external LO configuration command.</td>
<td>SENS:MIX:LO:NAME</td>
<td>LOName</td>
</tr>
<tr>
<td>LO freq sweep mode (fixed or swept)</td>
<td>SENS:MIX:LO:FREQ:MODE</td>
<td>LORangeMode</td>
</tr>
<tr>
<td>LO fixed frequency</td>
<td>SENS:MIX:LO:FREQ:FIX</td>
<td>LOFixedFrequency</td>
</tr>
<tr>
<td>LO start frequency</td>
<td>SENS:MIX:LO:FREQ:STARt</td>
<td>LOStartFrequency</td>
</tr>
<tr>
<td>LO stop frequency</td>
<td>SENS:MIX:LO:FREQ:STOP</td>
<td>LOStopFrequency</td>
</tr>
<tr>
<td>LO power</td>
<td>SENS:MIX:LO:POW</td>
<td>LOPower</td>
</tr>
<tr>
<td>LO power start</td>
<td>SENS:MIX:LO:POW:STARt</td>
<td>LOStartPower</td>
</tr>
<tr>
<td>LO power stop</td>
<td>SENS:MIX:LO:POW:STOP</td>
<td>LOStopPower</td>
</tr>
<tr>
<td>Input Greater / Less that LO</td>
<td>SENS:MIX:LO:FREQ:ILTI</td>
<td>IsInputGreaterThanLO</td>
</tr>
</tbody>
</table>

### Set IF

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sideband (high or low)</td>
<td>SENS:MIX:IF:FREQ:SIDE</td>
<td>IFSideband</td>
</tr>
</tbody>
</table>

### Set Output

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sideband (high or low)</td>
<td>SENS:MIX:OUTP:FREQ:SIDE</td>
<td>OutputSideband</td>
</tr>
<tr>
<td>Output start frequency</td>
<td>SENS:MIX:OUTP:FREQ:STAR</td>
<td>OutputStartFrequency</td>
</tr>
<tr>
<td>Output stop frequency</td>
<td>SENS:MIX:OUTP:FREQ:STOP</td>
<td>OutputStopFrequency</td>
</tr>
</tbody>
</table>
### Compression Marker
- Compression Marker level found: \(\text{CALC:MARKer:COMPression:LEVel}\) \(\text{CompressionLevel}\)
- Read Compression Marker Input power: \(\text{CALC:MARKer:COMPression:PIN}\) \(\text{CompressionPin}\)
- Read Compression Marker Output power: \(\text{CALC:MARKer:COMPression:POUT}\) \(\text{CompressionPout}\)
- New Search function: \(\text{CALC:MARKer:FUNCtion:SEL}\) \(\text{SearchCompressionPoint}\)
- New Execute function: \(\text{CALC:MARKer:FUNCtion:EXEC}\) \(\text{SearchCompressionPoint}\)

### Port Extensions
- Port Ext in distance: \(\text{SENS:CORR:EXT:PORT:DIST}\) \(\text{PortDistance}\)
- Set distance units: \(\text{SENS:CORR:EXT:PORT:UNIT}\) \(\text{PortDistanceUnit}\)
- Set Media per port: \(\text{SENS:CORR:EXT:PORT:MEDium}\) \(\text{PortMedium}\)
- Set waveguide cutoff freq per port: \(\text{SENS:CORR:EXT:PORT:WGCutoff}\) \(\text{PortWGCutoffFreq}\)
- Set Velocity Factor per port: \(\text{SENS:CORR:EXT:PORT:VELF}\) \(\text{PortVelocityFactor}\)
- Couple to system Velocity Factor: \(\text{SENS:CORR:EXT:PORT:SYSV}\) \(\text{PortCoupleToSystemVelocity}\)
- Couple to system Media type: \(\text{SENS:CORR:EXT:PORT:SYSM}\) \(\text{PortCoupleToSystemMedia}\)

### Electrical Delay
- Delay in distance: \(\text{CALC:CORR:EDELay:DISTance}\) \(\text{ElecDistanceDelay}\)
- Set units for distance: \(\text{CALC:CORR:EDELay:UNIT}\) \(\text{ElecDistanceDelayUnit}\)

---

The following are new programming commands for **PNA release 8.50** See What's New
### Phase Sweep

<table>
<thead>
<tr>
<th>Command</th>
<th>Command String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Phase sweep</td>
<td>CALC:FSIM:BAL:PHAS:SWE:STAT</td>
<td>PhaseSwpState</td>
</tr>
<tr>
<td>Start Phase port 1</td>
<td>CALC:FSIM:BAL:BPOR:SWE:PHAS:STAR</td>
<td>BalPort1 StartPhase</td>
</tr>
<tr>
<td>Stop Phase port 1</td>
<td>CALC:FSIM:BAL:BPOR:SWE:PHAS:STOP</td>
<td>BalPort1 StopPhase</td>
</tr>
<tr>
<td>Stop Phase port 2</td>
<td>CALC:FSIM:BAL:BPOR:SWE:PHAS:STOP</td>
<td>BalPort2 StopPhase</td>
</tr>
<tr>
<td>Enable as fixture offset</td>
<td>CALC:FSIM:BAL:FIXT:SWE:PHAS</td>
<td>PhaseSwpAsFixture</td>
</tr>
</tbody>
</table>

### Other

<table>
<thead>
<tr>
<th>Description</th>
<th>Command String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASYN argument added to CAQuire commands</td>
<td>SENS:CORR:CKIT:ECAL:ORI?</td>
<td>Learn more</td>
</tr>
<tr>
<td>Returns ECal orientation.</td>
<td>None</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release 8.35**  See What's New

### FIFO Data Buffer

<table>
<thead>
<tr>
<th>Action</th>
<th>Command String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIFO ON</td>
<td>OFF</td>
<td>SYST:FIFO[:STATe]</td>
</tr>
<tr>
<td>Read number of data points</td>
<td>SYST:FIFO:DATA:COUNt?</td>
<td>DataCount</td>
</tr>
<tr>
<td>Read data</td>
<td>SYST:FIFO:DATA?</td>
<td>Data</td>
</tr>
<tr>
<td>Read data compact form</td>
<td>None</td>
<td>DataInCompactForm</td>
</tr>
<tr>
<td>Clear data</td>
<td>SYST:FIFO:DATA:CLEAR</td>
<td>Clear</td>
</tr>
</tbody>
</table>

### Other N5264B Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Command String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FastCW</td>
<td>SENS:SWE:TYPE:FACW</td>
<td>FastCWPointCount</td>
</tr>
<tr>
<td>Enable Point Averaging</td>
<td>SENS:AVER:MODE</td>
<td>AverageMode</td>
</tr>
<tr>
<td>Enable Point Sweep</td>
<td>SENS:SWE:GEN:POINTsweep</td>
<td>PointSweepState</td>
</tr>
<tr>
<td>Set Trace Sweep</td>
<td>SENS:SWE:TRIG:MODE</td>
<td>Trigger Mode</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release 8.33**  See What's New
### Miscellaneous

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set display format units</td>
<td>Calc:Format:Unit</td>
<td>FormatUnit</td>
</tr>
<tr>
<td>Perform trace max</td>
<td>Disp:TMAX</td>
<td>TraceMax</td>
</tr>
<tr>
<td>Fast sweep mode</td>
<td>SENS:SWE:SPE</td>
<td>SweepSpeedMode</td>
</tr>
<tr>
<td>Launch Cal Wizard for apps (new behavior)</td>
<td>SYST:CORR:WIZ</td>
<td>LaunchCalWizard</td>
</tr>
<tr>
<td>Queries the TCP/IP port number for a TCP/IP socket connection.</td>
<td>SYST:COMM:TCPIP:CONT</td>
<td>None</td>
</tr>
<tr>
<td>Set CWFreq to Marker location</td>
<td>CALC:MARK:SET</td>
<td>SetCWFreq</td>
</tr>
<tr>
<td>Returns a list of channel numbers</td>
<td>SYST:CHAN:CAT?</td>
<td>Not new</td>
</tr>
<tr>
<td>Returns measurement numbers</td>
<td>SYST:MEAS:CAT?</td>
<td>Not new</td>
</tr>
<tr>
<td>Returns trace number</td>
<td>SYST:MEAS&lt;\text{n}&gt;:TRACe?</td>
<td>Not new</td>
</tr>
<tr>
<td>Returns Meas name</td>
<td>SYST:MEAS:NAME?</td>
<td>Not new</td>
</tr>
<tr>
<td>Return window number</td>
<td>SYST:MEAS&lt;\text{n}&gt;:WINDow?</td>
<td>Not new</td>
</tr>
<tr>
<td>Returns window numbers</td>
<td>SYST:WIND:CAT?</td>
<td>Not new</td>
</tr>
<tr>
<td>Same as calc:par:sel except takes a meas number</td>
<td>CALC:PAR:MNURM:[SEL]</td>
<td>Not new</td>
</tr>
<tr>
<td>Returns Limit line pass/fail status</td>
<td>CALC:LIMIT:FAIL?</td>
<td>Not new</td>
</tr>
<tr>
<td>Deletes the current limit line?</td>
<td>CALC:LIMIT:DATA:DELeTe</td>
<td>Not new</td>
</tr>
</tbody>
</table>

### Swept IMD

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a measurement</td>
<td>Calc:Custom:Define</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Set sweep type</td>
<td>SENS:IMD:SWE:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Set DeltaF</td>
<td>SENS:IMD:DFR</td>
<td>DeltaFrequency</td>
</tr>
<tr>
<td>Set center Freq</td>
<td>SENS:IMD:FREQ:FCEN</td>
<td>FrequencyCenter</td>
</tr>
<tr>
<td>Start for center freq sweep</td>
<td>SENS:IMD:FREQ:FCEN:STAR</td>
<td>FrequencyCenterStart</td>
</tr>
<tr>
<td>Stop for center freq sweep</td>
<td>SENS:IMD:FREQ:FCEN:STOP</td>
<td>FrequencyCenterStop</td>
</tr>
<tr>
<td>Center for center freq sweep</td>
<td>SENS:IMD:FREQ:FCEN:CENT</td>
<td>FrequencyCenterCenter</td>
</tr>
<tr>
<td>Span for center freq sweep</td>
<td>SENS:IMD:FREQ:FCEN:SPAN</td>
<td>FrequencyCenterSpan</td>
</tr>
</tbody>
</table>
Start for DeltaF sweep
SENSE:IMD:FREQ:DFR:STAR DeltaFrequencyStart
Stop for DeltaF sweep
SENSE:IMD:FREQ:DFR:STOP DeltaFrequencyStop
Set F1 for CW and Power sweep
SENSE:IMD:FREQ:F1 F1Frequency
Set F2 for CW and Power sweep
SENSE:IMD:FREQ:F2 F2Frequency
Set main tone IFBW
SENSE:IMD:IFBW:MAIN MainToneIFBandwidth
Set product tones IFBW
SENSE:IMD:IFBW:IMT IMToneIFBandwidth
Enables power coupling for F1 and F2
SENSE:IMD:TPOW:COUP CoupleTonePower
Set power level for F1 tone
SENSE:IMD:TPOW:F1 TonePower
Set power level for F2 tone
SENSE:IMD:TPOW:F2 TonePower
F1 start for power sweep
SENSE:IMD:TPOW:F1:STAR TonePowerStart
F1 stop for power sweep
SENSE:IMD:TPOW:F1:STOP TonePowerStop
F2 start for power sweep
SENSE:IMD:TPOW:F2:STAR TonePowerStart
F2 stop for power sweep
SENSE:IMD:TPOW:F2:STOP TonePowerStop
Read highest product allowed
None HighestOrderProduct

For CTB, CSO, and XMod parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normalization Mode</td>
<td>SENSE:IMD:NORM:MODE</td>
<td>CompositeNormalizationMode</td>
</tr>
<tr>
<td>Normalized CSO power</td>
<td>SENSE:IMD:CSO:NORM:POW</td>
<td>CompositeNormalizedCSOPower</td>
</tr>
<tr>
<td>CSO Offset</td>
<td>SENSE:IMD:CSO:OFFS</td>
<td>CSOOffset</td>
</tr>
<tr>
<td>CSO Number of Distortion products</td>
<td>SENSE:IMD:CSO:NDPR</td>
<td>CSONumDistortionProducts</td>
</tr>
<tr>
<td>Normalized CTB power</td>
<td>SENSE:IMD:CTB:NORM:POW</td>
<td>CompositeNormalizedCTBPower</td>
</tr>
<tr>
<td>CTB and XMod Number of carriers</td>
<td>SENSE:IMD:CTB:NCAR</td>
<td>CTBXMODNumCarriers</td>
</tr>
<tr>
<td>CTB Offset</td>
<td>SENSE:IMD:CTB:OFFS</td>
<td>CTBOffset</td>
</tr>
</tbody>
</table>

IMD Calibration

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Cal frequencies</td>
<td>SENSE:CORR:IMD:CAL:FREQ</td>
<td>CalibrationFrequencies</td>
</tr>
<tr>
<td>Max Products</td>
<td>SENSE:CORR:IMD:MPR</td>
<td>MaxProduct</td>
</tr>
<tr>
<td>Set power</td>
<td>SENSE:CORR:IMD:POW</td>
<td>PowerLevel</td>
</tr>
<tr>
<td>Sensor Cal Kit</td>
<td>SENSE:CORR:IMD:SENS.CKIT</td>
<td>PowerSensorCalKitType</td>
</tr>
<tr>
<td>Sensor connector</td>
<td>SENSE:CORR:IMD:SENS:CONN</td>
<td>PowerSensorConnectorType</td>
</tr>
</tbody>
</table>
The following are new programming commands for **PNA release 8.2**. See What's New

<table>
<thead>
<tr>
<th>iTMSA</th>
<th>CALC: FSIM:BAL:STIM:MOD</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Stimulus Mode</td>
<td></td>
<td>BalPort1PhaseOffset</td>
</tr>
<tr>
<td>Set Phase Offset</td>
<td>CALC: FSIM:BAL:BPOR:OFFS:PHAS</td>
<td>BalPort2PhaseOffset</td>
</tr>
<tr>
<td>Set Phase Offset as fixture</td>
<td>CALC: FSIM:BAL:FIXT:OFFS:PHAS</td>
<td>PhaseAsFixture</td>
</tr>
<tr>
<td>Set Source power for balanced ports</td>
<td>SOUR:POW</td>
<td>PowerAsFixture</td>
</tr>
<tr>
<td>Returns the number of source ports.</td>
<td>N/A</td>
<td>chan.SourcePortCount</td>
</tr>
<tr>
<td>Returns the string names of source ports.</td>
<td>N/A</td>
<td>chan.SourcePortNames</td>
</tr>
<tr>
<td>Returns the port number for the specified string port name.</td>
<td>N/A</td>
<td>chan.GetPortNumber</td>
</tr>
</tbody>
</table>

| Uncoupled Power Sweep | SOUR:POW:PORT:STARt     | StartPowerEx              |
| Set Start power for uncoupled power sweep |                          |                           |
| Set Start power for uncoupled power sweep | SOUR:POW:PORT:STOP      | StopPowerEx               |

| Choose FCA ports      | SENS:MIX:PMAP            | SetDutPorts               |
| Map PNA to DUT ports  |                          |                           |
| Read Input port number | SENS:MIX:PMAP:INP?       | DeviceInputPort           |
| Read Output port number | SENS:MIX:PMAP:OUTP?      | DeviceOutputPort          |
### LXI

<table>
<thead>
<tr>
<th>Command</th>
<th>Returns</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetIPConfigurationStruct</td>
<td>None</td>
<td>Structured status of the PNA networking</td>
</tr>
<tr>
<td></td>
<td></td>
<td>configuration.</td>
</tr>
<tr>
<td>GetIPConfiguration</td>
<td>None</td>
<td>Returns string status of the PNA networking</td>
</tr>
<tr>
<td></td>
<td></td>
<td>configuration.</td>
</tr>
<tr>
<td>LANConfigurationInitialize</td>
<td>None</td>
<td>Resets the PNA LAN configuration.</td>
</tr>
<tr>
<td>LANConfiguration</td>
<td></td>
<td>Modifies settings of the PNA networking</td>
</tr>
<tr>
<td></td>
<td></td>
<td>configuration.</td>
</tr>
<tr>
<td>LXIDeviceIDState</td>
<td>None</td>
<td>Displays the LAN Status dialog with LAN Status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Indicator showing IDENTIFY.</td>
</tr>
</tbody>
</table>

### Miscellaneous

<table>
<thead>
<tr>
<th>Command</th>
<th>Returns</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:PREF:DEF</td>
<td>cap.RestoreDefaults</td>
<td>Reset Preference Defaults</td>
</tr>
<tr>
<td>Sens:Class:Name?</td>
<td>Get_MeasurementClass</td>
<td>Returns the Measurement Class name</td>
</tr>
<tr>
<td>CALC:GCDAT:ITER</td>
<td>TotalIterations</td>
<td>GCA - Returns number of iterations required in a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SMART Sweep</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release 8.0**  See What's New
<table>
<thead>
<tr>
<th>Gain Compression Setup</th>
<th>Command Code(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of frequency points</td>
<td>SENS:GCS:SWE:FREQ:POIN</td>
</tr>
<tr>
<td>Number of power points</td>
<td>SENS:GCS:SWE:POW:POIN</td>
</tr>
<tr>
<td>Maximum number of points</td>
<td>None</td>
</tr>
<tr>
<td>Total number of points</td>
<td>None</td>
</tr>
<tr>
<td>Acquisition mode</td>
<td>SENS:GCS:AMOD</td>
</tr>
<tr>
<td>Smart tolerance</td>
<td>SENS:GCS:SMAR:TOL</td>
</tr>
<tr>
<td>Smart Iterations</td>
<td>SENS:GCS:SMAR:MIT</td>
</tr>
<tr>
<td>Smart settling time</td>
<td>SENS:GCS:SMAR:STIM</td>
</tr>
<tr>
<td>Smart show iterations</td>
<td>SENS:GCS:SMAR:SIT</td>
</tr>
<tr>
<td>Read compression failures</td>
<td>SENS:GCS:SFA?</td>
</tr>
<tr>
<td>Write port map</td>
<td>SENS:GCS:PORTM</td>
</tr>
<tr>
<td>Read Port Map (Input)</td>
<td>SENS:GCS:PORT</td>
</tr>
<tr>
<td>Read Port Map (Output)</td>
<td>SENS:GCS:PORT</td>
</tr>
<tr>
<td>End of Sweep</td>
<td>SENS:GCS:EOS</td>
</tr>
<tr>
<td>Linear input power</td>
<td>SENS:GCS:POW:LIN:INP:LEV</td>
</tr>
<tr>
<td>Reverse Power</td>
<td>SENS:GCS:POW:REV:LEV</td>
</tr>
<tr>
<td>Start power</td>
<td>SENS:GCS:POW:STAR:LEV</td>
</tr>
<tr>
<td>Stop power</td>
<td>SENS:GCS:POW:STOP:LEV</td>
</tr>
<tr>
<td>Compression algorithm</td>
<td>SENS:GCS:COMP:ALG</td>
</tr>
<tr>
<td>Compression Level</td>
<td>SENS:GCS:COMP:LEV</td>
</tr>
<tr>
<td>Backoff Level</td>
<td>SENS:GCS:COMP:BACK:LEV</td>
</tr>
<tr>
<td>X Delta</td>
<td>SENS:GCS:COMP:DELT:X</td>
</tr>
<tr>
<td>Y Delta</td>
<td>SENS:GCS:COMP:DELT:Y</td>
</tr>
<tr>
<td>Interpolation</td>
<td>SENS:GCS:COMP:INT</td>
</tr>
<tr>
<td>Safe Sweep enable</td>
<td>SENS:GCS:SAFE:ENAB</td>
</tr>
<tr>
<td>Safe Sweep coarse</td>
<td>SENS:GCS:SAFE:CPAD</td>
</tr>
<tr>
<td>Safe Sweep fine</td>
<td>SENS:GCS:SAFE:FPAD</td>
</tr>
<tr>
<td>Safe Sweep threshold</td>
<td>SENS:GCS:SAFE:FTHR</td>
</tr>
<tr>
<td>Read all GCA data</td>
<td>CALC:GCData:DATA</td>
</tr>
<tr>
<td>Read real GCA data</td>
<td>CALC:GCData:REAL</td>
</tr>
<tr>
<td>Read imaginary GCA data</td>
<td>CALC:GCData:IMAG</td>
</tr>
<tr>
<td>Read compression failures</td>
<td>NumberOfFrequencyPoints</td>
</tr>
<tr>
<td>Write port map</td>
<td>NumberOfPowerPoints</td>
</tr>
<tr>
<td>Read Port Map (Input)</td>
<td>MaximumNumberOfPoints</td>
</tr>
<tr>
<td>Read Port Map (Output)</td>
<td>TotalNumberOfPoints</td>
</tr>
<tr>
<td>End of Sweep</td>
<td>AcquisitionMode</td>
</tr>
<tr>
<td>Smart Sweep settling time</td>
<td>SmartSweepTolerance</td>
</tr>
<tr>
<td>Smart show iterations</td>
<td>SmartSweepMaximumIterations</td>
</tr>
<tr>
<td>Read compression failures</td>
<td>SearchFailures</td>
</tr>
<tr>
<td>Write port map</td>
<td>SetPortMap</td>
</tr>
<tr>
<td>Read Port Map (Input)</td>
<td>DeviceInputPort</td>
</tr>
<tr>
<td>Read Port Map (Output)</td>
<td>DeviceOutputPort</td>
</tr>
<tr>
<td>End of Sweep</td>
<td>EndOfSweepOperation</td>
</tr>
<tr>
<td>Linear input power</td>
<td>InputLinearPowerLevel</td>
</tr>
<tr>
<td>Reverse Power</td>
<td>ReverseLinearPowerLevel</td>
</tr>
<tr>
<td>Start power</td>
<td>chan.Start Power</td>
</tr>
<tr>
<td>Stop power</td>
<td>chan.Stop Power</td>
</tr>
<tr>
<td>Compression algorithm</td>
<td>CompressionAlgorithm</td>
</tr>
<tr>
<td>Compression Level</td>
<td>CompressionLevel</td>
</tr>
<tr>
<td>Backoff Level</td>
<td>CompressionBackoff</td>
</tr>
<tr>
<td>X Delta</td>
<td>CompressionDeltaX</td>
</tr>
<tr>
<td>Y Delta</td>
<td>CompressionDeltaY</td>
</tr>
<tr>
<td>Interpolation</td>
<td>CompressionInterpolation</td>
</tr>
<tr>
<td>Safe Sweep enable</td>
<td>SafeSweepEnable</td>
</tr>
<tr>
<td>Safe Sweep coarse</td>
<td>SafeSweepCoarsePowerAdjustment</td>
</tr>
<tr>
<td>Safe Sweep fine</td>
<td>SafeSweepFinePowerAdjustment</td>
</tr>
<tr>
<td>Safe Sweep threshold</td>
<td>SafeSweepFineThreshold</td>
</tr>
<tr>
<td>Read all GCA data</td>
<td>GetRaw2DData</td>
</tr>
<tr>
<td>Read real GCA data</td>
<td>GetDataIm</td>
</tr>
<tr>
<td>Read imaginary GCA data</td>
<td>GetDataRe</td>
</tr>
</tbody>
</table>
**Noise Figure Setup**

Create Noise figure meas

Sets the number of impedance states to use

Noise averaging ON and OFF

Set averaging of noise receiver.

Set bandwidth of noise receiver.

Set gain state of noise receiver.

Sets noise tuner identifier

Sets the port identifier of the ECal noise tuner that is connected to the PNA Source.

Sets the port identifier of the ECal noise tuner that is connected to the DUT.

Set the excess noise source ON or OFF.

Set mechanical switches

Sets the default setting for the Noise Tuner switch.

**Noise Figure Cal**

Create Noise Cal object

Set Noise Calibration method

Noise source ENR filename

Set noise source Cal Kit type

Set ambient temperature

Sets noise source connector type

Set Noise source temperature
### Noise Figure ENR File Data Management

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set ENR calibration data.</td>
<td>SENS:CORR:ENR:CAL:TABL:DATA</td>
<td>PutENRData</td>
</tr>
<tr>
<td>Read ENR calibration data.</td>
<td>SENS:CORR:ENR:CAL:TABL:DATA?</td>
<td>GetENRData</td>
</tr>
<tr>
<td>Get/set serial number of noise source.</td>
<td>SENS:CORR:ENR:CAL:TABL:SERial:DATA</td>
<td>ENRSN</td>
</tr>
<tr>
<td>Load ENR table from file.</td>
<td>MME:LOAD:ENR</td>
<td>LoadENRFile</td>
</tr>
<tr>
<td>Save ENR table to file.</td>
<td>MME:STORE:ENR</td>
<td>SaveENRFile</td>
</tr>
</tbody>
</table>

### Custom Cal Window

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turn ON</td>
<td>OFF Custom Cal window.</td>
<td>SENS:CORR:COLL:DISP:WIND</td>
</tr>
<tr>
<td>Show NO Custom Cal windows.</td>
<td>SENS:CORR:COLL:DISP:WIND:AOFF</td>
<td>DisplayOnlyCalWindowDuringCalAcquisition</td>
</tr>
<tr>
<td>Specify channel to sweep before Cal acquisition.</td>
<td>SENS:CORR:COLL:SWE:CHAN</td>
<td>AllowChannelToSweepDuringCalAcquisition</td>
</tr>
<tr>
<td>Sweep NO channel before Cal acquisition.</td>
<td>SENS:CORR:COLL:SWE:CHAN:AOFF</td>
<td>SweepOnlyCalChannelDuringCalAcquisition</td>
</tr>
<tr>
<td>Preview sweep before remote Cal acquisition.</td>
<td>SENS:CORR:COLL:GUID:PACQuire</td>
<td>SetupMeasurementsForStep</td>
</tr>
</tbody>
</table>

### Miscellaneous

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Trigger sweep mode</td>
<td>SENS:SW:E:TRIG:MODE</td>
<td>Trigger Mode</td>
</tr>
<tr>
<td>Copy a Cal Set</td>
<td>SENSE:CORR:CSET:COPY</td>
<td></td>
</tr>
</tbody>
</table>
COM versus SCPI

There are two methods you can use to remotely control the VNA: COM and SCPI. The following topics can help you choose the method that best meets your needs:

- Software Connection
- Physical Connection
- Programming Languages

Other Topics about COM Concepts

Software Connection

COM uses a binary protocol, allowing you to directly invoke a VNA feature. This is more efficient than SCPI. For example, the following statement calls directly into the VNA, executing the routine GetIDString.

```
PNA.GetIDString()
```

SCPI is a text based instrument language. To retrieve the ID string, you would send the following text string to the VNA:

```
IbWrite("*IDN?")
```

The VNA SCPI parser would first decode this text string to determine that the user has asked for the VNA to identify itself. Then the parser would call the COM method GetIDString().

The Physical Connection

Internal Control

With either COM or SCPI, the best throughput is attained by using the VNA's internal PC to execute your test code. However, if your test code uses too much system resources (CPU cycles and/or memory), this will slow the VNA's performance.

Using the SICL I/O Libraries, you can also connect to the VNA from a program running on the VNA.

External Control

You can control the VNA from a remote PC using either COM or SCPI.
**COM** - (Component Object Model) can be used to access any program like the VNA (835x.exe) or library (.dll) that exposes its features using a COM compliant object model. These programs or libraries are called "servers". Programs (like your remote program on your PC) that connect to and use the features of these servers are called "clients."

With COM, the server and the client do not need to reside on the same machine. DCOM, or distributed COM, makes the location of the server transparent to the client. When you access the VNA from a remote computer, you are using DCOM. In this case, the mechanical transport is a LAN (local area network).

However, using COM can add additional complexity:

- There are some DCOM security issues that may be a problem for you. Learn more.
- Using the default interface when compiling type libraries results in code that will only run with the latest firmware. Learn more.

**SCPI** - Using a GPIB interface card in a remote computer, you can connect to the instrument using a GPIB cable. There are some constraints on the length of this cable and the number of instruments that can be daisy-chained together.

Using the Keysight SICL I/O libraries, you can connect to the instrument over a LAN connection.

(LAN or INTERNAL) You can send SCPI commands using COM with the `ScpiStringParser` object.

If you have legacy code written in SCPI for another network analyzer, you may be able to leverage that code to control the VNA. However, the VNA uses a different platform than previous Keysight Network Analyzers. Therefore, not all commands have a direct replacement. See the VNA Code Translator Application.

**Programming Languages**

You can program the VNA with either COM or SCPI using several languages. The most common include:

**Keysight VEE** - With this language you can send text based SCPI commands and also use automation. VEE 6.0 or later is recommended.

**Visual Basic** - This language has great support for automation objects and can be used to drive SCPI commands. The use of VISA drivers for your GPIB hardware interface will make the task of sending SCPI commands easier.

**C++** - This language can do it all. It is not as easy to use as the above two, but more flexible.
Remotely Specifying a Source Port

In the 'not-too-distant past', it was a simple task to specify a VNA source port. It was either port 1 or port 2. Now, for the following reasons, it is not so simple:

- **Internal 2nd sources** are now offered on various VNA models. However, some source ports do not have a port number. One example is the second source on the PNA-X 2-port model (option 224). Learn more about Internal Second Sources.

- **External sources** can now be controlled by the VNA as though they are internal sources. External sources do not have a source port number, but use String names as identifiers.
  
  - **For FCA ONLY**: Once configured using the Configuration dialog, an external source can be selected remotely and controlled by the VNA by specifying the LOName using SCPI or COM.
  
  - **All other uses for External sources**: The external source must be configured and selected from the External Source dialog. You can then save an Instrument State file, then recall that state file remotely.

- **Multiport test sets**…choose between ports 1 through port N, where N is the number of ports on the test set. You still use a port number, but this port number refers to a logical port. The Port mapping feature maps the logical port to a physical port. Learn more about Multiport test sets.

- **iTMSA (Opt S93460A/B)** When this option is present, the string names for balanced source ports are returned with the appropriate COM and SCPI commands. For example, "SE Port 1" is used to access 'Single-ended Port 1".

Source Port String Names

The VNA User Interface (UI) makes it easy to configure and select the sources and ports. Remotely however, string names are used now, in addition to port numbers, to specify a Source port.

**COM** - The existing COM commands specify source ports as numbers and they are still used. It is necessary to learn the port number from the string using the GetPortNumber Method. Port numbers are assigned dynamically depending on whether external sources are selected and the number of ports of the VNA.

- **SourcePortNames Property**
- **GetPortNumber Method**
- **SourcePortCount Property**

An example:

```plaintext
dim app = 5700
```
set app = CreateObject("Agilentpna835x.application")
dim channel
set channel = app.Channel
dim portnum
portnum = Channel.GetPortNumber("Src2 Out1")
app.CreateMeasurement 1,"A",portnum

**SCPI** - ALL of the existing SCPI commands that specify a source port are extended to also allow the source port to be specified using string names. For example, send the following command to set the power on Src2 Out1:

- SOUR:POW 5, "Src2 Out1"
- Use Source:Cat? to list the available source port string names.
Shut Down the VNA Remotely

Shut down the VNA remotely by creating and then executing a macro.

Learn more about Macros in the VNA

Create the Macro

1. Click Utility, then Macro, then Macro Setup
2. In the Macro Setup dialog, select a blank line, then click Edit.
3. In the Edit Macro dialog:
   a. Under Title, enter Shutdown
   b. Under Macro Executable, enter Shutdown
   c. Under Macro run string parameters, enter -s -t 0 This will wait 0 seconds before executing the command.

Execute the Macro Remotely

Send the following SCPI command:

- `SYST:SHOR<n>:EXEC` where `<n>` is the number (formerly 'blank' line) of the macro that is stored in the VNA.

Variations

To see variations on this command, on any Windows computer:

- From any command prompt, enter `help shutdown`
The VNA has code emulators that allow you to control the VNA using test programs written in the remote control commands of the 8753. The code emulators process each of incoming 8753 commands in real time, by first recognizing it as a valid command, and then executing the VNA’s equivalent command(s) if one exist. The 8753 command emulation is performed by the CxL Application.

The CXL Application will also help translate your Legacy programs into VNA SCPI and COM commands.

You can launch these code emulators as follows:

1. Press **System > System Settings > Code Emulation**.
2. Select your required model. (To execute 8753, CXL application should be installed on the VNA.)

See the following document for operation.

- CxL Help from Windows Start menu

### Installing the CXL Application

Download the CXL application from [http://na.support.keysight.com/pna/cxl.html](http://na.support.keysight.com/pna/cxl.html). To install the CXL application, double click on the InstallShield package icon.

The CXL application has its own help file that includes a command cross-reference for 8510/8753/872x models to VNA commands.
Using Fixture Simulator

Fixture simulator allows user to mathematically add (embed) or remove (de-embed) circuits to, or from, the measurements. Learn more on Fixture Simulator.

In the new fixture generator with the firmware A.14.40 and above, multiple fixturing elements can be combined in any order, creating infinite combinations. New SCPIs are created with the concept of fixture “blocks’ or “circuits”.

In the legacy fixturing, a few specific features are tied to specific ports.

See Also

- Programming examples:
  - Create fixturing function (impedance conversion and port matching)
  - Create fixturing function (2-port deembed, port extension, port matching, impedance conversion)
  - Create fixuring function (port impedance conversion and port extension)

Concepts

There are two important concepts:

1. Fixtures are built from right to left, the circuit elements are added from right to left. There is no way to insert a block to the right of already existing blocks. And unlike legacy commands, there is not one SCPI command to change the order of the features. Instead, the blocks must be created in the order desired.

   Example of a fixture:
When building a fixture, the circuit elements are added from right to left. So in this example, the S2P blocks are added first, and then the S4P blocks.

**Note:** These blocks always exist and user cannot create them. In order to move them to the appropriate place in the fixture, first create the fixture elements to the right, then send the “calc:fsim:draft:extension:port[n]:end” command to move the port extension to the left-most side of the circuit sections. If more fixture elements are desired to be added to the left of the port extensions, create them after the “calc:fsim:draft:extension:port[n]:end” command is sent.

2. Fixture elements are built on a “draft” fixture and then “applied”. The process should be:
   a. First “discard” the draft fixture changes with **CALC:FSIM:DRAFT:DISCard** command. This copies the active fixture to the draft fixture.
   b. Then build fixture using new **Fsimulator Draft SCPI commands**.
   c. Lastly “apply” the draft fixture with **CALC:FSIM:APPLY**. This does a fixture computation (to ensure validity) and then the fixture is copied to the active fixture in the channel.

**New SCPI commands**

- **Fsimulator Draft SCPI commands, CALC:FSIM:DRAFT:XXX:YYY:ZZZ**: These commands are used to query and
set fixture values on a scratch fixture.

- **Fsimulator Active SCPI commands, CALC:FSIM:XXX:YYY:ZZZ:** These commands are used to query ONLY fixture values on the active fixture.
  
  - The only exceptions to these commands are:

  1. **CALC:FSIM:APPLY** which is same as legacy command
  2. **CALC:FSIM:POWer:PORT:COMPensate** commands because they are not fixture building commands, but rather specify if the fixture is being used in the channel (and how).

- **CALC:FSIM:APPLY:** Validate scratch fixture, copy to active fixture
- **CALC:FSIM:DRAFt:DISCard:** Discard scratch fixture changes by copying active fixture to scratch fixture.
- **CALC:FSIM:RESet:** Preset the fixture.

### Notes

- Legacy SCPI commands will continue to work, but they are incompatible with the new features. There will be an error if legacy SCPI commands are sent after new SCPI commands are sent or and/or after the new GUI is used to build the fixture.

- Blocks are created by default as "de-embed". User must send the **CALC:FSIM:DRAFt:CIRCuit[n]:EMBED:TYPE embed** command to switch to "embed" type.

- In many cases, « CALC:FSIM:DRAFt:CIRC » can replace the legacy-specific SCPI such as « CALC:FSIM:SEND:DEEM:PORT », but it is recommended to read the SCPI documentation linked above.

- To verify that the fixture has been properly created using SCPI, you can visualize the entire fixture by using the new Fixture GUI. Learn more about the Fixture generator dialog box.

- User also can create .topo or .snp file by GUI, and recall the file from SCPI. Learn more on Fixture generator dialog box.

### Simple Sample Program

:CALC1:FSIM:DRAFt:CIRCuit:RESET
! Port Extension

:CALC1:FSIM:DRAFt:EXTension:PORT1:DElay 180E-12
:CALC1:FSIM:DRAFt:EXTension:PORT1:STATe ON

:CALC1:FSIM:DRAFt:EXTension:PORT2:STATe ON

:CALC1:FSIM:DRAFt:SECTION:EXTension:ENABLE ON

! Port Matching for port 1

:CALC1:FSIM:DRAFt:CIRCUit1:ADD SLPC,2
:CALC1:FSIM:DRAFt:CIRCUit1:VNA:PORTs 1

:CALC1:FSIM:DRAFt:CIRCUit1:EMBED:TYPE EMBED
:CALC1:FSIM:DRAFt:CIRCUit1:STATe ON

:CALC1:FSIM:DRAFt:CIRCUit1:PAR:L 3E-9
:CALC1:FSIM:DRAFt:CIRCUit1:PAR:R 0
:CALC1:FSIM:DRAFt:CIRCUit1:PAR:C 0
:CALC1:FSIM:DRAFt:CIRCUit1:PAR:G 0

! Port Matching for port 2

:CALC1:FSIM:DRAFt:CIRCUit2:ADD SLPC,2
:CALC1:FSIM:DRAFt:CIRCUit2:VNA:PORTs 2

:CALC1:FSIM:DRAFt:CIRCUit2:EMBED:TYPE EMBED
:CALC1:FSIM:DRAFt:CIRCUit2:STATe ON

:CALC1:FSIM:DRAFt:CIRCUit2:PAR:L 2E-9
:CALC1:FSIM:DRAFt:CIRCUit2:PAR:R 0
:CALC1:FSIM:DRAFt:CIRCUit2:PAR:C 0
:CALC1:FSIM:DRAFt:CIRCUit2:PAR:G 0

! Apply the circuit from draft to active.
:CALC1:FSIM:APPLY
:CALC1:FSIM:STATe ON
Active Hot Parameters

This application measures the optimum Gamma that provides the maximum delivered power for nonlinear devices. Unlike classic S-parameters, which capture only linear device behavior and ignore nonlinear behavior, X-parameter terms (Xf, Xs, Xt) are used to predict the Gamma that yields the optimum power delivered to a load from a nearly matched device under large-signal drive and thus define the nonlinear behavior of the DUT.

The X-parameter terms assume the following:

- Fundamental frequency (harmonics ignored)
- Amplifier is nearly matched to 50 ohms. The spectrally linearized X-parameters (Xs and Xt terms) are sufficient for characterizing the effects of incident A2 waves while the nonlinear component is driven with a large A1 signal at the input port.

In this topic:

- Features and Requirements
- Measurement Parameters
- How to Make Active Parameter Measurements
  - Create an Active Parameter Channel
  - Active Parameter Sweep tab
  - Active Parameter RF Path tab
  - Active Parameter X-axis tab
  - Active Parameter Analysis dialog
- Accessing Measurement Parameters
- Calibrating an Active Parameter Channel
- Saving Data

Features and Requirements

Features

- Proper corrections for matched amplifiers operating in nonlinear (Compression, Saturation) region by using fundamental X-Parameters measurements (XS, XT, XF) of the DUT.
- Ensures system-to-system correlation across different test stands.
- Provides fast measurements of the Hot S22, Hot S21 of the DUT.
● Easy calibration and measurement setup. Comb generator is not needed for calibration.

● Provides measurements of the optimum match for maximum delivered power and the value of the maximum power, as well as, power delivered to 50 ohms.

● Uses X-parameter technology to solve the non-linear X-parameter equation (restricted to 50 GHz for option S93111A/B):

\[
\begin{align*}
b_1 &= XF_{11}(|a_1|)e^{i\varphi(a_1)} + XS_{12}(|a_1|)a_2 + XT_{12}(|a_1|)e^{i2\varphi(a_1)}\text{conjugate}(a_2) \\
b_2 &= XF_{21}(|a_1|)e^{i\varphi(a_1)} + XS_{22}(|a_1|)a_2 + XT_{22}(|a_1|)e^{i2\varphi(a_1)}\text{conjugate}(a_2)
\end{align*}
\]

For option S93111A/B, characterization above 50 GHz uses active source injection at device input and output to solve the traditional linear S-parameter equation:

\[
\begin{align*}
b_1 &= S_{11}a_1 + S_{12}a_2 \\
b_2 &= S_{21}a_1 + S_{22}a_2
\end{align*}
\]

Requirements

● Active Hot Parameter software for “B” model PNA-X (N524xB)
  o S93110A/B, Active Hot Parameters
  o S93111A/B, Active Hot Parameters, restricted to 50 GHz
  ● 4-Port PNA-X with 2 sources

Measurement Parameters

● IPwr, Input power - incident power \(|a_1|^2\).
● OPwr, Output Power - output power delivered into Z0.
● HotS11, Active input match \(|XF_{11}(|a_1|)/|a_1|\).
● HotS21, Active Gain – Non-linear gain measured in Z0 \(|XF_{21}(|a_1|)/|a_1|\).
● HotS12, Active Reverse Isolation \((XS_{12})\).
● HotS22, Active Output Match – Conjugate of the optimum load, Gamma.
● Gamma, Optimum load for maximum delivered power.
● Pmax, Maximum delivered power.
● Xs(2,2), Direct Reflection Coefficient. The fundamental XS22 parameter which is used, in combination with XT22, to describe the change in b2 as a function of a2, the signal reflected from the load.
● Xt(2,2), Direct Reflection Coefficient. The fundamental XT22 parameter which is used, in combination with XS22 to describe the change in b2 as a function of a2, the signal reflected from the load.
● Xf(2,1), Output Coefficient. The fundamental XF21 parameter.
- **DeltaOPwr**, Delta between the output wave and output power to Z0 load.
- **Option S9311A/B Only**
  - **HotGain**, Forward Gain using output/input linear S-parameter measurements.
  - **HotMatch**, Output Match using linear S-parameter measurements.
  - The following traces are restricted to 50 GHz: OPwr, HotS11, HotS21, HotS12, HotS22, Gamma, Pmax, Xs(), Xt(), Xf(), DeltaOPwr.

### How to Make Active Parameter Measurements

#### Create an Active Parameter Channel

1. On the VNA front panel, press **Meas > Main > Meas Class**.
2. Select **Active Parameters**, then either:
   - **OK** to delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.
3. An **Active Parameter** measurement is displayed.

#### Configure Active Parameter settings

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Freq &gt; Main &gt; AHP Setup</strong></td>
<td>1. Click <strong>Stimulus</strong></td>
</tr>
<tr>
<td></td>
<td>2. Click <strong>Frequency</strong> then select <strong>ACTS Frequency</strong>...</td>
</tr>
</tbody>
</table>

**Active Parameter Sweep** tab help
Sweep Type - Sets the sweep type to the active channel:

**Linear Frequency**  Sets a linear frequency sweep that is displayed on a standard grid with ten equal horizontal divisions.

**Log Frequency**  The source is stepped in logarithmic increments and the data is displayed on a logarithmic x-axis. This is usually slower than a continuous sweep with the same number of points.

**Power Sweep**  Activates a power sweep at a single frequency that you specify. Learn about power sweep.

**Multisweep**  Selects multiple sweep. This is also called 3D sweep (power, phase, and frequency). Each sweep can be set to a different power level. Within each sweep is a 2D sweep of phase and frequency. With a fixed input power, a frequency-swept trace is useful to analyze the DUT’s frequency response. With fixed input power and fixed frequency, offset-phase measurements are performed to get adequate values to compute the “active S22” and X parameters.

**Linear/Log Frequency Sweep Type**
Source

- **Start Frequency**  Sets the beginning value of the frequency sweep.
- **Center Frequency**  Sets the center frequency of the frequency sweep.
- **Stop Frequency**  Sets the end value of the frequency sweep.
- **Frequency Span**  Sets the frequency span of the frequency sweep.
- **Number of Frequencies**  Sets the number of data points that the analyzer measures during a sweep. Range: 2 to 20001. (Default is 201).
- **Power Level**  Sets the power level of the source. Learn more.

Power Sweep Sweep Type

Source

- **Start Power**  Sets the beginning value of the power sweep.
- **Stop Power**  Sets the end value of the power sweep.
- **Number Of Powers**  Sets the number of data points that the analyzer measures during a sweep. Range: 2 to 20001. (Default is 201).
- **CW Frequency**  Sets the single frequency where the analyzer remains during the measurement sweep.

Multisweep Sweep Type

Source

- **Start Frequency**  Sets the start frequency of the frequency sweep.
- **Center Frequency**  Sets the center frequency of the frequency sweep.
- **Stop Frequency**  Sets the stop frequency of the frequency sweep.
- **Frequency Span**  Sets the frequency span of the frequency sweep.
- **Frequency Sweep**  Sets the sweep type.
- **Number of Frequencies**  Sets the number of data points that the analyzer measures during a frequency sweep. Range: 2 to 20001. (Default is 201).
- **Start Power**  Sets the beginning value of the power sweep.
- **Stop Power**  Sets the end value of the power sweep.
- **Number of Powers**  Sets the number of data points that the analyzer measures during a power sweep. Range: 2 to 20001. (Default is 201).

Extraction

- **IF Bandwidth**  Sets the IF (Receiver) bandwidth. Learn more.
- **Absolute Power**  The tone power is an absolute power.
• **Relative to Input Power**  The tone power is relative to the input power (dBc).

**Extraction Tone (ET) Level**

The level of the extraction tone must be small enough to ensure a spectrally-linear response, but large enough to ensure that the response is measurable.

A Source Power Cal is performed and both the LSOP (large signal output power) and Extraction tones power levels are adjusted to compensate for loss, external amplifiers, and attenuators.

**Select the method used to set the ET Power Level:**

The goal of both the Absolute and Relative ET Level methods is to set the ET level ~16 dB below the LSOP level while ensuring a spectrally-linear response from the DUT.

- **Relative to the Input Power**
  For example: At the DUT output you know that the DUT gain at the maximum stimulus power is 12 dB. The extraction level is set to -4 dBc (DUT Gain – 16 dB).

- **Absolute Power**
  For example: You know that the DUT gain is 12 dB with the stimulus power at -10 dBm. Then the DUT Max Output power is +2 dBm. The extraction level is set to 16 dB below +2, or -14 dBm (Input Stimulus + DUT Gain – 16 dB).

**Number of Phases**  Sets the number of phase points. At each point, a phase sweep is performed. Several stimulus states are required to extract the unknown parameters. The following is an example of the output power in Polar format with the number of phases set to 21. Note that the oval shown below is due to the non-linearity of the device. The perfect circle shown below is a linear device.
DC

- DC Sources...

**Active Parameter RF Path** tab help

**Power On - all channels** - Check to enable source power for all channels.

**Source**

- **Source Port** Selects the port through which an internal source is available.
- **Source Attenuator** Sets the attenuation of the source.
- **Receiver A Attenuator** Sets the Receiver attenuation.
- **ALC Hardware**
  - **Internal** - ALC leveling. Power level within an attenuator setting is limited to the ALC Range. See **Source Unleveled**.
Open Loop - No ALC and NO Receiver Leveling. Learn more.

- **Receiver Leveling** Turns receiver leveling on/off. This function adjusts the source power until the measured receiver power is equal to the Port Power.

**Extraction**

- **Extraction Port** Selects the port to measure the output of the DUT.
- **Source Attenuator** Sets the attenuation of the source.
- **Receiver C Attenuator** Sets the Receiver attenuation.
- **ALC Hardware**
  - **Internal** - ALC leveling. Power level within an attenuator setting is limited to the ALC Range. See Source Unleveled.
  - **Open Loop** - No ALC and NO Receiver Leveling. Learn more.

- **Receiver Leveling** Turns receiver leveling on/off. This function adjusts the source power until the measured receiver power is equal to the Port Power.

**RF Path Config...**
This tab allows the user to select a subset of the data to be displayed on the screen (frequency, power, phase) by defining the x-axis and a set of fixed values for the other parameters.

**Active**  Selects a trace to be active. A trace must be active before its trace settings can be changed.

**Select all traces**  Check to select all traces.

**X-axis display**  Sets the x-axis sweep type.

  **Frequency**

  - **Fixed Parameters**

    - **Input Power**  Sets a fixed input power level.
    - **Enable interpolation**  Check to enable displayed data to be computed by interpolation.

  **Power**

  - **Fixed Parameters**
- **Frequency** Sets a fixed frequency.
- **Enable interpolation** Check to enable displayed data to be computed by interpolation.

**Phase**

- **Fixed Parameters**
  
  - **Input Power** Sets a fixed input power level.
  
  - **Frequency** Sets a fixed frequency.
  
  - **Enable interpolation** Check to enable displayed data to be computed by interpolation.

**DC**

- **Fixed Parameters**
  
  - **Input Power** Sets a fixed input power level.
  
  - **Enable interpolation** Check to enable displayed data to be computed by interpolation.

- **Name**
  
  - **AO1** Internal DC source #1.
  
  - **AO2** Internal DC source #2.
This dialog is identical to the X-axis tab above.

Access this dialog by right-clicking in the trace status area then selecting Active Parameter Analysis...:

---

Accessing Measurement Parameters

### How to select and configure Measurement Parameters

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Select a trace by pressing <strong>Trace &gt; Trace N &gt; Trace N.</strong></td>
<td>1. Right-click on a trace.</td>
</tr>
<tr>
<td>2. Press <strong>Trace &gt; Trace Setup &gt; Measure...</strong></td>
<td>2. Select <strong>Measure Tr N.</strong></td>
</tr>
<tr>
<td>3. Select a parameter.</td>
<td>3. Select a parameter.</td>
</tr>
</tbody>
</table>

---

**Active Hot Parameters help**

**Option S93110A/B Measurement Parameters**

**Note:** In the example below, Port 1 is the Source Port (DUT input) and Port 3 is the Extraction Port (DUT output). Port 3 or Port 2 can be chosen as the output of the DUT.
Option S9311A/B Measurement Parameters
IPwr  Check to measure input power.

OPwr  Check to measure output power.

HotS31  Check to measure Active Gain.

HotS33  Check to measure Active Output Match.

HotS13  Check to measure Reverse Isolation.

HotS11  Check to measure Active Input Match.

Gamma  Check to measure optimized Gamma.

Pmax  Check to measure maximum delivered power.

Xs(3,3), Direct Reflection Coefficient. The fundamental $XS_{33}$ parameter which is used, in combination with $XT_{22}$, to describe the change in $b_2$ as a function of $a_2$, the signal reflected from the load.

Xt(3,3), Direct Reflection Coefficient. The fundamental $XT_{33}$ parameter which is used, in combination with $XS_{22}$ to describe the change in $b_2$ as a function of $a_2$, the signal reflected from the load.
**Xf(3,1),** Output Coefficient. The fundamental $XF_{31}$ parameter.

**DeltaOPwr**  Check to measure the delta between the output wave and output power to Z0 load.

Available for option S9311A/B only (the following traces are not restricted to 50 GHz)

**HotGain**  Check to measure forward gain using linear S-parameters output/input.

**HotMatch**  Check to measure active output match using linear S-parameters.

**Buttons**

**Select All**  Click to select all measurement parameters.

**Clear All**  Click to deselect all measurement parameters.

**Create in New Window**  Check to create a new trace in a new window for the measurement.

**Channel Number**  Select a channel number for the measurement.

---

**Calibrating an Active Parameter Channel**

A calibration can be performed on the Active Parameter Channel using the Cal All Wizard.

**Saving Data**

The data can be saved to a *.csv file with a column for each measurement parameter.
### How to save data

**Using Hardkey/SoftTab/Softkey**
1. Press Save Recall > Save Other > Save Data....
2. Save as type = HotS22 Sweep Data (*.csv)

**Using a mouse**
1. Click File
2. Click Save Data....
3. Save as type = HotS22 Sweep Data (*.csv)
Fast CW Mode (Opt S93118A/B) and other Antenna Features

The following features, used with the PNA-X models and the N5264B, were designed specifically for Antenna applications.

- **Fast CW Mode Features (Opt S93118A/B)**
  - FIFO Buffer and Fast CW
  - FIFO Buffer and Fast Groups
  - FIFO Buffer and Fast Segments

- **Other Useful Antenna Features**
  - Point Averaging
  - Point Sweep
  - Trace Triggering

- **See Also**
  - N5264B
  - Pulsed Measurements
  - Frequency (Security) Blanking
  - External Triggering

### Fast CW Mode Features (Opt S93118A/B)

The following Fast Sweep features allow you to **very quickly** measure and download data to a remote computer.

- **Fast CW, Fast Groups, and Fast CW Segments** all work ONLY with the FIFO Data Buffer.

- These features can be used ONLY with **SCPI or COM commands**. COM is faster than SCPI when using `DataInCompactForm`. Otherwise, SCPI is faster than COM.
The FIFO Data Buffer

The FIFO (First-IN, First-OUT) data buffer is a circular buffer that allows very fast Read-Write access.

- 5 GB FIFO buffer file on the E:\ drive. If the "not enough disk space" error message is displayed, clear all unnecessary files on the E:\ drive.
- You can write to, and simultaneously read from, the FIFO buffer.
- A maximum of 1 million data points can be read for each query.
- REAL / IMAGINARY pairs is the ONLY supported format for the FIFO buffer.
- A preset or instrument state recall will turn off the FIFO buffer collection.
- When more than one measurement is present, data from each measurement is stored in the FIFO buffer in the following order. These measurements are separated into lines for easier reading.

4-port models

- \( R, A, B, C, D, \)
- \( R/A, A/A, B/A, C/A, D/A, \)
- \( R/B, A/B, B/B, C/B, D/B, \)
- \( R/C, A/C, B/C, C/C, D/C, \)
- \( R/D, A/D, B/D, C/D, D/D \)

2-port models

- \( R1, R2, A, B, \)
- \( R1/R1, R2/R1, A/R1, B/R1 \)
- \( R1/R2, R2/R2, A/R2, B/R2 \)
- \( R1/A, R2/A, A/A, B/A \)
- \( R1/B, R2/B, A/B, B/B \)

S-parameters are pre-defined, ratioed-receiver measurements. Learn more. S-parameters are placed in the FIFO in order based on their underlying receivers. For example, S21 is placed into the FIFO in the same manner as B/R1.
Fast CW

In Fast CW mode the VNA display is not updated. There is no background computation or other 'interference' from the VNA computer. Therefore, data is acquired real-time.

The following **requirements** must be met **before** sending the Fast CW command.

- FIFO is ON
- A single channel is being measured. Other channels can be in Hold.
- All measurements are acquired in a single sweep.
- The channel must be in CW mode (start frequency = stop frequency or sweep type = CW Time).

**IMPORTANT - Fast CW and IF Bandwidth setting**

- IF Bandwidth of **10 kHz and lower** - Data is transferred immediately to the FIFO after every acquisition.
- IF Bandwidths **greater than 10 kHz** - Data is transferred to the FIFO in groups. A triggered acquisition is **NOT** placed into the FIFO buffer until either the total number of points is completed, or an intermediate group of points is finished. The number of points within a group differs for each IF Bandwidth setting.

**Notes:**

- See example programs in SCPI and COM.
- Fast CW sets the number of data points, overwriting the standard channel setting.
- 400,000 points/second applies to 600 KHz at a CW frequency with no point triggering. For narrower bandwidths the speed is slower.
- Frequency switching time will reduce the speed. The frequency switching can be anywhere from 50 uS to 1 mS depending on the frequencies. The CW segment itself can go at 400,000 points per second.
- All 5 receivers can do 400,000 point/second at the same time. So the data comes out at 5 times that rate, i.e. 2 Mp/s.
- When exiting Fast CW, the FIFO data buffer is cleared.
- External measurement trigger signals are allowed only on Meas Trigger.
- The Aux 1 and Aux 2 triggers cannot be used per point in FastCW mode.
- Aux 1 and Aux 2 triggers can still be used for external source hardware triggering, but the external sources must be in CW mode.
An error message appears if triggering is sent to the VNA faster than it can respond.

**Fast Groups with FIFO Data Buffer**

With this speed optimization feature, interaction with Windows or other VNA 'overhead' calls are suspended, allowing very fast and predictable measurement timing.

Fast Groups is automatically enabled when the following **requirements** are met:

- FIFO is ON
- A single channel is being measured. Other channels can be in Hold.
- All measurements are acquired in a single sweep.
- Group trigger is enabled with count > 1.

**Notes:**

- Fast CW can **NOT** be used with Fast Groups.
- Fast Groups and Fast CW Segments were designed to be used together, but not required.
- The FIFO Tester example program demonstrates this feature.

**Fast CW Segments with FIFO Buffer**

In this optimization feature, each CW segment (where the start and stop frequency is identical) within a channel is measured at speeds as fast as the Fast CW mode sweep.

Fast CW Segments is automatically enabled when the following **requirements** are met:

- FIFO is ON
- Start and stop frequency of a segment is identical.
- External measurement trigger signals are allowed only on Meas Trigger.
- The Aux 1 and Aux 2 triggers cannot be used per point in Fast CW Segment.
- Aux 1 and Aux 2 triggers can be used for external source hardware triggering.

**Notes:**

- Fast CW can **NOT** be used with Fast CW Segments.
- The sweep can include non-CW segments, but these are not acquired in Fast mode.

- Fast Groups and Fast CW Segments were designed to be used together, but not required.

- The FIFO Tester example program demonstrates this feature.

- In Fast CW Segments, when data is not being acquired in real-time, the following message appears: Caution: Sweep time jitter. Try reducing the number of segments. To avoid this error, reduce the number of segments in the channel.

**Other Antenna Features**

**Point Averaging**

This feature is selected on the Average dialog.

When selected, each data point is measured the specified number of averages before stepping to the next data point. When point trigger is selected, only one trigger is required for each data point regardless of the number of averages.

**Point Sweep**

This feature is selected on the Sweep Setup dialog.

In Point Sweep mode, the VNA measures both the forward and reverse parameters at each frequency point before stepping to the next frequency. The display trace is updated as each data point is measured. Point sweep is the same as stepped sweep mode of the 8510 and 8530.

**Trace Triggering**

This feature is selected under Trigger Mode on the Trigger dialog.

Available ONLY when Point Sweep is selected. Each trigger signal causes two identical measurements to be triggered separately - one trigger signal is required for each measurement. Other trigger mode settings cause two identical parameters to be measured simultaneously.

Trace triggering is NOT permitted when a channel is using a 2 port (or more) S-Parameter calibration.

**See Also**

- Pulsed Measurements
- Frequency (Security) Blanking
- External Triggering
Differential I/Q is very flexible, allowing a variety of phase and frequency settings and measurements.

In this topic:

- Features, Requirements, and Limitations
- Differential I/Q Setup Overview
- How to make Differential I/Q Measurements
- Setup Dialog
- Frequency Range Settings Dialog
- Source Configuration Dialog
- Define a Measurement Parameter
- Measure Dialog
- Edit Parameters Dialog
- Select X-Axis
- Calibration
- DIQ Examples (Separate topics)
  - Single-Ended IQ Upconverter / Modulator
  - Differential I/Q Upconverter
  - Differential Mixer (single-ended LO)
  - Differential Amplifier Harmonic Distortion and THD
  - Differential Amplifier IMD
  - Mixer Spurs (Single Ended)
  - DIQ trace parameter example *.xml files are stored on the VNA.

If the VNA is running the Windows 7 operating system, files are stored at:
C:\Users\Public\Documents\Network Analyzer\Samples\Setups\DiffIQ

If the VNA is running the Windows 10 operating system, files are stored at:
D:\Samples\Setups\DiffIQ

See Also

Programming commands
Features, Requirements, and Limitations

Features

Combines and extends the capabilities of ALL three of these features:

- **Source Phase Control** (Option S9x088A/B) is limited to 2-ports, only 4 receivers - Diff IQ can measure ALL receivers.
- **External sources** (control at the same time as internal sources).
- **FOM** (Frequency Offset Mode - Opt S9x080A)

This allows you to set an internal or external source to any frequency range, any power level, and any phase relative to a defined reference source. Then, measure any receiver over any frequency range independent of the source frequency range(s), with optional match correction. Display the results of any receiver combination using mathematical expressions available with the **Equation Editor** feature.

The magnitude and phase of a controlled source is set relative to a reference source using a pair of receivers (one for the reference source and one for the controlled source), in an iterative loop. The controlled source is adjusted within the loop while measuring the ratio of the two receivers. The loop continues until either the desired magnitude and phase offset is achieved (within user-settable tolerance values), or the maximum number of iterations has occurred.

When controlling an external source, a portion of the output signal must be routed to an internal receiver (usually one of the reference receivers), via a splitter or directional coupler and one of the front- or rear-panel access points. The source can be controlled via any of the available remote interfaces, such as GPIB, USB, or LAN. The 10 MHz frequency reference should be shared between the source and network analyzer.

Requirements

- **Frequency Offset Mode - Opt S9x080A**
- **Available ONLY on 4-port N522xB and N524xB (PNA-X) models.**

Limitations

The following Features are **NOT** Available in a Differential IQ channel:
Differential I/Q Setup Overview

1. Define **frequency ranges** for sources and receivers.
   - Ranges can be fixed or swept frequency.
   - To accomplish necessary measurements, receivers can be tuned to an arbitrary list of frequencies independent of those used for the sources. Define **controlled sources** for frequency, power, and phase relationships to other sources, using ranges from

2. Define **trace parameters**.
   - Parameters can use single receivers or multiple receivers combined with mathematical operators.
   - Parameters can combine receiver measurements made at different frequencies.

3. Define **x-axis** (can be frequency, power, phase, DC, points).

4. **Calibrate** all channels using the Cal All wizard, and if needed for additional direct-receiver response calibrations, the Cal Plan Manager wizard.

How to make Differential I/Q Measurements

**Create a Differential I/Q Channel**

1. On the VNA, press **Meas > S-Param > Meas Class...**

2. Select **Differential I/Q**, then either:
OK delete the existing measurement, or

New Channel to create the measurement in a new channel.

3. A **Differential I/Q measurement** is displayed.

Configure Differential I/Q settings

Using **Hardkey/SoftTab/Softkey**

1. Press **Setup > Main > DIQ Setup...**

Using a mouse

1. Click **Stimulus**.
2. Select **DIQ Setup...**

**Differential I/Q Setup Dialog help**

**Frequency Range** (See **Example configurations**)

Sets the source and receiver frequency range. Each defined frequency range can be measured by all of the VNA receivers. The **Define Parameters** dialog allows you to configure the measurements
to be displayed.

Only Linear sweeps are allowed.

**To make changes**, click the **Range Name** to be changed, then click **Edit** to start the **Frequency Range dialog**.

**New** - Creates a new frequency range.

**Remove** - Deletes a range. This also removes all measurement parameters that use the deleted range.

**Save...** - Saves the measurement setup.

**Load...** - Loads a measurement setup.

**Sources**

**Source Name** - Lists the test ports through which an internal source is available. If an external source has been configured, it will appear at the bottom of the list.

**To make changes**, click the **Source Name** to be changed, then click **Edit** to start the **Source Configuration dialog**.

**Add Source** - Starts the **External Source Configuration** dialog.

**Power ON (All Channels)** - Turns power on to all channels.
To start this dialog, click the **range name** to be changed, then click **Edit**, in the above **Differential IQ Setup dialog**.

**Note:** The Number of Points is always the same for ALL defined frequency ranges. Press **Sweep**, then **Number of Points**

This dialog sets a frequency range for the source and receiver. The frequency range will be measured by ALL of the receivers.

**F1** can NOT be coupled to another range.

**IFBW** - Set the IFBW to be used by the receivers for the specified frequency range.

**Note:** In Diff IQ mode, the IFBW in the VNA status bar may not reflect the channel’s actual IFBW, which can be set independently for each frequency range in the Setup dialog. While the user can modify the status bar IFBW from the status bar, it is not applied to the measurement.

**Procedure**

Either set specific frequencies for the selected range:

- Enter **Start** and **Stop** frequencies or select **Center** and **Span**.
- For a CW frequency, set the **Start = Stop** or **Span = 0**.
Or set frequencies that are based on another already-created range:

- Check **Couple to**, then select the range to base the selected range on.
- Optionally select any of the following:
  - Select a predefined Offset frequency range.
  - **Check Up** to ADD the Offset range to the Couple To range.
  - **Clear Up** to SUBTRACT the Offset range from the Couple To range.
  - Select a **Multiplier** or **Divisor** value.

**How the selected range is calculated**

For the following frequency ranges (as seen in above dialog):

- **Selected (Output) = F3**
- **Couple to = F1**
- **Offset = F2**

The following formula applies for each data point in each range:

\[
F3 = F1 \times \text{Multiplier} / \text{Divisor} + F2
\]

**Note:** The Number of Points is always the same for ALL defined frequency ranges. Press **Sweep**, then **Number of Points**

**Examples**

For **Mixers / Frequency Converters** (See above dialogs)

- F1 = RF or Input Frequencies: Enter Start and Stop Frequency range
- F2 = LO Frequency: Enter CW Frequency (Start = Stop)
- F3 = Couple to F1; Offset = F2

For **Amplifier Harmonics**
F1 = Enter fundamental Start and Stop Frequency range

F2 = Couple to F1; no Offset; Multiplier= 2 (Measures 2nd Harmonic)

F3 = Couple to F1; no Offset; Multiplier= 3 (Measures 3rd Harmonic)

Source Configuration Dialog help

This dialog is started by clicking the Source Name to be changed, then click Edit, in the Differential IQ Setup dialog.

Configures the source for the specified port. You can set or sweep Power, Phase, and perform Match Correction for each source port.

Source State: Choose from the following:

Auto-on - Source power is turned ON at the specified test port when required by the
measurement. This is the most common (default) setting. Auto-on sources are turned OFF when other sources are performing Match Correction sweeps.

**Always On** - Source power is ALWAYS ON, regardless of measurements that are in process. Use this setting to supply source power to a DUT port that always requires power, such as an LO port. This could turn OFF power at another test port. [Learn about internal second source restrictions.](#)

**Off** - Source power is turned off unless needed for match correction at a given port. Use this setting to prevent damage to a sensitive DUT test port.

**Uncontrolled** - (External Sources ONLY). The source state and other settings are NOT changed. Match correction can NOT be performed while uncontrolled.

**Frequency Range** - Selects a frequency range defined in the Differential IQ Setup dialog.

**Ext Source Port** Selects the source port that the external source is routed through. Displayed when an external source is selected as the active port.

**Power**

**Sweep Power** - Check to perform a power sweep.

**Start / Stop Power** - Set both the Start and Stop power levels for power sweep. Set a single (Start) power level for the specified source port.

**Leveling Mode:** Choose from the following:

- **Internal** - ALC leveling loop ONLY; NO Receiver Leveling.
- **Internal R<n>** - Both ALC and Receiver Leveling - where <n> is the reference port (not shown in diagram). To change the reference port, click [Power and Attenuators](#), then Receiver Leveling. [Learn more.](#)
- **Open Loop** - No ALC and NO Receiver Leveling - Used during pulse conditions with the internal source modulators. No leveling is used in setting the source power. The lowest settable power, without attenuation, is limited to -30dBm. The source power level accuracy is very compromised. Use a source power calibration to make the source power somewhat
more accurate.

- **Open Loop R<n>** - No ALC; Receiver Leveling ONLY - where <n> is the reference port. To change the reference port, click **Power and Attenuators**, then **Receiver Leveling**. Learn more.

**Source Attenuator:** Enter source attenuation for the specified port number. Not available for external sources.

**Auto range source attenuator:** Check to allow the analyzer to select the optimum attenuation value to achieve the specified test port power. Learn more.

**Power and Attenuators** - Click to start the **Power and Attenuators** dialog.

**Phase**

**Phase State** - Choose from the following:

- **Off** - Phase is NOT set or controlled.
- **Controlled** - Phase is measured and iterated to within the specified tolerance. Specify the receivers and iteration properties to use to control phase.
- **Open Loop** - Phase and power is set, but receivers are NOT used to measure and iterate the phase or power of the source. Therefore, the setting of phase is not as accurate or stable. Open Loop mode can be used with phase sweep (for example, from 0 to 360 degrees). However, each sweep may not start at 0 degrees. NO settings on the Phase Control Setup dialog are used in Open Loop. **Note:** After selecting Open Loop, set each source to **ON** (not Auto) using the **Power and Attenuators** dialog.

**Sweep Phase** Check to sweep on the specified source port.

**Start / Stop Phase** - Set both the Start and Stop power levels for power sweep. Set a single (Start) power level for the specified source port

**Referenced To** - Select the port to which the controlled phase port is to be referenced. The two internal VNA sources are available ONLY at specific ports. These choices are limited for you. For example on a 4-port PNA-X, the possible port pairings are: 1/3, 1/4, 2/3, or 2/4. Port 1 can NOT be paired with Port 2, and Port 3 can NOT be paired with Port 4. Learn more about these limitations.

**Control Param.** - Select the receivers to be used to measure the phase of the sources.

- The LEFT receiver (a2 in the above image) measures the controlled source.
- The RIGHT receiver (a3 in the above image) measures the reference source.
The swept phase or phase offset will be the difference between these two receivers. Learn more.

Select the receivers based on your application. You are responsible to make sure that your DUT configuration routes the signals of interest to the correct receivers. Otherwise, the phase will not be properly controlled.

**Tolerance**  When consecutive phase measurements of the same data point are within this value of each other, then the phase measurement is considered settled.

**Max Iterations**  Sets the maximum number of background phase measurements to perform in order to achieve settling. If the phase is not sufficiently settled after these measurements, then the closest value is used.

---

**Note:** When both Power and Phase are swept, both are swept SIMULTANEOUSLY in a single sweep.

**Match Correction**

**Note:** Without match correction, receiver-power measurements use simple response calibrations, which do not compensate for the mismatch between the DUT and the match of the test system. This mismatch can appear as ripple in swept-frequency power measurements. With match correction enabled on a given port, an extra sweep is taken at that port to measure the match of the DUT using a 1-port calibration. These data are then used with the port-match data obtained from the Cal All calibration to correct for the mismatch between the DUT and the test system, providing more accurate power measurements.

**Match Correction On**

- Check to perform Match Correction on the specified source port over the specified frequency range. An extra sweep is performed to measure the match term.
- Clear this box if the input of your DUT is well-matched to the analyzer source port.

**Test Receiver** - Select a test receiver to be used for Match Correction. Only logical receiver notation (b1, b2 and so forth) is available. Learn more.

**Reference Receiver** - Select a reference receiver to be used for Match Correction. Only logical receiver notation (a1, a2 and so forth) is available. Learn more.

**Select Frequency Range** - Starts the following dialog.
Select the frequency ranges over which Match Correction is to be performed.

When match-correction is enabled, extra trace parameters are generated in the channel for the match measurements (e.g. S22_F1). The number of match parameters generated depends on which ports enable match correction, and the frequency ranges selected for each port. If these parameters are deleted, or if a set of trace parameters are loaded from an .xml file that doesn’t include the match parameters (loading an xml file overwrites the existing trace definitions), then the match measurements will not be performed, even if the setup dialog includes them. To regenerate the match parameters and enable match-corrected power measurements, match correction must be turned off for a port and then reenabled.

Define a Measurement Parameter

Four 'default' parameters are offered.

In addition, custom parameters can be defined using the **Edit Parameters Dialog**

### How to select and configure Measurement Parameters

#### Using Hardkey/SoftTab/Softkey

1. Press Meas > Main > Other....

#### Using a mouse

1. Click Response
2. Select Meas
3. Select More Measurements...
When match correction is enabled, the firmware automatically generates DUT-reflection parameters Sxx_Fn (e.g. S11_F1, S11_F2 ...) measured with a 1-port calibration. These are the parameters that should be selected for accurate reflection measurements of the DUT. A user-defined Sxx (e.g. UserS11 below) will only be response corrected, and will have considerably more error than the equivalent firmware-generated reflection parameter, which has a full 1-port calibration.

Select one of the following 'default' parameters, or click **Edit Parameters** to show the following **Edit Parameters** dialog.

**Create in New Window** - When checked, the selected measurements are created in a new window.
Parameters

Select a predefined parameter from the list.

**Parameter Name** - Change the 'default' name. Click **Edit Name** to start the virtual keyboard.

*Note:* Do not use underscores in the parameter name. For example, b2_f1 cannot be used as a parameter name. However, b2f1 is a valid parameter name.

To edit / create a parameter:

**Receiver** - Select a receiver for which to plot measurement results.

**Frequency** - Select the frequency range over which the results will be plotted.

***(Operator)*** - Select an operator when combining multiple receivers. For example, when calculating differential or common mode parameters.

**Buttons**

**New** - Click to create a parameter.

**Remove** - Click to delete the selected parameter.

**Save** - Click to store the parameter to a *.diq file for later recall.

**Load** - Click to recall a parameter that you previously defined and saved.

The following are the contents from an example *.diq file that contains the above parameters.

The file can be created using the above dialog, or edited directly using the format shown below.
<Source_Phase_Control>
  <ParameterList>
    <Param1 ParamName="I pwr" ParamValue="a1_F1"/>
    <Param2 ParamName="Q pwr" ParamValue="a3_F1"/>
    <Param3 ParamName="LO+I/Q" ParamValue="b2_F3"/>
    <Param4 ParamName="LO-I/Q" ParamValue="b2_F4"/>
    <Param5 ParamName="LO leakage" ParamValue="b2_F2"/>
    <Param6 ParamName="I/Q input imbal" ParamValue="a1_F1/a3_F1"/>
    <Param7 ParamName="Image rejection" ParamValue="b2_F3/b2_F4"/>
    <Param8 ParamName="LO+I/Q comp" ParamValue="(b2_F3)/(a1_F1)"/>
    <Param9 ParamName="M_Port 2_F2" ParamValue="b2_F2/a2_F2"/>
    <Param10 ParamName="M_Port 2_F3" ParamValue="b2_F3/a2_F3"/>
    <Param11 ParamName="M_Port 2_F4" ParamValue="b2_F4/a2_F4"/>
  </ParameterList>
  <FrequencyRanges>
    <Range1 Start="1200000000.000000" Stop="1200000000.000000" IFBW="10000.000000" IsCoupling="false" CouplerId="1" Offset="0" IsUpConvert="true" Multiplier="1" Divisor="1"/>
    <Range2 Start="2500000000.000000" Stop="2500000000.000000" IFBW="10000.000000" IsCoupling="false" CouplerId="1" Offset="0" IsUpConvert="true" Multiplier="1" Divisor="1"/>
    <Range3 Start="3700000000.000000" Stop="3700000000.000000" IFBW="10000.000000" IsCoupling="true" CouplerId="2" Offset="1"/>
  </FrequencyRanges>
</Source_Phase_Control>
IsUpConvert="true" Multiplier="1" Divisor="1"/>

<Range4 Start="1300000000.000000" Stop="1300000000.000000" IFBW="10000.000000"
  IsCoupling="true" CoupleId="2" Offset="1"
  IsUpConvert="false" Multiplier="1" Divisor="1"/>

</FrequencyRanges>
</Source_Phase_Control>

How to Select X-Axis

Using Hardkey/SoftTab/Softkey

1. Press **Sweep > Main > X-Axis Type...**

Using a mouse

1. Click **Stimulus**
2. Select **Sweep**
3. Select **X-axis Type...**

Select X-Axis Dialog help

Select the domain to display on the X-axis of the window.

Markers will be annotated and data files are saved using these choices.

Choose from the following:
Calibrating a Differential IQ Channel

Calibration of the Differential IQ Channel is performed using the **Cal All Calibration Wizard**.

Cal All should be performed with all of the front panel jumpers in place and all of the path switches in the normal configuration. After the cal has been performed, the path switches can be set to the rear panel position if external sources are routed from the rear panel to the test ports.

This calibration appears as an S-parameter calibration in the Cal All Wizard.

**Direct Access Receiver Calibration**

When all four test ports are being used, but at least one port uses ONLY the source and its reference receiver, you can use the test port receiver for that port accessed through the front-panel jumper ports.

This measurement is calibrated by first performing the Cal All Calibration.

Then use the Cal Plane Manager - Direct Receiver Calibration to remove the effects of the transmission line that is used to connect the DUT to the receiver through the front-panel jumper port. See the detailed procedure.

**Note:** Only corrected scalar magnitude measurements are possible at the direct-access receiver. The phase information from the Cal All calibration is preserved in the Cal Set and is NOT modified by the Direct Access Receiver Cal.
FCA includes both Scalar (SMC) and Vector (VMC) measurements and calibrations.

In this topic:

- FCA Options Explained
- Comparison of VMC and SMC
- SMC Overview
- Requirements and Limitations
- How to make SMC or VMC Measurements
  - Create a Measurement
  - Make Measurement Settings
    - Sweep Tab
      - Segment Sweep
    - Power Tab
    - Mixer Setup Tabs (separate topic)
    - Select X-axis Display
    - Save Trace Data
    - Avoid Spurs

See Also

- **SMC** Measurements and Calibrations
  - SMC + Phase
- **VMC** Measurements and Calibrations
  - Configure an External LO Source
• SMC with a Booster Amp
• Measure a DUT with an Embedded LO
• For a detailed understanding of FCA, see our Mixer Measurements App Notes.

Examples

• How to make a VMC Measurement
• How to make an SMC Measurement

Other Application topics

FCA Options Explained

• Option S93083A/B provides FCA which includes Scalar Mixer (SMC) and Vector Mixer (VMC) Measurements.
• Option S93082A/B provides ONLY SMC measurements. This is the ONLY FCA option that is allowed on the N5230C.
• Option S93084A/B provides Embedded LO measurements. This option requires one of the Converter Applications.
• See all PNA-X Options and Configurations.

Comparison of SMC and VMC

<table>
<thead>
<tr>
<th>Overview</th>
<th>Scalar Mixer Calibration</th>
<th>Vector Mixer Calibration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>See Hardware setup</td>
<td>See Hardware setup</td>
</tr>
<tr>
<td>Provides highest Scalar accuracy for measurements of conversion loss/gain.</td>
<td>Optionally measures phase</td>
<td>Provides unparalleled accuracy for measurements of relative phase and absolute group delay.</td>
</tr>
<tr>
<td>Combines SOLT and power-meter calibration.</td>
<td>Simpler setup than Vector Mixer Calibration.</td>
<td>Uses combination of SOLT standards and a reciprocal mixer/filter pair during calibration.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>More complicated setup and calibration procedure than Scalar Mixer Calibration.</td>
</tr>
</tbody>
</table>
After calibration, both reciprocal and non-reciprocal mixers and converters can easily be measured.

| Measurements Offered | | Amplitude response $V_{C21}$  
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Both forward and reverse directions.</td>
<td>Phase response</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DUT can be connected to any VNA ports.</td>
<td>Group delay</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DUT input must be connected to VNA port 1.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DUT output can be connected to any other VNA port.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Equipment Required | Power meter and sensor | Reference mixer  
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Power meter and sensor</td>
<td>Calibration mixer/filter combination (must be reciprocal $S_{21} = S_{12}$).</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| Common equipment for both SMC and VMC  
| | Mechanical cal kit or ECal module |  


Requirements and Limitations

The following VNA features are NOT available with FCA:

- Analog Sweep (Stepped sweep mode only)
- Log frequency sweeps
- ECal User Characterization (can NOT be created in FCA channel)
- Time Domain
- Balanced measurements
- Port extensions
- Some Fixturing Features
- External Test Set Control (Option S93551A/B)
- PMAR (Power Meter As Receiver)
How to make SMC or VMC Measurements

The following is an overview of how to make an FCA measurement:

1. DECIDE to use either a SMC or VMC measurement. See a comparison of these two measurement types.
2. CREATE an SMC or VMC Measurement.
3. SETUP the measurements.
4. CALIBRATE your SMC or VMC measurement.

Create an SMC or VMC Measurement

1. Press Setup > Main > Meas Class...
2. Select SMC or VMC, then either:
   - OK delete the existing measurement, or
   - New Channel to create the measurement in a new channel.
3. The default SMC or VMC measurement is displayed.
4. See SMC measurements or VMC measurements to learn about the parameters that are offered in each.

How to make SMC or VMC settings

Using Hardkey /SoftTab /Softkey

1. Press Freq > Main > SMC Setup... or VMC Setup... .

Using a mouse

1. Click Stimulus
2. Select Frequency
3. Select SMC or VMC Frequency...

Valid Mixer Configuration / Sweep Type Combinations
Configuring the SMC and VMC Setup dialog can be challenging at first. RED messages like this one appear at the bottom of the Setup dialog to notify you of an invalid setup.

At least one range (Input, LO, or Output) MUST be Fixed.

The following are the **Valid Mixer Configurations**:

<table>
<thead>
<tr>
<th>Sweep Type</th>
<th>Input</th>
<th>LO</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>Swept</td>
<td>Swept</td>
<td>Fixed</td>
</tr>
<tr>
<td></td>
<td>Swept</td>
<td>Fixed</td>
<td>Swept</td>
</tr>
<tr>
<td></td>
<td>Fixed</td>
<td>Swept</td>
<td>Swept</td>
</tr>
<tr>
<td>CW Time</td>
<td>Fixed</td>
<td>Fixed</td>
<td>Fixed</td>
</tr>
<tr>
<td>Power</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Tips**

Although you will soon become comfortable navigating these tabs, at first it may be best to complete the dialog in the following order:

1. For 2-stage mixers, select Mixer Setup settings.
2. Select Sweep tab settings.
3. Select Mixer Frequency settings.
4. Select Power settings.
5. Select Mixer (LO) Power settings.

The following FCA settings are common to VMC and SMC:
Sweep Tab - SMC and VMC dialog box help

**Sweep Type**

- **Linear**  Sweep frequency. Measurements are displayed on a standard grid with ten equal horizontal divisions. Learn how to select the range to display on the X-Axis.

- **CW Time**  All ranges are set to a Fixed (CW) frequency, and the data is displayed versus time.

- **Segment Sweep**  Sweep user-defined segments. Learn more.

- **Power**  Sweep Input or LO power.

**X-axis Point Spacing**  (Available only with Segment Sweep) - Learn about this feature

**Avoid Spurs**  - Learn about this feature.

**Reversed Port 2 Coupler**  (SMC ONLY)  Check when making SMC measurements with the Port 2 coupler reversed. Learn how. Do this to increase power at low frequencies. Checking this box has the following effects:

- The measurement is faster because Reduce IFBW is no longer performed on SC21 measurements.

- The SMC Calibration no longer attempts to level the Port 2 power (it is not possible to level the power through the coupler at those low frequencies). This keeps the VNA from producing "source unleveled" errors on Port 2.

**Number of Points**  Learn about this feature.

**IF Bandwidth**  Learn about this feature.

**Phase Reference Point**  (SMC ONLY)  Learn about this feature.

Learn about these buttons.
### Power Tab - SMC and VMC dialog box help

**Note:** Set LO Power on the Mixer (LO) Power tab.

Configures Input and Output power settings for an FCA measurement. Use the Mixer Power tab to set LO power.

**Power ON (All channels)**  Check to turn RF Power ON or clear to turn power OFF for all channels.

**Port Powers Coupled**  Check to set the same power level at the DUT Input and Output ports. The LO power is NOT coupled. Clear to set power levels independently for each test port. Uncouple power, for example, to apply more power in the reverse direction than in the forward direction. Learn more about Setting Independent Port Power.

**DUT Input / Output Port**

Select the VNA port that is connected to the DUT Input and Output. For VMC, the DUT input must always be connected to VNA port 1 because of the need for a reference mixer on port 1.

**Power Level**  Set the power level to the DUT Input port. To set power at the Output port, clear the **Port Powers Coupled** checkbox.

**Source Attenuator Auto**  Check to automatically select the correct attenuation to achieve the specified input power. Clear, then select attenuator setting that is used achieve the specified Power Level. Learn more about Source Attenuation.

All VNA channels in continuous sweep must have the same attenuation value. Learn more.
**Receiver Attenuator**  Specifies the receiver attenuator setting for the DUT port.

**Source Leveling**  Choose from: **Internal** (normal operation), **Open Loop** (used only for Wideband Pulse measurements), or **Receiver - R1** for Receiver Leveling.

**DUT Input and Output Port Power Sweep**

Available when Power (sweep) is selected on the Sweep tab.

**Input Start** and **Stop Power**  To set Start and Stop power at the Output port, clear the **Port Powers Coupled** checkbox.

**Note:** If your DUT requires more input power than this setting allows below 3.2 GHz, use the PNA-X **Hi-power mode**, available from the RF Path Configuration dialog. The disadvantage to this is higher harmonic content.

**Power Points**  Number of power points to measure.

**Power Step (Size)**  Calculated value from current Start, Stop, and Points settings. This setting can NOT be changed directly.

**Path Configuration**  click to launch the RF Path Configuration dialog.

The following tabs are shared with all Mixer / Converter Applications:

- Mixer Frequency tab
- Mixer (LO) Power tab
- Mixer Setup tab
Mixer Frequency tab - SMC and VMC Setup - dialog box help

Learn about this dialog

Mixer Setup tab - SMC and VMC Setup - dialog box help

Learn about this dialog
Learn about this dialog

**FCA Segment Sweep**

The following settings appear on the Mixer Frequency tab when **Segment Sweep** is selected on the Sweep tab.

**Mixer Frequency tab - Segment Sweep - SMC and VMC dialog box help**

**How to configure a segment using the GUI:**

1. Click **Add**. Click Delete to remove a segment and renumber all subsequent segments.

2. State is **ON** by default. Click **OFF** and that segment will not be included in the sweep.
3. Configure **Frequency settings** for Input, LO, and Output ranges.

- For each segment, the same sweep requirements apply as a standard (non-segment) sweep. For example, at least one range MUST be Fixed (Start = Stop frequencies).
- The Input, Output, and LO frequencies of segments ARE allowed to overlap other segments.
- All segments must sweep in either the forward (Start<Stop) or reverse (Start>Stop) directions. Mixed sweep directions are NOT allowed.
- The following settings can be set independently for each segment:
  - **Number of Points** - Total number of points for all segments is limited to the Max allowed by the VNA.
  - IF Bandwidth
  - **Port Powers**: (Input, Output, LO 1, LO2). These settings override the settings on the Power tab.

- The following settings apply to ALL segments:
  - Number of Converter/Mixer Stages (1 or 2).
  - LO Source Selections
  - All Input and LO Multipliers and Dividers
  - Source and Receiver Attenuator Settings
  - Source Leveling
  - Avoid Spurs
  - Nominal Incident Power (SMC only)
  - X-Axis Display (Input, LO1, LO2, Output) There must be at least two data points for this setting to be available.
  - X-Axis Point Spacing (vs Normal point spacing).
  - SMC + Phase

- Mixer Segment sweep data can be saved to a *.S2PX file (NOT *.S2P).
- Mixer Segment setup information is saved to a *.MXRX or *.CSV file. Learn more.

**How to add multiple segments using the GUI and Excel *.csv file:**
Start with the GUI to set up mixer settings and to create the first segment then save it as a *.csv file. The *.csv file will then be opened in Excel and the additional segments will be entered. Learn more.

1. Click **Add**.

2. State is **ON** by default. Click **OFF** and that segment will not be included in the sweep.

3. Configure **Frequency settings** for Input, LO, and Output ranges.

4. Click on the **Save...** button.

5. Select **CSV (Comma delimited) (*.csv)** to save the settings and first segment to a *.csv file.

6. Open the *.csv file in Excel.

7. Highlight the segment row then copy it.

8. Paste the copied segment row to add the desired number of segments into the Excel file.

9. Modify the settings for each added segment as necessary. The following is an example showing 3 segments:

10. Save the *.csv file.

11. In the GUI, load this *.csv file.
Learn about these buttons

Apply and Interpolate FCA Cal Sets

In general, when a Cal Set covers a wider frequency range than the channel, the VNA will offer to interpolate the Cal Set when it is applied. Learn more. However, with FCA measurements the LO frequency range may also be considered.

- VMC measurements ALWAYS CONSIDER the LO frequency range and performs interpolation if possible. If the LO frequency range of the measurement is NOT within the LO frequency range of the Cal Set, than the Cal Set can NOT be applied.

- SMC measurements ALWAYS IGNORE the LO frequency range. Therefore, if the Input and Output frequency ranges of the measurements are within those of the Cal Set, then the Cal Set is interpolated if necessary and applied. For example, this would allow you to perform ONE SMC calibration with Input range = the VNA frequency span, LO at 0 Hz, and Output range + the VNA frequency span. This Cal Set could be applied to ALL SMC measurements. Learn more about applying SMC Cal Sets.

These same general concepts apply to segment sweeps. However, if ALL applicable frequency ranges (SMC: Input and Output and VMC: Input, Output, and LO) are NOT within those ranges of the measurement for ONE segment, then the Cal Set is NOT applied for ANY segment.
Select X-axis Display for FCA Measurements

Click **Sweep > Main > X-Axis Type**, then select the desired type.

When **Sweep Type = Linear**, you can choose to show the frequency range of any of the swept parameters on the X-axis.

For example, the following image shows an SMC Fixed Output response with the **Input frequency range** on the X-axis:

- **Output**: 100 MHz (data trace)
- **Input**: 2 GHz to 23 GHz (X-axis)
- **LO**: 1.9 GHz to 22.9 GHz (not shown)

Marker annotation shows Output power at Input frequency.

Save Trace Data

You can save your FCA measurement data in several standard formats.

Click **Save Recall > Save Other > Save Data...**.

The following shows how CSV and SNP files are saved.

Mixer Trace Data

When you select **Mixer Trace Data**, the FCA data is saved to a CSV file in the following format:
# MIXER TRACE FILE, A.01.00

SegIndex, InputFreq, OutputFreq, LO1Freq, InputPower, LO1Power, SC21 Mag (dB), SC21 Phase (Deg)

**SNP Format**

Each record contains 1 stimulus value and 4 parameters (total of 9 values) as follows:

Stim Real($p_1$) Imag($p_1$) Real($p_2$) Imag($p_2$) Real($p_3$) Imag($p_3$) Real($p_4$) Imag($p_4$)

where $p_X$ is the parameter depending on measurement type:

<table>
<thead>
<tr>
<th>Measurement Type</th>
<th>$p_1$</th>
<th>$p_2$</th>
<th>$p_3$</th>
<th>$p_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalar</td>
<td>S11</td>
<td>SC21 (FWD)</td>
<td>SC12 (REV)</td>
<td>S22</td>
</tr>
<tr>
<td>Vector</td>
<td>S11</td>
<td>VC21</td>
<td>VC12</td>
<td>S22</td>
</tr>
<tr>
<td>Mixer Characterization</td>
<td>Directivity</td>
<td>Source Match</td>
<td>Reflection Tracking</td>
<td>M21</td>
</tr>
</tbody>
</table>

- If correction is OFF, data is only saved for the active parameter. Zeros are saved for all other parameters.
- If correction is ON, data is saved for all of the parameters.

All files contain the following Header Information: Brackets [ ] contain parameters.

!Keysight [Instrument Model Number]: [version]
!Mixer S2P File: [Mixer Measurement Type]
!Parameters: [Parameter List]
!Calibration State: [On/Off]

!# Begin Mixer Setup
! [Mixer Setup parameters listed here]
! [Mixer Parameter 1]
·
·
! [Mixer Parameter n]
!# End Mixer Setup

# [S2P data here]

**Avoid Spurs**

On the Mixer Setup dialog box, check **Avoid Spurs**
The Avoid Spurs feature of the Frequency Converter Application attempts to prevent unwanted mixing products from appearing on the VNA screen. The Avoid Spurs feature does not significantly impact measurement speed.

**Note:** The Avoid Spurs feature is OFF by default for FCA calibrations. For highest accuracy, make measurements with the Avoid Spurs feature at the same state (ON or OFF) as was used when calibrating.

**Description**

A spur, or spurious signal, is a term used to describe the unwanted product of two signals mixing together. When you configure the mixer setup dialog box for a desired Output, the VNA computes the frequencies of potential unwanted signals. By manipulating internal VNA hardware, these signals are avoided and do not appear on the VNA display. This means you do not need to use external filters to prevent spurious signals from appearing on the VNA display.

The time required for the VNA to compute the frequencies of unwanted spurious signals MAY be noticeable depending on the number of data points in your measurement. However, once computed, the time required for the VNA to avoid the spurs is usually insignificant.

**Limitations**

The Avoid Spurs utility cannot avoid every spur. However, when there is a choice of spurs to avoid, it will avoid the largest spur.

**The Computation of Avoided Spurs**

The Avoid Spur computer avoids the following spurs:

- LO, and its interaction with internal VNA components, and 16 of its harmonics.
- Input frequencies and 16 of its harmonics.
- Undesired Image frequencies (Sum or Difference) and 16 of its harmonics.
Scalar Mixer/Converter Measurements (SMC)

SMC Setup and Calibration is very similar to VMC. See FCA Overview to learn about the features that are common to these two applications.

The following information is unique to SMC:

- SMC Hardware Setup
- Create an SMC Measurement
- SMC Parameters Offered
- The SMC Mixer Setup dialog
- Speed Up SMC Measurements
  - Use Nominal Incident Power
  - Apply a Cal Set or SMC Cal Type
    - Reverse Port 2 Coupler below 55 MHz (separate topic)
- SMC Calibration
- SMC + Phase (separate topic)

See Also

Embedded LO
How to make an SMC Fixed Output Measurement
SMC with a Booster Amp
Programming Commands

SMC Hardware Setup
SMC requires a **power meter/sensor, two sources, and a Cal Kit or ECal module**

- Your DUT can be connected to any VNA ports. Learn more.
- When using a PNA-X with an Internal Second Source, the external source is NOT necessary. Learn which VNA ports can be used for the LO.
- For 2-port VNAs, connect **External Source** to the VNA GPIB Controller port. Learn how to Configure an External Source.
- Connect the **10 MHz reference signal** of an external source to the VNA. This is especially important with SMC + Phase measurements.
Use either a GPIB power meter or USB power sensor.

**How to configure two power sensors** to cover the SMC measurement frequency range.

Using a dual channel power meter, with both sensors connected:

1. At the SMC Select DUT Connectors dialog, click **Source Cal Settings**
2. At **Source Calibration Settings** dialog, click **Power Meter Config**
3. At **Power Meter Settings** dialog, click **Sensors**
4. At **Power Sensor Settings** dialog, clear the "Use this sensor only..." checkbox for both sensors.
5. Then enter the Min and Max Frequencies for both sensors.

During the SMC Cal, you will be prompted to connect each sensor at the appropriate time.

**Create an SMC Measurement**

1. Press **Setup > Main > Meas Class....**
2. Select **SMC**, then either:
   - **OK** delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.
3. An SC21 measurement is displayed.

**SMC Parameters Offered**

To select additional parameters to display, click **Trace**, then click on a new trace, then select a parameter from the list.
**Important Note:** Connecting your DUT to the VNA:

**RF** and **IF** terminology is NOT used in FCA because the VNA does not know how the DUT is labeled or how it will be used. Instead, the general terms **INPUT** and **OUTPUT** are used.

- **INPUT** - The DUT port being stimulated with frequencies before conversion.
- **OUTPUT** - The DUT port outputting converted frequencies.

**INPUT** and **OUTPUT** Frequencies are specified using the **Mixer Setup dialog box**.

The DUT input and output can be connected to any VNA ports.

**Note:** Although there are MANY configuration possibilities, the following images and descriptions show ONLY a DUT connected to VNA ports 1 and 2.

Legend:

- **Black** are ratioed measurements (test port/reference receiver).
- **Green** are unratioed measurements (either a test port OR reference receiver).

<table>
<thead>
<tr>
<th>DUT <strong>Input</strong> to VNA <strong>port 1</strong></th>
<th>DUT <strong>Input</strong> to VNA <strong>port 2</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>DUT <strong>Output</strong> to VNA <strong>port 2</strong></td>
<td>DUT <strong>Output</strong> to VNA <strong>port 1</strong></td>
</tr>
</tbody>
</table>

**Ratioed**

- **SC21 (Conversion Loss)** Stimulus at Input, response at Output (B/R1).
- **SC12 (Reverse Isolation)** Stimulus at Output, response at Input (A/R2)

**Ratioed**

- **SC12 (Conversion Loss)** Stimulus at Input, response at Output (A/R2)
- **SC21 (Reverse Isolation)** Stimulus at Output, response at Input (B/R1)
### S11 (Input match)
Stimulus and response at Input (A/R1)

### S11 (Output match)
Stimulus and response at Output (A/R1)

### S22 (Output match)
Stimulus and response at Output (B/R2)

### S22 (Input match)
Stimulus and response at Input (B/R2)

#### Unratioed
Absolute test port receiver measurements. The receiver is automatically selected depending on the DUT configuration.

- **IPwr** (Incident Power) - stimulus and response at Input.
- **RevIPwr** (Reverse Incident Power) - stimulus and response at Output.
- **OPwr** (Output Power) - stimulus at Input, response at Output.
- **RevOPwr** (Reverse Output Power) - stimulus at Output, response at Input.

#### Internal ADC Stimulus / Response Measurements (AI).

AI(x,y) where:

- **x**=ADC to use. Choose 1 or 2.
- **y**=port number at which ADC is available

Learn more about [ADC Measurements](#).

#### Externally-configured DC Measurements
Learn more.

---

### SMC Mixer Setup

#### How to start the SMC Mixer Setup dialog

**Using Hardkey/SoftTab/Softkey**

1. Press **Sweep > Main > SMC Setup... > Mixer Setup** tab.

**Using a mouse**

1. Click **Stimulus**
2. Select **Sweep**
3. Select **SMC Setup...**
4. Select **Mixer Setup**

---

The following SMC Mixer Setup dialog tabs are presented:
- Sweep Tab (shared with VMC)
- Power Tab (shared with VMC)
- Mixer Freq Tab (shared with all converter apps)
- Mixer Power Tab (shared with all converter apps)
- Mixer Setup Tab (shared with all converter apps)

**Speed Up SMC Measurements**

Using default SMC settings, any calibrated SMC measurement requires four sweeps. However, you can reduce the number of sweeps required by selecting one or more of the following settings.

- Use Nominal Incident Power
- Apply Cal Set or Cal Type
- To speed up a Swept LO measurement when using an external source for the LO, use Hardware List (BNC) Trigger setting. Learn more.
- Reverse Port 2 Coupler below 55 MHz (separate topic)

**Use Nominal Incident Power**

Click **Response**, then **Measure**, then **Use Nominal Incident Power**

Each data sweep of a fully corrected SMC transmission measurement actually requires FOUR data sweeps. When you clear **Use Nominal Incident Power**, the reference receiver (R1 or R2) does NOT measure incident power. Instead, the incident power is assumed to be at the level that was set with the Source Power Calibration that is done as part of every SMC measurement. The degradation in accuracy is very negligible if the input or output of your DUT is well-matched.

This selection eliminates sweeps ONLY when both Include Input Match AND Include Output Match is cleared on the Cal Type dialog. Learn more.

**Apply a Cal Set or SMC Cal Type**

You can create an FCA measurement and apply an existing Cal Set as you can with any VNA measurement. Learn about Cal Sets. In addition, from a Cal Set, you can apply a specific SMC Cal Type to an existing SMC measurement.
How to apply an SMC Cal Type

1. Create an SMC measurement
2. Calibrate or apply an existing SMC Cal Set, then...

Using Hardkey/SoftTab/Softkey

1. Press Cal > Main > Correction Methods....

Correction Method dialog box help

By default, each SMC calibration requires FOUR sweeps. Clearing boxes will eliminate sweeps and speed up your SMC measurements. The difference in speed is most noticeable when making fixed input or fixed output measurements with an external LO source.

Include input match correction Check to perform a sweep to measure and correct for INPUT match. Clear this box if the input of your mixer is well-matched to the VNA, or if your setup does not permit a valid S11 measurement.

Include output match correction Check to perform a sweep to measure and correct for OUTPUT
match. Clear this box if the output of your mixer is well-matched to the VNA, or if your setup does not permit a valid S22 measurement.

**Include SC12 Sweep** Check to perform a reverse sweep to measure SC12.

- When checked (default setting), a calibrated SMC measurement sweeps in both forward (SC21) and reverse (SC12) directions.
- Clear this checkbox to eliminate sweeps in the reverse direction. This means that the following measurements will NOT be corrected: SC12, RevOPwr, RevIPwr.

**Corrected Parameters** Lists the parameters that can be corrected given the boxes that are currently checked. These parameters may not be currently measured.

**Calibration Type** Shows the type of SMC Cal that will be applied given the boxes that are currently checked.

Learn about **Use Nominal Incident Power**

**How many sweeps can be eliminated?**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Parameters</th>
<th>Learn about parameter abbreviations</th>
<th># of sweeps</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL checked and clear Use Nominal Incident Power</td>
<td>IPwr, OPwr, RevIPwr, RevOPwr, SC21, SC12, S11, S224 Total</td>
<td></td>
<td>Total</td>
</tr>
<tr>
<td>Perform this action...</td>
<td>to REMOVE these parameters...</td>
<td>and these sweeps</td>
<td></td>
</tr>
<tr>
<td>Clear &quot;Include SC12&quot;</td>
<td>Remove RevIPwr, RevOPwr, SC12</td>
<td>Removes 1</td>
<td></td>
</tr>
<tr>
<td>Clear &quot;Include OUTPUT match&quot;</td>
<td>Remove S22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clear &quot;Include INPUT match&quot;</td>
<td>Remove S11</td>
<td>Removes 1 when Nominal is checked*</td>
<td></td>
</tr>
<tr>
<td>Check &quot;Use Nominal Incident Power&quot;</td>
<td>Remove IPwr, RevIPwr</td>
<td>May remove up to 2*</td>
<td></td>
</tr>
<tr>
<td>ALL cleared and check Nominal Incident Power</td>
<td>OPwr, SC21</td>
<td></td>
<td>1 Total</td>
</tr>
</tbody>
</table>

*S11 shares a sweep with IPwr and S22 shares a sweep with RevIPwr. Therefore, when **Include Input Match** or **Include Output Match** is checked, then checking Nominal incident power does nothing.

**VMC measurement** sweeps can NOT be eliminated.
SMC Calibration Overview

The SMC Calibration Wizard guides you through this process.

When applying a Phase Reference cal set, step 1 (power cal) is NOT performed.

1. Connect a power meter / sensor to VNA Port 1. At each step of the input and output frequency, the VNA measures:
   - input match of the power sensor
   - source power of the VNA

2. Perform two Full 2-port calibrations: one over the INPUT frequencies and one over the OUTPUT frequencies of the DUT. (If your DUT is a linear device, the calibration uses only the INPUT frequency range.) Use either a mechanical calibration kit or an ECal module.

For Mixers / Converters with High-output Power

The Unknown Thru method is NOT valid when there is over 40 dB of combined loss in the Unknown Thru and calibration path. In this case, the following calibration and correction method is recommended.

- On the Cal Wizard Modify Frequency page, select Defined Thru or Flush Thru as the Thru method. When using an ECal module, also on the Modify Frequency page, disable (clear) Do Orientation due to very low power.
- After calibration, on the Correction Method dialog, CLEAR the Include output match correction and Include SC12 Sweep check boxes. Check ONLY Include input match correction.
- To learn more about High-power measurements, see our App Notes.

SMC Cal Wizard

The following dialog boxes are presented during an SMC Calibration.

Indented steps are optional.

- Calibration Setup
  - Waveguide/In-fixture/On-Wafer Setup
- Select DUT Connectors and Cal Kits
- Modify Frequency Cal
- Specify how the ECal module is connected
- Power Cal Settings
- SMC Cal Steps
- Calibration Completed
- Specify Adapter Delay

**How to Perform a SMC Calibration**

1. Create an SMC measurement, then...

**Using Hardkey/SoftTab/Softkey**

1. Press Cal > Main > Smart Cal....

**Using a mouse**

1. Click Response
2. Select Cal
3. Select Smart Cal...

**SMC Calibration Setup dialog box help**

Allows you to review and change the settings for your SMC calibration.

**Note:** With release A.09.90 and before, checking both 'Independent power cals' AND 'Use Phase Reference Calset' would generate an error after performing the calibration. With releases AFTER A.09.90, the two settings are compatible.

**Waveguide/In-fixture/On-Wafer Setup** Click Next to launch the following Setup dialog box.
Independent power cals for input and output ports (no thru)  Check if a Thru standard is NOT available. During the power cal, you will be prompted to connect the power sensor to the Input, then the Output port.

**Additional Power Cal Steps**

**Enable LO1 / LO2 Power Cal**  Check when LO1 / LO2 is controlled (on the Mixer Setup tab) to perform a Power Cal on the LO source(s).

**Phase Correction**

**Enable Phase Correction**  Check to enable Phase measurements.

Choose one of the following methods to specify the delay through the characterized mixer. With the first two methods, the phase delay through a Calibration Mixer is measured and compared to the known delay, either entered, or stored in an *.S2PX file.

- **Use Known Mixer Delay**  Enter the fixed, known, delay through the calibration mixer.
- **Use Characterized Mixer**  Select, then browse to the *.S2P file that characterizes the calibration mixer. Use an *.S2PX file when making segmented SMC + Phase measurements. Learn more. Use either of the following two methods to characterize the Cal Mixer over the SMC measurement frequency range:
  
  1. Use the Mixer Characterization Wizard. (Click Response, then Cal, then Mixer Characterization Wizard.) The Cal Mixer has the same requirements as the VMC Cal Mixer. Learn more.

  2. In a calibrated VMC channel, measure the group delay of the calibration mixer, then save to an *.S2P or *.S2PX file. However, a characterized mixer is required to calibrate the VMC channel.

- **Use Phase Reference Calset**  Select, then browse to the Phase Reference Calset that covers the frequency range of the current measurement. Learn more about SMC with a Phase Reference Calibration.

**Waveguide/In-fixture/On-Wafer Setup** dialog box help
This dialog box appears ONLY if you checked the Waveguide/In-fixture/On-Wafer Setup box in the previous Cal Setup dialog.

Allows you to embed or de-embed circuit networks on the input and output of your mixer under test.

For Network1 (Input) and Network2 (Output) select Embed, De-embed, or None.

**Browse** Click to navigate to the .S2P file that models the network to embed or de-embed.

**Reverse port positions for input/output** Check to cause the Fixture/Adapter to be configured with Port 2 connected to the VNA and Port 1 to be connected to the DUT. The image in the dialog is updated to reflect that change.

**Enable Extrapolation** Check (default setting) to apply a simple extrapolation when the S2P file has a narrower frequency range than the channel. The values for the first and last data points are extended in either direction to cover the frequency range of the measurement. A warning message is also displayed when extrapolation is necessary.

**To Embed or De-embed**

- When you have a 2 port network that needs to be connected between the Cal reference plane and the DUT during the measurement, but it is NOT present during the calibration, then that network has to be **De-Embedded** from the port in question during the calibration. In other words, De-Embedding in FCA calibration extends the calibration reference plane to include the two port network.

- When you have a 2 port network that is included as part of the calibration reference plane but has to be disconnected during the measurement, then that 2-port network has to be **Embedded** for the port in question during the calibration. In other words, Embedding in FCA calibration retracts the calibration reference plane to exclude the two port network during the measurement.

**Notes**

- Interpolation is performed when more frequencies are included in the file than in the channel, and the data points do not exactly match those of the measurement.
Select DUT Connectors and Cal Kits dialog box help

Allows you to specify the connector type and Cal Kit for each DUT port.

**Port n**  For each listed VNA port, specify the DUT connector type and gender, and the Cal Kit to use.

**Note:** If your DUT connectors are:

- **Waveguide** Change the system impedance to 1 ohm before performing a calibration. See Setting System Impedance.

- **Not listed** (male and female) Select **Type A** as the connector type. Type A requires a calibration kit file containing the electrical properties of the standards used for calibration (see Calibration kits).

- **Unspecified** (like a packaged device) Select **Type B** as the connector type. Type B requires a calibration kit file containing the electrical properties of the standards used for calibration (see Calibration kits).

**Modify Cal** Check, then click **Next**, to start the Modify Frequency Cal dialog.

**Source Cal Settings** Click to start the Source Cal Settings dialog.
Modify Frequency Cal dialog box help

This dialog appears only when **Modify Cal** is checked on the previous dialog.

**Thru Cal Method**  For each Thru connection, choose the Thru method. Learn more about these choices.

**Cal Type/Stds**  Click to start the Modify Calibration Selections dialog box.

**The following selections are available ONLY if using an ECal module.**

**Do orientation**  When this box is checked (default) the VNA senses the ECal model and direction in which the ECal module port is connected to the VNA ports. If power to the ECal module is too low, it will appear as if there is no ECal module connected. If you use low power and are having this problem, clear this check box to provide the orientation manually. Orientation occurs first at the middle of the frequency range that you are calibrating. If a signal is not detected, it tries again at the lowest frequency in the range.

See note about calibrating for high-output mixer/converters using an ECal module.

**View/Detect ECal Characterizations**  Appears only if an ECal module is selected for use. Click to invoke the View ECal Modules and Characterizations dialog box. Displays a list of ECal modules that are connected to the VNA.
**Power Cal Settings dialog box help**

**Note:** A Use Power Table checkbox (not shown) is available when a mmWave SMC measurement is active. [Learn more.](#)

**Power Cal at:** Select the source port for which a Power Calibration will be performed. The source and receiver correction will be transferred to all other sources and receivers involved in the S-parameter measurements.

**Use Multiple Sensors** (NOT available with mmWave SMC measurements.) Check this box when you want to use more than ONE power sensor to cover the measurement frequency range. The dialog is replaced with the Multiple Sensors dialog (see following image). When "Use Multiple Sensors" is cleared (default setting), connect only ONE sensor to the VNA.

**Power Meter Settings** Click to start the standard Power Meter Settings dialog.

**De-embed (power sensor) adapter** When the power sensor connector is NOT the same type and gender as the DUT connector for the specified port, then for optimum accuracy, extra cal steps are required to measure and correct for the adapter that is used to connect the power sensor to the reference plane.

- Clear this box to NOT compensate for the added adapter.

- Check this box to perform extra calibration steps to measure and correct for the adapter.

Then select the **Power Sensor Connector** type and gender of the power sensor. "Ignored" does NOT compensate for the added adapter, just as if the checkbox were cleared.

- When this connector matches the DUT connector for the same port, then the VNA assumes that there is no adapter. Extra cal steps are NOT required and the Cal Kit selection is not available.

- Otherwise, select the **Cal Kit** to be used to calibrate at the adapter.

See **Accuracy Settings** below.
**Specify how the ECal module is connected** dialog box help

![Specify how the ECal module is connected](image)

This dialog box appears when the **Do orientation** checkbox in the previous **Modify Frequency** dialog box is cleared.

Click the ECal Port that is connected to each VNA port.

---

**SMC Calibration Steps** dialog box help

![SMC Calibration Steps](image)

**Power Level** at which to perform the Power Cal.

It is usually best to set power level to 0 dBm at the power sensor because the power sensor is calibrated at that level. Lower power levels will yield a slower and noisier calibration.

If an external component is used between the PNA-X test port and the calibration reference plane, then adjust the power level so that the power at the sensor is about 0 dBm if possible.

The current source attenuation value is shown on the dialog.

**LO Power Cal (Optional)** When enabled, perform a Source Power Cal at the DUT LO connector. An LO must already be selected. Learn how. The power level of the LO source calibration is set on the **(LO) Power Tab**.
Calibration Completed dialog box help

Finish  Save to the channel's calibration register.

Save As User Cal Set  Starts the Save as User Cal Set dialog box AND save to the channel's calibration register.

Cancel  Calibration is NOT applied or saved.

Learn about Calibration Registers.

Learn about User Cal Sets

Specify delay dialog box help

This dialog appears ONLY when Adapter Removal or Unknown Thru calibrations are performed.

The following values were estimated from the measurement. Most of the time, they are adequate. However, for CW sweep or frequency sweep with large step sizes, the accuracy of the values may be improved.

Adapter delay  To improve this value, measure and record the delay of the adapter with a dense step size. Enter that value here. The required precision value is the accuracy that is required to characterize the delay value.

Nominal phase offset  (Waveguide ONLY). To improve this value, measure and record the phase offset of the Waveguide adapter with dense step size. Enter that value here.

When one connector is coax and the other connector is waveguide, the phase offset has an
ambiguity of 180 degrees. For consistency, the estimate provided here is always between 0 and 180 degrees. You can change this estimate to any value between -180 degrees and +180 degrees.

For SMC calibrations, this dialog box appears twice: once for the input frequencies and once for the output frequencies. The values can be slightly different.
Mixer/Converter Setup

The following dialogs are common to all Converter Apps: FCA, Swept IMDx, IMx Spectrum, NFx, and GCx applications.

**Note:** Swept IMDx, IMx Spectrum, NFx are supported by PNA only. GCx is currently not supported on the M9370A/71A/72A/73A/74A/75A.

- Mixer Frequency tab
- Mixer Setup tab
- Mixer (LO) Power tab
- Fractional Multiplier Examples

**Important Note:** Connecting your Mixer/Converter DUT to the VNA.

**RF** and **IF** terminology is NOT used in this topic because the VNA does not know how your DUT is labeled or how it will be used. Instead, the general terms INPUT and OUTPUT are used to describe the following VNA behavior:

- INPUT - the stimulus frequencies BEFORE conversion by your DUT.
- OUTPUT - the response frequencies, AFTER conversion (either UP or DOWN) by your DUT. Specify UP or DOWN conversion using the + or - symbol for each output.
Settings

**Frequency Format**  Select Start/Stop (Swept) or Fixed. For Linear sweep type, at least one of these must be fixed. For CW or Power, ALL must be Fixed.

**Frequencies** Enter the frequency values for each of the Mixer/Converter ports.

- **Mixer-Product Selector** Determines whether the receivers will tune to the Sum (+) or the Difference (-) of the Input and LO frequencies.

**Calc buttons** Calculates frequency settings based on your other mixer settings. For example, enter the Input frequency range and LO1 frequency range, then press **Calc Output**. The VNA will calculate and display the Output frequencies.

**Input > LO or IF1>LO2** Removes ambiguity when using a Calc button to determine the INPUT frequency.

These check boxes are used ONLY when all 3 of the following conditions are TRUE: (If ALL 3 are NOT true, the VNA does not read these check boxes).

1. Difference (Low) sideband is selected for the corresponding Calculate button AND
2. Output frequency is less than the LO frequency AND

3. One of the Calculate buttons are used to calculate the Input frequency.

**Rules for Configuring a Mixer**

A Red message across the bottom of the dialog indicates that one or more of the following settings are invalid:

- Either ALL ranges (Input, LO, Output) must be Fixed, or ONE Range fixed. TWO ranges can NOT be Fixed or THREE ranges can NOT be Swept.

- For determining a valid mixer configuration with 2 LOs, one Fixed LO and one Swept is equivalent to having a single-stage Swept LO. To configure a 2-stage LO, select Converter Stages: 2 on the Mixer Setup tab.

- INPUT or OUTPUT frequencies cannot be outside the range of the VNA.

- Any combination of INPUT and LO which results in an OUTPUT that sweeps through Zero Hz is NOT allowed.

---

**About Mixer Configuration Files (*.mxr, *.mxrx, or *.csv)**

Save  Saves SOME of the mixer settings to a *.mxr, *.mxrx, or *.csv file.

Load  Recalls a previously-configured mixer *.mxr, *.mxrx, or *.csv file.

**Note:** By default, mixer configurations are saved to a *.mxrX file. Previously, they were saved to a *.mxr file.

ONLY *.mxrx and *.csv files allow saving segmented sweep mixer setups. Currently, only FCA allows segmented sweeps. In all other respects, these new file types are completely backward compatible with *.mxr files.

**What Mixer Settings are Saved?**

- Sweep Type, frequency, and power settings.

- With Segment Sweep Type, all segment settings are saved.

**Converter App Compatibility**

The mixer setup files that are used with FCA, NFx, and GCx for PNA ARE compatible. However, *.mxr(x) files created in IMDx contain information that is NOT included with other
*mxr(x) files.

- When *.mxr(x) files created in other apps files are recalled into IMDx, the VNA attempts a 'best guess' at missing information.
- When IMDx *.mxr(x) files are recalled into other apps, the extra information is ignored.
- When *.mxr(x) files are recalled into an IMx Spectrum channel with any range being swept, the recall is ignored and an error message appears.

**External Sources**

A *.mxr(x) file includes an LO source name. However, it does NOT include the LO Source configuration. Therefore, when using a *.mxr(x) file that was created on a different VNA, the VNA will display an error if it does not find the LO Source configuration using EXACTLY the same LO source name.

**CSV Files**

Mixer configuration files can be saved as a *.csv file. This format was implemented to be used for setting up mixer segments. Start with the GUI to set up mixer settings and to create the first segment then save it as a *.csv file. The GUI can be used to set up multiple segments but is not as convenient as editing an existing Excel *.csv file directly.

The *.csv file has two main categories:

- #BaseSetting
- #MixerSegments

**Note:** The CSV file also supports adding comments. All comments must be added before the #BaseSetting row in the CSV file.

#BaseSetting captures the following fields:
<table>
<thead>
<tr>
<th>Default Columns¹</th>
</tr>
</thead>
<tbody>
<tr>
<td>NumberOfStages²</td>
</tr>
<tr>
<td>SegmentSweepMode</td>
</tr>
<tr>
<td>ReduceIFBandOFF</td>
</tr>
<tr>
<td>CenterSpan</td>
</tr>
<tr>
<td>AvoidSpurs</td>
</tr>
<tr>
<td>PowerCalibrationOn</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Conditional Columns (auxiliary setting): Swept IMD Converter</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMDXSweepType</td>
</tr>
<tr>
<td>IMDXVersion</td>
</tr>
<tr>
<td>RF2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Conditional Columns: Dual Converter Stage</th>
</tr>
</thead>
<tbody>
<tr>
<td>LO2Numerator</td>
</tr>
<tr>
<td>LO2Denominator</td>
</tr>
<tr>
<td>LO2 (auxiliary setting)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Conditional Columns: PowerMode is Swept, and Converter is not IMDX⁴</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputStartPower</td>
</tr>
<tr>
<td>InputStopPower</td>
</tr>
<tr>
<td>OutputStartPower</td>
</tr>
<tr>
<td>OutputStopPower</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Conditional Columns: IMDX Converter⁵</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputStartPower</td>
</tr>
<tr>
<td>InputStopPower</td>
</tr>
<tr>
<td>OutputStartPower</td>
</tr>
<tr>
<td>OutputStopPower</td>
</tr>
</tbody>
</table>

---

1. Default columns are always included in the csv.

2. NumberOfStages (previously IsSingleStage) has been moved under ‘#BaseSettings’ and will be used to set converter stage value for each segment.

3. InputPowerMode, OutputPowerMode, LO1PowerMode, and LO2PowerMode (in dual converter stage) have been consolidated into PowerMode in the CSV.

4. Start and Stop powers for Input, Output, LO1, and LO2 (in dual converter stage) are included in the CSV when PowerMode is Swept. When PowerMode is Fixed, these values will be based on the power values from the segments.
For IMDX converter, Start and Stop powers for Input, Output, LO1, and LO2 are included regardless of value in PowerMode column.

#MixerSegments captures the following fields:

<table>
<thead>
<tr>
<th>Default Columns</th>
</tr>
</thead>
<tbody>
<tr>
<td>SegmentNumber</td>
</tr>
<tr>
<td>Dwell</td>
</tr>
<tr>
<td>LO1MixingMode</td>
</tr>
<tr>
<td>State</td>
</tr>
<tr>
<td>InputRangeMode</td>
</tr>
<tr>
<td>InputGreaterThanLO1</td>
</tr>
<tr>
<td>IFBandwidth</td>
</tr>
<tr>
<td>LO1RangeMode</td>
</tr>
<tr>
<td>OutputRangeMode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Conditional Columns: Segment Sweep Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>NumberOfPoints</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Conditional Columns: PowerMode in Base Setting is Fixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputPower</td>
</tr>
<tr>
<td>LO1Power</td>
</tr>
<tr>
<td>OutputPower</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Conditional Columns: Dual Converter Stage</th>
</tr>
</thead>
<tbody>
<tr>
<td>LO2Power</td>
</tr>
<tr>
<td>LO2RangeMode</td>
</tr>
<tr>
<td>IFRangeMode</td>
</tr>
<tr>
<td>LO2MixingMode</td>
</tr>
<tr>
<td>InputGreaterThanLO2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Conditional Columns: Based on InputRangeMode Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Value</strong></td>
</tr>
<tr>
<td><strong>Columns</strong></td>
</tr>
<tr>
<td>Fixed (for all Segments) 1</td>
</tr>
<tr>
<td>Swept (for all Segments) 2</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Fixed/Swept (for all Segments) 3</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Conditional Columns: Based on LO1RangeMode Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Value</strong></td>
</tr>
<tr>
<td><strong>Columns</strong></td>
</tr>
<tr>
<td>Fixed (for all Segments)</td>
</tr>
<tr>
<td>Swept (for all Segments)</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Fixed/Swept (for all Segments)</td>
</tr>
<tr>
<td>-------------------------------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

**Conditional Columns: Based on OutputRangeMode Value**

<table>
<thead>
<tr>
<th>Fixed (for all Segments)</th>
<th>OutputFixedFreq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swept (for all Segments)</td>
<td>OutputStartFreq</td>
</tr>
<tr>
<td></td>
<td>OutputStopFreq</td>
</tr>
<tr>
<td>Fixed/Swept (for all Segments)</td>
<td>OutputFixedFreq</td>
</tr>
<tr>
<td></td>
<td>OutputStartFreq</td>
</tr>
<tr>
<td></td>
<td>OutputStopFreq</td>
</tr>
</tbody>
</table>

**Conditional Columns: Based on IFRangeMode Value (dual stage only)**

<table>
<thead>
<tr>
<th>Fixed (for all Segments)</th>
<th>IFFixedFreq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swept (for all Segments)</td>
<td>IFStartFreq</td>
</tr>
<tr>
<td></td>
<td>IFStopFreq</td>
</tr>
<tr>
<td>Fixed/Swept (for all Segments)</td>
<td>IFFixedFreq</td>
</tr>
<tr>
<td></td>
<td>IFStartFreq</td>
</tr>
<tr>
<td></td>
<td>IFStopFreq</td>
</tr>
</tbody>
</table>

**Conditional Columns: Based on LO2RangeMode Value (dual stage only)**

<table>
<thead>
<tr>
<th>Fixed (for all Segments)</th>
<th>LO2FixedFreq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swept (for all Segments)</td>
<td>LO2StartFreq</td>
</tr>
<tr>
<td></td>
<td>LO2StopFreq</td>
</tr>
<tr>
<td>Fixed/Swept (for all Segments)</td>
<td>LO2FixedFreq</td>
</tr>
<tr>
<td></td>
<td>LO2StartFreq</td>
</tr>
<tr>
<td></td>
<td>LO2StopFreq</td>
</tr>
</tbody>
</table>

1. If SegmentSweepMode is true and all the segments have InputRangeMode set to Fixed, then only InputFixedFreq column will be displayed in the CSV.
   If SegmentSweepMode is false and the non-segment configuration’s InputRangeMode is Fixed, the behavior will be the same. This behavior will be true for Output, LO1, IF and LO2 as well.

2. If SegmentSweepMode is true and all the segments have InputRangeMode set to Swept, then only InputStartFreq and InputStopFreq columns will be displayed.
If SegmentSweepMode is false and the non-segment configuration’s InputRangeMode is Swept, the behavior will be the same. This behavior will be true for Output, LO1, IF and LO2 as well.

3 If SegmentSweepMode is true and the segments have a mix InputRangeMode, i.e. segment #1 has Fixed and segment #2 has Swept, then InputFixedFreq, InputStartFreq and InputStopFreq columns will be displayed in the csv. This will be true for Output, LO1, IF and LO2 as well.

The following is an example of an SMC segment sweep csv file:

![Example CSV file](image.png)

For a procedure, refer to How to configure multiple segments using the GUI and Excel *.csv file.

**Apply**  Applies the settings for your mixer/converter test setup to the measurement. The mixer setup dialog box remains OPEN.

**OK**  Applies the settings for your mixer/converter test setup to the measurement. The mixer setup dialog box CLOSES.

**Cancel**  Closes the mixer setup dialog box and does NOT apply the settings.

**See Also**

How to measure a DUT with an Embedded LO
**Note:** A variation of this dialog is used in the Swept IMDx application. Learn more.

**Converter Stages** Select either 1 or 2-stage converters.

**Hardware Configuration**

Shows the path configuration for Port 3 and Port 4 switches. See Hardware Configurations.

**Add Source** Click to start the External Device Configuration dialog.

**Path Configuration** (PNA Only) Click to start the Path Configuration dialog.

**DUT Ports**

**VMC ONLY** - select the VNA port to connect to the DUT output. The DUT input must always be connected to VNA port 1 because of the need for a reference mixer on port 1.

All other converter applications - select VNA ports to connect to the DUT input and output.

**Fractional Multipliers**

The combination of (numerator / denominator) forms a fractional value that is multiplied by the input and LO frequency ranges. These values are used to calculate the response.
frequency of the VNA receiver for the converter output. Use the fractional multipliers to:

- simulate the action of harmonic mixers
- simulate the action of multipliers and dividers that may exist in your test setup
- tune the VNA receiver frequency to a harmonic of the mixer/converter

The range for the numerator and denominator of a fractional multiplier is from +1 to +10. Negative values are NOT allowed.

See Fractional Multiplier examples.

**LO1 and LO2**

Select *Not controlled* to allow an external source to provide a Fixed LO Frequency at all times. Otherwise, select an internal VNA source or External source to be used as the LO. Learn how to Configure an External Device (Source).

**See Also**

How to measure a DUT with an Embedded LO

Learn about these buttons.
Configures LO Power settings.

**Power ON (All channels)** Check to immediately turn ON or OFF ALL VNA internal RF Sources for all channels.

**LO1 Power** Sets the power level for LO1.

  **Source Leveling** When using internal source for LO1, always set to Internal.

**LO2 Power** Sets the power level for LO2.

  **Source Leveling** When using internal source for LO2, always set to Internal.

**Port Settings**

**Port 3** Used when an LO is supplied through the rear panel and out port 3.

**Source Attenuator** Specifies the port 3 attenuator. This attenuator affects the range of available power into the LO port of the DUT Learn more about Source Attenuation.

**Receiver Attenuator** Set the attenuation for the Port 3 receiver. Learn more about Receiver Attenuation.

**Port 4** Used when an LO is supplied through the rear panel and out port 4.
**Source Attenuator**  Specifies the port 4 attenuator. This attenuator affects the range of available power into the LO port of the DUT. Learn more about Source Attenuation.

**Receiver Attenuator**  Set the attenuation for the Port 4 receiver. Learn more about Receiver Attenuation.

**Swept Power Settings**  Set the power sweep setting.

**Path Configuration** (PNA only)  Click to launch the RF Path Configuration dialog.

---

**Note: VMC measurements using a VNA with Internal Second Source**

Source 2 is automatically configured to supply power to BOTH available ports simultaneously. This setting can NOT be changed.

In addition, power can be uncoupled to provide different power levels at each port. This feature allows power to be delivered to both the DUT LO and Reference Mixer LO without use of a splitter. See VMC setup.

---

**See Also**

How to measure a DUT with an Embedded LO

[Save...][Load...]  Learn about these buttons.

---

**Fractional Multiplier Examples**

**Example 1**

Use the LO fractional multiplier to replicate the action of the third-harmonic mixer so the VNA can accurately calculate the receiver frequency. The input and LO frequencies are known.
Enter these settings in the **Mixer Setup** dialog box:

- **Input Start Freq**: 30 GHz
- **Input Stop Freq**: 40 GHz
- **LO Fixed Freq**: 16 GHz
- **Mixer-Product Selector**: - (difference)
- **LOs**: 1
- **LO fractional multiplier**: 3/1
- **INPUT fractional multiplier**: 1/1

Click **Calculate Output**

Results:

- **Output Start Freq**: 18 GHz
- **Output Stop Freq**: 8 GHz

---

**Example 2**

Use the fractional multipliers to tune the VNA receiver frequency to the second harmonic of the mixer's 14 GHz fundamental output. The input, LO, and output frequencies are known.

![Diagram](image)

Enter these settings in the **Mixer Setup** dialog box:

- **Input Start Freq**: 4 GHz
- **Input Stop Freq**: 4 GHz
- **LO Fixed Freq**: 10 GHz
- Mixer-Product Selector: + (Sum) of the input and LO signals
- LOs: 1
- INPUT fractional multiplier = 2/1
- LO fractional multiplier = 2/1

Click **Calculate Output**

Results:

- **Output Start Freq**: 28 GHz
- **Output Stop Freq**: 28 GHz

---

**Example 3**

Use the LO fractional multiplier to replicate the action of the divide-by-two mechanism inside the mixer package. Having done this, the VNA can accurately calculate the receiver frequency. The input and LO frequencies are known.

Enter these settings in the **Mixer Setup** dialog box:

- **Input Start Freq**: 45 MHz
- **Input Stop Freq**: 50 MHz
- **LO Fixed Freq**: 670 MHz
- **Mixer-Product Selector**: + (Sum) of the input and LO signals
- **LOs**: 1
- **INPUT fractional multiplier**: 1/1
LO fractional multiplier = 1/2

Click Calculate Output

Results:

- Output Start Freq: 380 MHz
- Output Stop Freq: 385 MHz
With SMC you can optionally measure phase. This feature is available with the S9x083A/B. SMC + phase is NOT offered in the N523xB.

In this topic:

- Overview
- How to make SMC+Phase measurements
- Measuring Mixers below 55MHz
- Comparing SMC+Phase with VMC Phase Measurements
- How to improve the stability of SMC+Phase measurements
- How to Calibrate a 2-stage (LO) SMC+Phase Measurement

See Also

Phase Reference Calibration
SMC Measurements and Calibration
SMC+Phase Demo (Internet connection required)

Overview

There are three methods used in the VNA to calibrate SMC+Phase measurements. All three methods rely on newer phase-coherent synthesizers in the VNA to produce phase capability in frequency offset measurements.

With the first two methods, during an SMC calibration, the phase delay through a Calibration Mixer is measured and compared to the known delay. The difference is used to correct subsequent SMC+Phase measurements.

**A Calibration Mixer is required with the first two methods.**

1. Enter the known delay into a dialog.
2. Uses the known delay at various frequencies from an *s2p file from a mixer characterization. You create the *S2P file from a separate Mixer Characterization. This method is NOT supported with Cal All.

3. Uses a Phase Reference to perform a ‘tier 1’ calibration. A Calibration Mixer is NOT required with this process. Learn more about this process.

Notes

- A Reference Mixer is NOT required with any of these methods as it is with VMC.
- SMC+Phase can be measured on Converters with an Embedded LO. Learn how.
- Phase can be measured with Power Sweeps.
- Phase can NOT be measured on Swept LO measurements.
- It is especially important with SMC + Phase measurements to connect the 10 MHz reference signal of an external source to the VNA.
- you can use a Phase Reference Cal Set with Independent power cals for input and output ports (no thru). Learn more.

How to make SMC+Phase measurements

1. On the Mixer Sweep tab, check Enable Phase, then select the Phase Reference Point.

   SMC Mixer Sweep tab - Phase Settings help

   Enable Phase with SMC  Check to perform phase measurements.

   You can also enable phase measurements at the SMC Cal Setup dialog.

   Phase Reference Point

   The SMC Phase measurement technique provides for a coherent phase relationship from one frequency to the next in each sweep. However, the phase measurement of the first data point is random from sweep to sweep. This initial phase offset does not impact measurements such as group delay or deviation from linear phase. However, in order to keep a phase trace from appearing random, all phase data in
the sweep is normalized against a single point. This results in a stable, normalized
phase trace.

For this normalization, select the measurement point that has the best signal-to-
noise ratio. The phase at the selected point will always be zero. This selection
applies for both the measurement and the calibration sweeps.

The Reference Point is fixed at the middle point when segment sweep is selected.

2. Select a Phase Format for the SMC measurement: click Response, then Format, then phase, unwrapped
   phase, or group delay.

3. During SMC Cal, at the Cal Setup dialog, select the Phase Correction method as follows:

   Enable Phase Correction  Check to enable Phase measurements. This can also be
done during the mixer setup.

   Phase Correction

   Choose one of the following methods to specify the known delay through the
characterized mixer. With the first two methods, the phase delay through a
Calibration Mixer is measured and compared to the known delay, either entered, or
stored in an *.S2PX file.

   • Use Known Mixer Delay  Enter the fixed, known, delay through the calibration mixer.

   • Use Characterized Mixer  Select, then browse to the *.S2P file that characterizes the
   calibration mixer. Use an *.S2PX file when making segmented SMC+Phase
   measurements. Learn more. Use either of the following two methods to characterize the
   Cal Mixer over the SMC measurement frequency range:

   1. Use the Mixer Characterization Wizard. The Cal Mixer has the same requirements as
      the VMC Cal Mixer. Learn more.
2. In a calibrated VMC channel, measure the group delay of the calibration mixer, then save to an *.S2P or *.S2PX file. However, a characterized mixer is required to calibrate the VMC channel.

- **Use Phase Reference Calset**: Select, then browse to the Phase Reference Cal Set that covers the frequency range of the current measurement. Learn more about SMC with a Phase Reference.

### Measuring phase on Mixers below 55 MHz

Measuring phase on mixers below 55 MHz is challenging for the following two reasons:

1. The PNA-X couplers have significant losses at these low frequencies. The PNA-X automatically reduces the IFBW at low frequencies in order to compensate for the coupler roll-off. And with SMC+Phase, we recommend setting 100 averages in order to reduce the noise of the SC21 delay measurement. These two settings combine to make the measurement speed VERY slow.

   - **Recommendation**: When measuring a downconverter with an output frequency below 55 MHz, check the “Reversed Port 2 Coupler” checkbox on the SMC Mixer Sweep tab and reverse the Port 2 coupler for the DUT measurement. With this setup, the IFBW is no longer reduced when measuring the output power. This significantly speeds up the measurement without reducing accuracy.

2. The PNA-X has a significant number of **frequency bands** below 55 MHz. SMC+Phase must have multiple measurement points within every frequency band.

   - **Recommendation**: Select the number of data points to yield a point spacing of NO MORE than 200 kHz per point. **Segment Sweep** can be used so that the point spacing above 55 MHz is not as dense.

### Comparing SMC+Phase with VMC Phase Measurements

SMC phase measurements do NOT require a reference mixer, and are therefore easier to make than VMC phase measurements. Also the SMC calibration mixer is only required to have a known delay value, although an S2P characterization file provides more accurate results. The Phase Reference method provides the highest accuracy phase measurements for mixer/converters.

- When measuring converters with an embedded LO, SMC with phase can provide results that are as stable and free from sweep-to-sweep jitter as VMC.

- When measuring converters with an external LO that is shared with the reference mixer (as shown in the **VMC Setup diagram**), VMC provides results that are more stable than SMC+Phase.

### How to improve the stability of SMC+Phase measurements

5799
Stable phase measurements are attained by increasing Sweep Averaging, and sometimes lowering the IFBW, until you attain the desired compromise between sweep time and trace jitter (the amount of random phase change at a single data point). For SMC+Phase, the default IFBW is 10 Hz, and 1 average. During calibration, the Averaging factor is temporarily multiplied by 4 to ensure an accurate phase calibration.

The following procedure shows how to view and improve phase jitter:

1. Create an SMC+Phase channel (Click Response, Measurement Class).
2. Enable Phase with SMC (See above) On the phase trace (to follow) notice that the only point that has NO jitter is the data point that you selected as the Phase Reference point.
3. Change the measurement to IPWR: (Click Response, Measure, IPWR)
4. Change Format to Phase. (Click Response, Format, Phase)
5. Normalize the trace. (Click Math, Memory, Normalize) Learn more about Normalization.
6. Autoscale the trace. (Click Response, Scale, Autoscale).
7. Optionally monitor the jitter with Trace Statistics (Std Dev)
8. Increase Averaging and possibly lower IFBW to improve jitter. (Click Response, Avg BW, then Averaging). Use a MINIMUM of 10 sweep averages (never use Point Averaging).
9. After the adjustments are made, change the measurement back to your measurement of interest.
10. When measuring a new DUT, restart Averaging.

How to Calibrate a 2-stage (LO) SMC+Phase Measurement

Note: The following discussion does NOT pertain when a Phase Reference cal is used to correct the SMC+Phase Measurement.

When calibrating a dual-stage SMC+Phase measurement for group delay using a characterized mixer, the channel setup requires frequency values for two LOs, but the characterized-thru mixer uses only one LO. The frequencies of LO1 and LO2 are different. There are two ways to overcome this challenge:

1. Before the calibration, set the LO that is provided by an external source to uncontrolled. Then manually set the frequency of this external source to the LO frequency that gives the same input and output frequencies, and the same sweep direction, as the dual-stage setup. Perform the calibration under this condition. After the calibration, return the LO to controlled so that its frequency will be properly set during the measurement of the DUT.
2. Configure a 1-stage mixer setup, with the LO set to the frequency that gives the same input and output frequencies and the same sweep direction as the dual-stage setup. Perform the calibration under this condition. Save the calibration data as a user calset. Configure the dual-stage case, and apply the 1-stage calibration.
Phase Reference Calibration

A Phase Reference calibration is performed to simplify the SMC+Phase or MODX (Modulation Distortion Converters) calibration process.

In this topic:

- Features, Requirements, and Limitations
- How it Works
- Hardware Setup
- Phase Reference Calibration Summary
- How to Perform the Phase Reference Calibration

**Other FCA Topics**

**Features**

- Greatly simplified phase measurements on mixers.
- A Reference Mixer or Calibration Mixer is **NOT** required.
- Works well with segmented sweeps, and mixers with 2-stage LOs.

**Requirements**

- One of the following Keysight Comb Generators (Phase Reference):
  - U9391C (26.5 GHz)
  - U9391F (50 GHz)
  - U9391G (67 GHz) - Receiver attenuation must be used.

**Note:** The U9391-60009 sine-to-square wave is recommended when using the instrument’s 10 MHz reference output as the driving source to the comb generator. Connect between the instrument’s 10 MHz output and the input to the comb generator. See the [U9391 Technical Overview](#) for more information.
● Power meter or USB power sensor.

● S-parameter Cal Kit (mechanical or ECal module).

● For measurements below 55 MHz, an additional 'Unknown mixer' is required. Learn more.

● For measurements between 50 GHz and 67 GHz, an additional high-pass filter is required for power calibration. Two back-to-back Keysight V281A waveguide-to-coax adapters is recommended.

**Note:** The 67 GHz comb generator creates a large pulse for generating useful harmonics up to 67 GHz. At low frequencies, receiver attenuation must be used to prevent overloading the VNA receiver due to the large signal. However, above 67 GHz, the extra receiver attenuation causes degradation in the signal-to-noise ratio (SNR) of the high harmonics, which would result in noisy phase calibrations. In order to prevent overloading and get good SNR, the receiver attenuation is removed and a high-pass filter is inserted between the comb generator and the VNA's test port. The calibration wizard provides extra steps to guide the user to insert the filter.

**Limitations**

● Available with opt. S93083A/B (NOT with opt. S93082A/B) Opt S9x083A/B ONLY available on a PNA-X or N522x

● Swept LO measurements are NOT allowed.

● Lowest frequency for Phase Reference Cal is 10 MHz.

**How it Works**

A Phase Reference Cal is performed, saved, and later recalled during a SMC or MODX Calibration. This is sometimes referred to as a 'tier 1" calibration. Due to stability of the VNA, the Phase Reference Cal can be performed infrequently. It is typically performed over the full frequency range of the VNA or Phase Reference so that it can be applied to all SMC or MODX calibrations that will be needed in the future.

The Phase Reference is a comb generator which produces signals at the multiple of its input frequency. By driving it with the VNA 10 MHz reference output, the Phase Reference produces signals at multiples of 10 MHz with a flat phase response.

During the tier 1 calibration, the VNA port 2 (B) receiver measures the phase at each 10 MHz comb frequency in order to generate correction values. These correction values are stored to a Phase Reference Cal Set which can be recalled and applied to correct subsequent SMC+Phase or MODX measurements.
Although only the B receiver is phase calibrated, an S-parameter calibration is performed at specified test ports and used to transfer the characterization to other VNA receivers.

A power calibration is also performed on port 1 as part of the Phase Reference Cal. It does NOT have to be performed again during the SMC or MODX Cal.

**Unknown Mixer Calibration (extends the phase reference calibration below 55 MHz)**

By itself, the phase reference can only be used to calibrate down to 55 MHz. With one additional unknown mixer connection, the phase reference calibration can be extended to the lower limit frequency which is 10 MHz and maximum frequency range of the VNA. The unknown mixer is also used to improve the quality of the phase reference calibration below 300 MHz.

With the unknown mixer process, the phase of the unknown mixer is measured as a down converter with the output frequencies set between 10 MHz and 600 MHz. By using the data acquired using the phase reference, the system is able to make produce calibrated measurements of the mixer above 300 MHz. The phase response of the unknown mixer is assumed to be linear, so the expected phase response of the mixer is extrapolated down to 10 MHz. The variation in the measurements versus the expected phase response is attributed to, and used to correct, the phase response of the VNA receivers from 10 MHz to 300 MHz.

**Unknown Mixer Properties**

- The mixer must be able to output a signal from DC to 1GHz.
- Over these frequencies, the delay response of the mixer should be constant. Therefore, the unknown mixer must be passive, with no filtering or amplification.
- Add a 10 dB attenuator on the mixer input and output to minimize mismatch effects.

**Hardware Setup**

1. Connect the U9391 Phase Reference to a DC power supply by one of the following cables. See the U9391 Technical Overview for more information.

   - The black wire is “ground”.
   - The red wire is +15Vdc. The U9391C and U9391F only require +15 Vdc.

2. Connect a cable from the VNA 10 MHz Reference signal output to the U9391 input. The cable should be as short as possible.
Note: The U9391-60009 sine-to-square wave is recommended when using the instrument’s 10 MHz reference output as the driving source to the comb generator. Connect between the instrument’s 10 MHz output and the input to the comb generator. See the U9391 Technical Overview for more information.

3. Connect the U9391 output directly to the VNA test port 2.

4. Insert the U9391 USB connector into any of the VNA USB ports.

5. Because of its low-frequency roll off, the coupler on Port 2 must be reversed which bypasses the coupled arm. Flip the port 2 front loops as shown in the following image. This increases power to the receiver by approximately 16 dB at low frequencies and removes noise from the measurement.

Configure the Port 2 'B' receiver port front-panel loops to a vertical orientation as shown here.

Block diagram showing port 2 thru coupler main arm to B receiver.

Phase Reference Calibration Summary

The Phase Reference Cal Wizard steps will show the appropriate prompts.

When Unknown Mixer is enabled (Start Frequency is 10 MHz)

- Reverse the Port 2 loops.

- The input and output match of the unknown mixer is measured. Then SC21 is measured using 100 averages.
Phase Reference measurement and S-Parameter calibration below 50 GHz.

- S-parameter cal is NOT performed when Omit Coupler Measurements is checked.
- Reverse the Port 2 loops.
- When using the 67 GHz phase reference, use receiver attenuator or external attenuator.

**When mixer output is above 50 GHz**, Phase Reference measurement and S-Parameter calibration.

- Connect 50 GHz high pass filter to phase reference output.
- Reverse the Port 2 loops.
- Receiver attenuator or external attenuator is NOT used.

Full band power sensor measurement.

- Connect Power Sensor to Port 1.
- Port 2 coupler in normal position.

Full band S-parameter calibration.

- Connect standards to Port 1.
- Port 2 coupler in normal position.
- High-pass filter is NOT used.
- Receiver attenuator or external attenuator is NOT used.
How to start the Phase Reference Calibration

With an Scalar Mixer / Converter + Phase or MODX measurement active...

Using Hardkey/SoftTab/Softkey

1. Press Cal > Main > Other Cals > Phase Reference Wizard....

Using a mouse

1. Click Response
2. Select Cal
3. Select Other Cals
4. Select Phase Reference Wizard...

Phase Reference Cal Settings dialog box help

Use Unknown Mixer

Check to use an Unknown Mixer to calibrate the VNA receivers below 55 MHz. The Start Frequency becomes 10 MHz and can not be changed. Learn how it works.

Ports

- Ports 1 and 2 are always selected.
- Select port 3 and more to perform an S-parameter (SOLT) cal at those ports. The phase correction is transferred to all checked ports.

Start / Stop Frequency

Select the start and stop frequency of the Phase Reference Cal. Subsequent SMC+Phase or MODX calibrations MUST be the same or a subset of these frequencies. Interpolation is
performed when the 10 MHz 'grid' of characterized frequencies is off from measurement frequencies.

- When Unknown Mixer is NOT checked, the lowest start frequency is 55 MHz. The lowest stop frequency is 1 GHz.
- When Unknown Mixer IS checked, the start frequency is 10 MHz and can NOT be changed. The lowest stop frequency is 5 GHz.

### Source Attenuation

Select the attenuation value for the VNA source port.

#### Important Source Attenuation Notes

- This setting should match the port 1 source attenuator setting used for subsequent SMC+Phase or MODX measurements. If the settings are the same, only an S-parameter cal need be performed with the SMC+Phase or MODX calibration wizard to complete the calibration process. If the settings are different, a Cal All calibration must be performed, which requires both an S-parameter cal and another power-sensor cal. As an alternative, if you plan to make SMC+Phase or MODX measurements with different input attenuator settings, you may want to perform a couple of Phase Reference cals using these different attenuator settings.

- The phase reference cal is not as good when using input attenuation. You will get a better calibration if you use 0 dB of source attenuation. Then use the Cal All feature to perform the 2nd tier calibration with the required input attenuation. This is because the Cal All feature includes an attenuator calibration that will properly move the phase reference cal from the 0 dB attenuator plane to the new (required) attenuator plane.

### Phase Reference

When the Phase Reference is detected by the VNA, it should appear in this field. If it does NOT appear here, try a different USB port, then click **Refresh**.

**Phase Reference Connector type and gender.** When the Phase Reference does NOT mate directly with the test port, you can add a well-behaved, broadband (no filters, waveguide, etc.) adapter to the Phase Reference and ignore it. That is because the adapter will stay with the Phase Reference and the calibration plane will still be the VNA test port connectors. The adapter adds constant delay; the deviation in delay is what is being calibrated out. Loss in the adapter is not a concern because the adapter is not used during the S-parameter and the power meter portion of the cal.

### Cal Kit Connector

Select the connector type and gender of your Cal Kit.
Only ONE connector type and gender is specified. This is because the entire Phase Reference cal is performed at the VNA test ports to reduce the effects of cable flex in the characterization.

There may be times when an adapter or connector-savers are used to connect the phase reference and cal standards to the test ports. In these cases, for highest accuracy use that adapter style and gender for ALL connections to ALL test ports. The effects of the adapter will be de-embedded automatically during subsequent SMC+Phase or MODX calibrations.

**Note:** For highest accuracy, perform the phase reference cal at the test ports.

**Cal Kit**  Select the Cal Kit that will be used to perform the S-parameter Cal.

**B Receiver Attenuation**

The 67 GHz (U9391G) phase reference outputs too much power and overloads the VNA Port 2 (B) receiver. Use at least 16 dB and not more that 20 dB of receiver attenuation to make accurate measurements using that phase reference.

This message is shown when a 67 GHz phase reference is connected to USB.

The B receiver attenuator control is shown if your VNA has receiver attenuators. Otherwise, connect external attenuation to the U9391G output when prompted.

**Omit Coupler Measurements**

This choice can be selected when the stop frequency is less than or equal to 50 GHz. Enabling the checkbox simplifies the calibration procedure by reducing the number of calibration steps required, with some tradeoffs. Clearing the checkbox will always result in the most accurate phase-reference calibration.

With the checkbox selected:

- Port 2 can NOT be used as the input to the DUT when setting up subsequent SMC+Phase or MODX measurements.
- The power-sensor calibration is performed at the end of a test cable connected to Port 1, which may degrade the accuracy of conversion loss/gain measurements due to cable movement during the calibration. This problem can be minimized by using a high-quality cable and avoiding cable movement between calibration steps.
Unknown Mixer Settings (Phase Reference Cal Wizard) dialog box help

The following wizard pages appear when Unknown Mixer is checked on the previous page:

**Select LO source**  The LO can be an internal 2nd source or an external source.

**Sources**  Click to start the External Devices dialog where you can select or configure an external RF source.

**LO Frequency**  Select a frequency that results in the unknown mixer output of 10 MHz to 600 MHz using this formula: Input = LO + Output.

**Note:** For best results, use the default LO frequency which avoids the VNA input (source) frequency band crossings.

The Port 2 coupler must be reversed for these measurements. Learn how.

- Connect the unknown mixer input to the between port 1 and port 2. For best results, connect the mixer output as close as possible to port 2.
- Connect the LO of the unknown mixer.
- Click **Measure**.
  - The input and output match of the unknown mixer is measured, then, SC21 using is measured using an averaging count of 100.
  - The data point spacing of the calibration is decreased to 156 kHz. This point spacing is necessary to accurately characterize all the low frequency band breaks of the VNA.
Power Cal Settings (Phase Reference Cal Wizard) dialog box help

A power calibration is performed on port 1. This is done to simplify subsequent SMC or MODX calibrations since the power calibration will not need to be repeated.

For highest accuracy, connect the power sensor directly to the test port with no adapter.

Learn more about these settings.

Phase Reference Calibration Steps dialog box help

- Connect the Phase Reference directly to the port 2 test port.
  
  a. If an adapter is necessary, then that adapter must be left in place for the SOLT cal at port 2.
  
  b. When using a 67 GHz phase reference, attenuation or high-pass filter is required.

- Ensure that the port 2 coupler is reversed. Learn how.

- Press Measure.

- The VNA will make a series of measurements which can take several minutes:
  
  a. First, the match of the phase reference is measured.
b. Then, a sweep is made at multiples of 10 MHz across the entire frequency span. Each measurement is repeated 100 times and averaged to reduce the noise in the measurement.

c. An error is reported if the approximate power level is not detected.

- 5. When finished, press Next>

### Power sensor A

Connect the power sensor to port 1 test port.

**SOLT cals are performed at the test ports.**

- The THRU standard always performs an Unknown Thru.
- Follow the prompts to complete the Phase Reference Cal.

### Phase Ref Cal Finished

Enter a name for the Phase Ref Cal, then click Finished.

During subsequent SMC or MODX calibrations, select the Phase Reference Cal Set at the SMC or MODX Cal setup dialog. Learn more.
If you know the delay value of the Unknown Thru connection, enter it here.

Otherwise, click **OK** to accept the calculated value.
How to make an SMC Fixed Output Measurement

The following is a step-by-step example illustrating how to measure a 1-stage mixer in swept LO mode using FCA Scalar Mixer Calibration.

There are fewer components required for SMC as compared to VMC, and fewer measurement steps. You can now make relative phase measurements with SMC. Also, ONLY SMC (not VMC) can measure the reverse conversion loss of the mixer.

This procedure can also be used for making fixed LO measurements, which is quite similar. Although a second source is still required, when using an external source, the physical triggering cables between the VNA and External Source are not required.

Required Equipment

- PNA-X or PNA ‘C’ models
  - with option S93083A/B (FCA) or option S93082A/B (SMC)
- GPIB External Source. Not necessary when using PNA-X with Internal Second source.
- ECal module with connectors that match the Input and Output connectors of the DUT. You can use adapters to make the ECal module match the DUT connectors, but first perform an ECal user-characterization with the adapters attached. ECal makes the FCA calibration much easier.
- GPIB or USB power meter / sensor
- Cables and adapters

Note: This procedure refers to an External Source to control the LO. If using a PNA-X with an Internal second source, an external source is not necessary. Connect the LO directly to the second source output.

The example mixer

The example device is a down-converter mixer with the following characteristics:

- LO and Input Frequency Range: 2 GHz to 4.2 GHz
- Output Frequency Range: DC to 1.3 GHz

We will measure:
- Fwd Conversion Loss (SC21)
- Input Match (S11)
- Output Match (S22)
- Reverse Conversion Loss (SC12)

**SMC Setup**

Connect the devices as shown in the following diagram:

The DUT can be connected to any VNA ports. Learn more.

This procedure uses DUT input to VNA port 1 and output to VNA port 2.

This procedure refers to an External Source to control the LO. If using a PNA-X with an **Internal second source**, an external source is not necessary. Connect the LO directly to the second source output (port 3 or port 4 on a PNA-X).

**Make Connections on the rear panels:**
1. If using a GPIB power meter, connect the power meter to the external source GPIB connector. If using a USB power meter, connect it to any unused USB port.

2. If using a PNA-X with an Internal second source, the following three steps are not necessary. Connect the LO directly to the second source output.

3. Using a GPIB cable, connect the VNA GPIB controller port to the external source GPIB connector.

4. Using two BNC cables, connect the Source and VNA Trigger connectors as shown in the following image. This is not necessary when making fixed LO measurements.

5. Using a BNC cable, connect the VNA 10 MHz Reference Output to the Ext. Source 10 MHz Reference Input.

![Diagram of Rear Panel Connectors]

Create the Measurement

1. Connect the DUT.

2. Press Setup > External Hardware > Power Meter Setup....

3. Under Interface, select GPIB, then enter the power meter address. Or select USB, then select the USB power meter that is connected to the VNA.

4. Press Preset to make sure you are starting with a known state.

5. Press Meas > S-Param > Meas Class....

6. Press Meas > S-Param > SC21 to replace the S11 trace.

Configure the Mixer settings
1. Press **Freq > Main > SMC Setup...** to start the SMC Setup dialog.

2. On the **Mixer Frequency** tab, enter the Mixer setup values as shown in the image below.

   ![Mixer Frequency Tab](image)

### Notes:

- Rather then enter ALL of the frequency settings, you can enter the Input and the Output frequencies, then click **Calculate LO**.

- If **Input>LO** is NOT checked, the VNA assumes you want the Input < LO frequencies, and higher LO frequencies are calculated as a result.

- The LO power level setting specifies the power out of the external source; not at the DUT) unless an LO power cal is performed.

- When the settings are valid, the background color around the **Apply** button is available.

### Configure the LO Source

1. On the **Mixer Setup** tab:

2. Change LO1 to either an internal VNA source or a pre-configured external source.

3. To configure an external source:

   a. Click **Add Source** in the upper-right corner of the Mixer Setup dialog.

   b. Complete the **External Source Configuration** dialog.

   c. If there is a problem communicating with the source, the VNA will display an error. **See Problems?**

   d. Click **OK** to return to the Mixer Setup dialog.

   e. The new external source is now available as an LO1 selection.

4. Save the mixer settings in a file so you can recall them easily. Click **Save...**, then type a descriptive filename, such as “FixedOutputMixer”.

5. The trace should begin to sweep as the external source steps in frequency. It should look something like this:
Problems?

Not sweeping:

- On the VNA, press Trigger > Main > Continuous to start the VNA sweeping. Watch for error messages on the VNA and source.

Problems communicating with the source:

- Press Setup > External Hardware > External Device... In the External Device Configuration dialog, select your LO source, click on Device Properties..., then click Software CW as the Trigger Mode, then close the dialog. Again, press Trigger > Main > Continuous to start the VNA sweeping. If this works, then something is wrong with Hardware (BNC). Check the trigger cables on the rear panel.

- Can the VNA communicate with the power meter? If not, there is something wrong with the GPIB or USB communication.

- As a last resort, try rebooting the VNA. First, save the entire setup to a .csa file. When the VNA preset measurement appears, recall this .csa file and continue at this step.

If the source is sweeping, and the VNA Input is sweeping, but there is still no output.

- Check power levels at the LO and Input.

- Check the DUT by making a fixed LO measurement - much easier.

Tip: You can optionally calibrate the LO Power level at the DUT using a standard Source Power Calibration. Select the source port in the Source Power Cal dialog.
Perform an SMC calibration

1. Disconnect the DUT.

2. Connect the ECal module to a VNA USB port.

3. Press Cal > Main > Smart Cal... Because the SC21 measurement is active, the Cal Wizard automatically begins an SMC calibration.

4. At the Calibration Setup dialog, click Next.

5. At the Select DUT Connectors and Cal Kits dialog, for DUT Port 1 select the connector type and gender of your DUT INPUT. For DUT Port 2 select the connector type and gender of your DUT OUTPUT. Then select ECal as the Cal Kit to use for each connector. Click Next.

6. At the Scalar Mixer Calibration Step 1 of 2 dialog, connect the power sensor to the Port 1 test cable, then click Measure. The data will be used to correct for input mismatch errors.

7. At the Scalar Mixer Calibration Step 2 of 2 dialog, connect the ECal module Port A to the Port 1 cable, and Port B to the Port 2 cable. Then click Measure. This portion of the calibration gathers the linear (non-frequency-translating) error terms of the test setup at the input and output frequencies.

8. At the Calibration completed dialog, you can choose to save the SMC calibration as a User Cal Set. Otherwise, click Finish to complete the SMC calibration. Correction is turned ON and applied to the SMC trace.

What is happening?

When an external source is sweeping, the measurements are much slower. When correction is ON, you will see that there are times when nothing is happening on the screen. This is because there are background measurements being made but not displayed.

This is exactly the same as when full 2-port correction is applied to an S-parameter. All four parameters are measured, then correction is applied, then all four measurements are updated. This occurs much faster when there is no external source. With correction OFF, the traces are updated as the data is measured. You can see this taking place by creating the following measurements.

Create S12 Upconverter, S11 Input and S22 Output Match

1. Right-click in the window and add new traces to the same channel. Add S11, SC12, and S22 measurement traces.

2. While the source is sweeping, watch the source port indicator on the front of the VNA. First, the port 1 indicator will light for two sweeps, then the port 2 indicator will light for 2 sweeps. During the last sweep, all 4 traces update.

3. Press Cal > Main > Correction Off. Notice that the relevant traces update as the sweep is occurring.
With the SC12 measurement you can see the reciprocity of the mixer.

SMC forward and reverse measurements can reside in the same channel and are calibrated automatically at the same time.
SMC with a Booster Amp

If your mixer measurement requires more source power on the input than the VNA can provide, a booster amplifier can be used to provide the additional power. This topic describes how to configure and make a calibrated SMC measurement using a booster amplifier.

Connect

Connect the booster amplifier between the Source-Out and Coupler-Thru connectors on the front-panel as shown in the following diagram. Your VNA block diagram may not look like this.

Measurement and Calibration Setup

In the following procedure:
- **Test Port power** is the power level out of the source.
- **Corrected power** is the power level you require at the mixer input and output.

This procedure assumes you will applying stimulus power to the mixer **input** to make SC11 and SC21 measurements, and to the **output** of the mixer to make SC22 and SC12 measurements.

1. Determine the gain of the booster amplifier. If the gain has significant slope across the **input and output range** of the mixer, see Booster Amp with a Gain Slope.

2. Determine the corrected power for both the input (port 1) and output (port 2) of the mixer.

3. Calculate the Test Port power for both ports by subtracting the gain of the amplifier from both the input and output corrected power levels.

   For example, the following values assume a 25 dB booster amp on port 1 as in the diagram above.

<table>
<thead>
<tr>
<th>Port</th>
<th>Corrected Power</th>
<th>Amp Gain</th>
<th>Test Port Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1 (input)</td>
<td>0 dBm</td>
<td>-25 dB</td>
<td>-25 dBm</td>
</tr>
<tr>
<td>Port 2 (output)</td>
<td>-20 dBm</td>
<td>-25 dB</td>
<td>-45 dBm</td>
</tr>
</tbody>
</table>

4. On the VNA **Power dialog**, clear the **Port Power Coupled** checkbox, which allows different power levels for each port.

5. Enter the calculated **Test Port Power** values for each port.

6. During the SMC Cal Wizard **Select DUT Connectors and Cal Kits** dialog, click **View/Modify Source Cal Settings** to invoke the **Source Calibration Settings** dialog.

7. In **Power Offset**, enter the booster amplifier gain.

**Booster Amp with a Gain Slope**

SMC calibration takes place over the entire input and output range of the mixer. Therefore, the booster amplifier will also be subjected to the entire input and output frequency range of the mixer.

To compensate for a gain slope, you might have to experiment with the source attenuator setting, power-offset value, and initial power value to get a combination that will not cause the VNA source to go unleveled during or after the cal.

For example, assume the booster amp gain is 30 dB at the low end, and 20 dB at the high end. If you
enter 30 dB for the power offset value, the VNA might run out of ALC range when the actual gain drops to 20 dB. The VNA will try to increase its source power to account for the 10 dB gain drop. Therefore, pick a power offset value that is in the middle of the amplifier gain band (25dB).

If possible, select a VNA attenuator setting that puts the ALC approximately in the middle of its range at the desired corrected power with the mid-band gain. This condition means the ALC can set the power higher and lower to account for the gain slope, without unleveling.

If the gain slope is too large, then there may not be a setting that prevents a source unlevel. In this case, a flatter booster amp must be used.
VMC Measurements

VMC Setup and Calibration is very similar to SMC. See FCA Overview to learn about the features that are common to these two applications.

The following information is unique to VMC:

- VMC Hardware Setup
- Create a VMC Measurement
- VMC Parameters Offered
- The VMC Mixer Setup dialog
- VMC Calibrations

See Also

Embedded LO

How to make a VMC Fixed Out measurement

VMC Hardware Setup
DUT Input (RF) must be connected to VNA port 1.
DUT Output (IF) can be connected to any other VNA port.

Notes:

- When using a PNA-X with an Internal Second Source, the external source is NOT necessary.
- See note regarding LO power out both second source ports
- Learn which VNA ports can be used for the LO.
- Measure a DUT with an Embedded LO

Reference Mixer

The Reference mixer provides a phase reference for the measurements. The reference mixer is connected in the reference receiver path of the network analyzer, between the source out and receiver R1 in ports, as shown below.

The reference mixer is considered part of the test system setup like the test cables. It remains in place during the entire calibration and measurement process. The reference mixer is switched in and out of the measurement path by the VNA as needed. See how to manually switch the reference mixer.

The reference mixer does not need to be reciprocal and does not have to match the calibration mixer or the mixer-under-test in performance. The only requirement of the reference mixer is that it cover the same frequency range as the mixer under-test. In general, it is valuable to select a reference mixer that can be used with a variety of different setups. For example, a broadband mixer can be used in place of several narrow-band alternatives.

A low pass filter on the output of the reference mixer can be used to suppress the LO leakage signal that comes out of the reference mixer output. It is not strictly needed, but ensures that the VNA will not have any source unlock or unlevel errors due to the LO leakage.

- Connect the Reference Mixer INPUT to VNA Ref 1 Source out
- Connect the Reference Mixer OUTPUT to VNA Rcvr R1 In

Calibration Mixer/Filter

The Calibration mixer/filter is characterized either before or during a VMC calibration. It is used during the VMC calibration as the THRU standard. The calibration mixer/filter combination must meet the following requirements:

- The mixer must be reciprocal over the frequency range of the mixer under test. This means that it has the same magnitude and phase response in the up-converting and down-converting directions (C21 = C12) as
If the Input and Output frequency ranges are overlapping, the mixer must have Input to Output Isolation greater than 10 dB more than the conversion loss in the overlapping range.

The filter must reject the undesired mixing product, and pass the desired mixing product, at the output of the cal mixer. This requirement can be made easier by characterizing the mixer/filter as a downconverter.

**Note:** With a corrected VC21Swept LO measurement, the phase data is displayed relative to the phase of the calibration mixer that was used during the VMC calibration. In addition, Group delay display format is NOT valid.

See an example of a Fixed Output VMC Measurement

**Important note: Orientation of Reference mixer and Calibration mixer/filter**

The reference mixer is ALWAYS connected in the same orientation as the DUT, since the output frequency of the reference mixer has to match that of the DUT. The same applies to the calibration mixer/filter if it is characterized as part of a full VMC cal.

If you characterize the calibration mixer/filter separately, you can characterize it as either an upconverter or downconverter.

**LO Source**

**Note:** When using a PNA-X with Internal Second Source, the external source is NOT necessary.

- See note regarding LO power out both second source ports
- Learn which VNA ports can be used for the LO.
● Connect **External Sources** to the VNA GPIB Controller port.

● Learn how to **Configure an External LO Source**

**Create a VMC Measurement**

1. Press **Setup > Main > Meas Class...**

2. Select **VMC**, then either:
   - **OK** delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.

3. A VC21 measurement is displayed.

4. To select additional parameters to display, press **Trace > Trace 1-8 or Trace 9-16 > Trace N.** to add additional traces.

5. Right-click on the trace, select **Measure TrN...**, then select a parameter from the list then click **OK**.

**VMC Parameters Offered**

**Important Note:** Connecting your DUT to the VNA:

**RF** and **IF** terminology is NOT used in FCA because the VNA does not know how the DUT is labeled or how it will be used. Instead, the general terms **INPUT** and **OUTPUT** are used.

- **INPUT** - The DUT port being stimulated with frequencies before conversion.
- **OUTPUT** - The DUT port outputting converted frequencies.

**INPUT** and **OUTPUT** Frequencies are specified using the **Mixer Setup dialog box**.

The DUT input is always connected to VNA port 1. However, the DUT output can be connected to any other VNA port.

- **VC21, VC31, or VC41 Conversion Loss/Gain (default)** - stimulus at Input, response at Output
- **S11** - stimulus and response at Input
- **S22, S33, or S44** - stimulus and response at Output
- **R1 (or R)** - stimulus at Input, measures absolute power at the R1 receiver (uncorrected)
- **B, C, or D** - stimulus at Input, measures absolute power at the output receiver (uncorrected)
- Reverse conversion loss is NOT offered because of the reference mixer.

### See Also

Measure a DUT with an Embedded LO

### VMC Mixer Setup

#### How to start the VMC Mixer Setup dialog

**Using Hardkey/SoftTab/Softkey**

1. Press *Freq > Main > VMC Setup... > Mixer Setup* tab.

**Using a mouse**

1. Click **Stimulus**
2. Select **Sweep**
3. Select **VMC Setup...**
4. Select **Mixer Setup**

The following VMC Mixer Setup dialog tabs are presented:

- **Sweep Tab** (shared with SMC)
- **Power Tab** (shared with SMC)
- **Mixer Freq Tab** (shared with all converter apps)
- **Mixer Power Tab** (shared with all converter apps)
- **Mixer Setup Tab** (shared with all converter apps)

### VMC Calibration Overview

The Calibration Wizard guides you through this process. The first three steps characterize the calibration mixer that is used as the THRU standard during the calibration process.

1. Perform a **2-port SOLT calibration** over the INPUT frequency range of the DUT, and another **2-port SOLT calibration** over the OUTPUT frequency range. Use either a mechanical calibration kit or an ECal module.
2. Characterize the input and output match of the calibration mixer/filter combination with the external LO connected and the output terminated with an open, short, and load. Learn how to connect the calibration mixer/filter. Once characterized, an S2P file is saved and can be recalled for use in subsequent VMC calibrations using the same stimulus settings. **Note:** Use an *.S2PX file for SEGMENTED VMC measurements. Learn more.

3. Connect the reference mixer between the Source Out and Rcvr R1 front-panel connectors. Connect the output port of the calibration mixer/filter combination to VNA Port 2 (or at the end of the cable attached to the port).

4. Measure the calibration mixer/filter combination as the THRU calibration standard.

5. The VNA calculates the error terms necessary to make corrected phase measurements of your mixer/ converter under test.

**VMC Cal Wizard**

The following dialog boxes are presented during VMC Calibration and VMC Mixer Characterization.

- Calibration Setup
  - Waveguide/In-fixture/On-Wafer Setup
- Calibration Mixer Characterization
- Measurement Direction
- Select DUT Connectors and Cal Kits
  - Modify Frequency Cal
  - Specify how the ECal module is connected
  - Modify Mixer Cal
  - Select the ECal Port to be connected to the Output of the Calibration Mixer
- Vector Mixer Cal Steps
- Measure Calibration Standards
  - Save Mixer Characterization
- Calibration Completed
How to Perform a VMC Calibration

1. Create an FCA measurement, then...

Using **Hardkey/SoftTab/Softkey**

1. Press **Cal > Main > Smart Cal...**

Using a mouse

1. Click **Response**
2. Select **Cal**
3. Select **Smart Cal...**

To perform Mixer Characterization ONLY

1. Press **Cal > Main > Other Cals > Mixer Char Wizard...**

1. Click **Response**
2. Select **Cal**
3. Select **Other Cals**
4. Select **Mixer Char Wizard...**

---

**Calibration Setup** dialog box help

![Calibration Setup dialog box](image)

**Waveguide/In-fixture/On-Wafer Setup**  Check to embed or de-embed circuit networks on the input and output of your mixer under test. Starts the following dialog box.

**Additional Power Cal Steps**
**Enable LO1 / LO2 Power Cal**  Check when LO1 / LO2 is controlled (on the Mixer Setup tab) to perform a Power Cal on the LO source(s).

**Waveguide/In-fixture/On-Wafer Setup** dialog box help

This dialog box appears ONLY if you checked the Waveguide/In-fixture/On-Wafer Setup box in the previous Cal Setup dialog.

Allows you to embed or de-embed circuit networks on the input and output of your mixer under test.

For Network1 (Input) and Network2 (Output) select **Embed, De-embed, or None**.

**Browse**  Click to navigate to the .S2P file that models the network to embed or de-embed.

**Reverse port positions for input/output**  Check to cause the Fixture/Adapter to be configured with Port 2 connected to the VNA and Port 1 to be connected to the DUT. The image in the dialog is updated to reflect that change.

**Enable Extrapolation**  Check (default setting) to apply a simple extrapolation when the S2P file has a narrower frequency range than the channel. The values for the first and last data points are extended in either direction to cover the frequency range of the measurement. A warning message is also displayed when extrapolation is necessary.

**To Embed or De-embed**

- When you have a 2 port network that needs to be connected between the Cal reference plane and the DUT during the measurement, but it is NOT present during the calibration, then that network has to be **De-Embedded** from the port in question during the calibration. In other words, De-Embedding in FCA calibration extends the calibration reference plane to include the two port network.

- When you have a 2 port network that is included as part of the calibration reference plane but has to be disconnected during the measurement, then that 2-port network has to be **Embedded** for the port in question during the calibration. In other words, Embedding in FCA calibration retracts the calibration reference plane to exclude the two port network during the measurement.
Notes

- Interpolation is performed when more frequencies are included in the file than in the channel, and the
data points do not exactly match those of the measurement.

Calibration Mixer Characterization dialog box help

What is Calibration Mixer Characterization? For a brief explanation, see Calibration Mixer.

Select Mixer Characterization Method

Perform Characterization (requires a reference mixer) Performs a Mixer characterization in
addition to the VMC calibration. The mixer characterization file will be saved at the end for use
in subsequent VMC calibrations. Choose this selection if you do NOT already have a mixer
characterization file to load.

Load characterization from file Loads an S2P calibration mixer characterization file. Click
Browse to locate the file.

Note: Load an *.S2PX file for SEGMENTED VMC measurements. Learn more.

- The frequency range of the S2P file MUST be the same, or larger than, the frequency range
  of the FCA measurement. If the S2P file frequency range is larger, or the data points do not
  exactly match those of the measurement, interpolation will be performed.

- The VMC calibration requires that the calibration mixer be connected in the same
  orientation as that in which it was characterized. The direction in which it was characterized
  is not part of the file that is recalled. You have to remember and connect it appropriately.

"Invalid Mixer Characterization File" is displayed if the frequency range of the S2P file is
smaller that those of the measurement.
**Note:** A Mixer Characterization Cal can be performed separately. Learn how.

**Measurement Direction** dialog box help

This dialog box appears ONLY if your settings in the Mixer Setup dialog box indicate that your DUT is being tested as an upconverter (input < output). It allows you to characterize the Calibration Mixer / Filter as a downconverter (input > output) or an upconverter.

The following example shows why you would choose to characterize the calibration mixer as a downconverter. Consider a DUT being used as an upconverter. The input frequency is 70 MHz, the LO is 20 GHz, and the selected (+) output frequency is 20.07 GHz. If we chose (-) in the mixer setup dialog, the output frequency would be 19.93 GHz.

- **Characterize as upconverter** A very sharp cutoff filter is required to reject the undesired output of 19.93 GHz and pass the desired 20.07 GHz.

- **Characterize as downconverter** The input frequency is 20.07 GHz; the LO is 20 GHz. The sum (+) output is 40.07 GHz and the diff (-) output is 70 MHz. These are very easy to separate with a low-pass filter. The original frequencies are always used in the downconversion process, so be sure to choose a filter that will pass 70 MHz and reject 40.07 GHz.

See connection diagrams.
Select DUT Connectors and Cal Kits dialog box help

Allows you to specify the connector type and Cal Kit for each DUT port.

**Port n** For each listed VNA port, specify the DUT connector type and gender, and the Cal Kit to use.

**Mixer Out Port** Output port of the image filter that is connected to the calibration mixer. Specify the Cal Kit / standards to use for the measurement of the calibration mixer / filter combination.

**Note:** When selecting a cal kit for the Mixer Out Port, be sure that the kit has standards with connectors that can mate to the mixer output port. If you choose an ECal, the ECal must have at least one port that can mate to the mixer output port.

**Note:** If your DUT connectors are:

- **Waveguide** Change the system impedance to 1 ohm before performing a calibration. See Setting System Impedance.

- **Not listed** (male and female) Select **Type A** as the connector type. Type A requires a calibration kit file containing the electrical properties of the standards used for calibration (see Calibration kits).

- **Unspecified** (like a packaged device) Select **Type B** as the connector type. Type B requires a calibration kit file containing the electrical properties of the standards used for calibration (see Calibration kits).

**Modify Cal** Check to start the Modify Cal dialog. If performing a Mixer Characterization Cal at the same time as VMC Cal, two Modify Cal dialogs will be presented, one after the other.
**Modify Frequency Cal** dialog box help

For VMC calibrations - NOT for Mixer Characterization.

**Thru Cal Method**  For each Thru connection, choose the Thru method. Learn more about these choices.

**Cal Type/Std**  Click to start the Modify Calibration Selections dialog box.

**The following selections are available ONLY if using an ECal module.**

**Do orientation**  When this box is checked (default) the VNA senses the ECal model and direction in which the ECal module port is connected to the VNA ports. If power to the ECal module is too low, it will appear as if there is no ECal module connected. If you use low power and are having this problem, clear this check box to provide the orientation manually.

Orientation occurs first at the middle of the frequency range that you are calibrating. If a signal is not detected, it tries again at the lowest frequency in the range.

**View/Detect ECal Characterizations**  Appears only if an ECal module is selected for use. Click to invoke the View ECal Modules and Characterizations dialog box. Displays a list of ECal modules that are connected to the VNA.

**Specify how the ECal module is connected** dialog box help

This dialog box appears when the Do orientation checkbox in the previous Modify Frequency dialog box is cleared.

Click the ECal Port that is connected to each VNA port.
Modify Mixer Cal dialog box help

**Mixer Characterization ONLY.** The Thru standard is not measured. Therefore, the Thru Cal Method choices are not available.

**View / Detect ECal Characterizations** Available ONLY if using an ECal module. Invokes the Select ECal Module and Characterization dialog box.

Select the ECal Port to be connected to the Output of the Calibration Mixer dialog box help

Select the ECal Port to be connected to the output of the image filter of the Calibration Mixer / Filter combination. See connection diagram of Calibration Mixer / Filter combination.
**Measure Calibration Standards** dialog box help

Prompts for standards to be measured. Connect the standard, then click **Measure**.

**Measure**  Measures the mechanical standard and continue to the next calibration step.

**[Re]Measure**  Replaces Measure after standard has been measured. Allows you to remeasure a standard.

**Done**  Click to proceed to the **Calibration Complete** dialog. Available only after all measurements for the calibration are complete.

**Back**  Returns to the previous dialog box.

**Next**  Does NOT make a measurement. Proceeds to the next required step.

**Cancel**  Exits the Calibration Wizard.

---

**Vector Mixer Cal Steps** dialog box help

Connect the Open, Short, and Load standards to the image filter output, then click **Measure**.

This portion of the calibration characterizes the calibration mixer.

The connection is different depending on if the calibration mixer is an upconverter being characterized as a down converter.
**Note:**

The following are simplified connection diagrams - the reference mixer and LO signals must also be connected. These images assume that the DUT output is connected to VNA port 2.

As a **Downconverter**. (The VNA automatically switches to make the S22 measurement on the device.)

As an **Upconverter**

**Done** Click to proceed to the **Calibration Complete** dialog. Available only after all measurements for the calibration are complete.
**Save Mixer Characterization** dialog box help

Allows you to save the characterization data of your calibration mixer. When performing another VMC calibration using the same calibration mixer, this S2P file can then be recalled.

**Browse** Navigate to the location where you want to save the characterization data of your calibration mixer. Either use the default file name or enter a custom file name.

**Next** Saves the mixer characterization file and continues with the next step in the full system calibration routine.

**Finish** Replaces Next if you are only characterizing the calibration mixer instead of performing a full system calibration. Saves the mixer characterization file and exits the mixer characterization routine.

---

**Calibration Completed** dialog box help

**Finish** Save to the channel's calibration register.

**Save As User Cal Set** Starts the Save as User Cal Set dialog box AND save to the channel's calibration register.

**Cancel** Calibration is NOT applied or saved.

Learn about Calibration Registers.

Learn about User Cal Sets
Specify delay dialog box help

This dialog appears ONLY when Adapter Removal or Unknown Thru calibrations are performed.

The following values were estimated from the measurement. Most of the time, they are adequate. However, for CW sweep or frequency sweep with large step sizes, the accuracy of the values may be improved.

**Adapter delay**  To improve this value, measure and record the delay of the adapter with a dense step size. Enter that value here. The required precision value is the accuracy that is required to characterize the delay value.

**Nominal phase offset**  (Waveguide ONLY). To improve this value, measure and record the phase offset of the Waveguide adapter with dense step size. Enter that value here.

When one connector is coax and the other connector is waveguide, the phase offset has an ambiguity of 180 degrees. For consistency, the estimate provided here is always between 0 and 180 degrees. You can change this estimate to any value between -180 degrees and +180 degrees.

**For FCA calibrations**, this dialog box appears twice: once for the input frequencies and once for the output frequencies. The values can be slightly different.
How to make a VMC Fixed Output Measurement

The following is a step-by-step example illustrating how to measure a mixer in swept LO mode using FCA Vector Mixer Calibration.

There are fewer components required for SMC as compared to VMC, and fewer measurement steps. You can now make relative phase measurements with SMC. Also, ONLY SMC (not VMC) can measure the reverse conversion loss of the mixer.

This procedure can also be used for making fixed LO measurements, which is quite similar. Although the LO source is still required, the physical triggering cables that connect the VNA and External Source are not required.

**Required Equipment**

- PNA-X or VNA 'C' models
  - with option S93083A/B (FCA)
- GPIB External Source (Keysight ESG or PSG works best). This source is NOT necessary when using PNA-X with Internal Second source.
- Reference Mixer (see requirements)
- Calibration Mixer/Filter (see requirements)
- Power splitter - Not necessary when using PNA-X with Internal Second source.
- ECal module with connectors that match the Input and Output connectors of the DUT. You can use adapters to make the ECal module match the DUT connectors, but first perform an ECal user-characterization with the adapters attached. ECal makes the FCA calibration much easier.
- Cables and adapters
- Optional GPIB Power meter and sensor (for LO power calibration)

**The example mixer**

The example device is a mixer with the following characteristics:

- LO and Input Frequency Range: 2 GHz to 4.2 GHz
- Output Frequency Range: DC to 1.3 GHz
We will measure:

- Fwd Conversion Loss (VC21)
- Input match (S11)
- Output match (S22)
- Rev Conversion Loss is NOT possible because of the reference mixer.

**VMC Setup**

Connect the devices as shown in the following diagram:

**Note:** This setup can also be used for SMC measurements, allowing you to make VMC and SMC measurements simultaneously on separate channels. The Reference Mixer is automatically switched during SMC measurements. The Cal Mixer/Filter is not used.
Notes:

- When using a PNA-X with an Internal Second Source, the external source is NOT necessary.
  - See note regarding LO power out both second source ports
  - Learn which VNA ports can be used for the LO.
- The low-pass filter on the output of the Reference Mixer is recommended, but NOT required. Learn more.
- When using 4-port models, the mixer input must be connected to VNA port 1. The mixer output can be connected to any other VNA port.

Make Connections on the Instrument rear panels:

1. Connect the Source to the VNA GPIB Controller port using a GPIB cable.
2. Using two BNC cables, connect the Source and VNA Trigger connectors as shown in the following image. This is not necessary when making fixed LO measurements, or using a PNA-X with Internal Second Source.
3. Using a BNC cable, connect the VNA 10 MHz Reference Output to the Ext. Source 10 MHz Reference Input.

Create the Measurement

1. Connect the DUT.
2. Press Preset to make sure you are starting with a known state.
3. Press **Meas > Main > Meas Class**.

4. Select **Vector Mixer/Converter** then **OK**. At the **Confirm**... dialog, click **OK**. An S11 trace is created.

5. Press **Meas > Main > VC21** to replace the S11 trace.

Configure the Mixer settings

1. Press **Freq > Main > VMC Setup**... to start the VMC Setup dialog.

2. On the **Sweep** tab, no changes from the default settings are required. The Avoid Spurs feature is useful for eliminating spurs in test setups with excessive LO leakage.

3. On the **Power** tab, change the DUT Input Port Power Level to -17 dBm.

4. On the **Mixer Frequency** tab, enter the frequency values as shown in the following image:

![Mixer Frequency Tab](image)

   a. You can enter the Input and the Output frequencies, then click **Calc LO**.

   b. If **Input > LO** is NOT checked, the VNA assumes you want the Input < LO frequencies, and higher LO frequencies are calculated as a result.

External Source Configuration

When using a PNA-X with Internal Second Source, the external source is NOT necessary.

* See note regarding LO power out both source ports

* Learn which VNA ports can be used for the LO.

1. On the **Mixer Setup** tab, click **Add Source** to start the External Source Configuration dialog. Learn how to configure an external source.

2. On the External Source Configuration dialog, click **Device Properties**.

   a. Click **Hardware List (BNC)**, which is the fastest measurement method. This method requires the BNC Trigger cables that connect the VNA and source. If not available, **Software CW** can be used, but measurements are much slower.
b. If necessary, select the Interface (usually GPIB), then Refresh, then select the Available IO Configuration.

c. Click OK to close the Device Properties dialog, then OK to close the External Source Configuration dialog.

3. Next to LO1, select the configured external source name.

**Complete Mixer Setup**

1. On the Mixer Power tab, change LO1 Power to 0.00 dBm. This setting specifies the power out of the external source (not at the DUT) unless an LO power cal is performed.

2. When the settings are valid, the Save, Apply and OK buttons are available to click. To save the mixer settings in a file so you can recall them easily, click Save..., then type a descriptive filename. Then click OK.

3. Click OK to close the VMC Setup dialog.

4. To change the X-axis values from the default Output selection (800 MHz), press Sweep > Main > X-Axis Type > Input.

5. The trace is updated as the external source steps in frequency. It should look something like the following image. Because of the reference mixer, the uncorrected VMC measurement can look like it has gain.
Problems?

Not sweeping:

- On the VNA, press **Trigger > Main > Continuous** to start the VNA sweeping. Watch for error messages on the VNA and source.

Problems communicating with the source:

- Press **Setup > External Hardware > External Device...** On the External Device Configuration dialog, select the external source, then click Device Properties. Next to Trigger Mode: select **Software CW trigger**, then close the dialog. Again, press **Trigger > Main > Continuous** to start the VNA sweeping. If this works, then something is wrong with **Hardware (BNC)**. Check the trigger cables on the rear panel.

- As a last resort, try rebooting the VNA. First, save the entire setup to a \*.csa file. When the VNA preset measurement appears, recall the \*.csa file to resume at this step.

If the source is sweeping, and the VNA Input is sweeping, but there is still no output.

- Check power levels at the LO and Input.
- Check the DUT by making a fixed LO measurement which is much easier.

Perform a VMC Calibration

1. Disconnect the DUT.

2. Connect the ECal module to a VNA USB port.

3. Press **Cal > Main > Smart Cal...**. Because the VC21 measurement is active, the Cal Wizard automatically begins a VMC Calibration.

4. At the Calibration Setup dialog, click **Next**. Or check Enable LO1 Power Cal to perform a Source Power Cal to specify the LO Power at the DUT. This requires a power meter or USB power sensor be connected.

5. At the Calibration Mixer Characterization dialog, click **Next**. We will perform characterization of the Calibration mixer as part of the VMC cal. Later we will save the Calibration mixer characterization so that, in future VMC calibrations that use this same frequency range, we can recall the Calibration mixer characterization by clicking **Load Characterization from file**.

6. At the Select DUT Connectors and Cal Kits dialog, for Port 1 select the connector type and gender of your DUT INPUT. For Port 2 select the connector type and gender of your DUT OUTPUT. Then select ECal as the Cal Kit to use for each connector. Click **Next**.
7. At the **Select the ECal Port to be Connected** dialog, ensure that **Port A** is selected for **Port 1**, then click **Next**.

8. At the **Vector Mixer Calibration Step 1 of 3** dialog, connect the ECal module Port A to the Port 1 cable, and Port B to the Port 2 cable. Then click **Measure**. This portion of the calibration gathers the linear (non-frequency-translating) error terms of the test setup at the input and output frequencies.

9. At the **Vector Mixer Calibration Step 2 of 3** dialog, connect the following, then click **Measure**. This portion of the calibration will connect reflection standards to characterize the S-parameters of the calibration mixer/filter.

   - Port 1 cable to the Input of the calibration mixer.
   - LO cable to the LO port of the calibration mixer.
   - ECal module to the Output of the calibration mixer/filter.

10. At the **Vector Mixer Calibration Step 3 of 3** dialog, disconnect the ECal module and connect the Port 2 cable to the output of the calibration mixer/filter, then click **Measure**. This step completes the calibration using the characterized mixer/filter as a Thru standard.

11. At the **Save Mixer Characterization** dialog, click **Browse**, then type a unique filename and click **OK**. Then click **Next**. This saves the Calibration Mixer characterization to an S2P file. This file can be recalled for subsequent VMC calibrations.

12. At the **Calibration completed** dialog, you can choose to save the VMC calibration as a User Cal Set. Otherwise, click **Finish** to complete the VMC calibration. Correction is turned ON and applied to the VMC trace that we set up earlier.

### What is happening?

When an external source is sweeping, the measurements are much slower. When correction is ON, you will see that there are times when nothing is happening on the screen. This is because there are background measurements being made but not displayed.

This is exactly the same as when full 2-port correction is applied to an S-parameter. All four parameters are measured, then correction is applied, then all four measurements are updated. This occurs much faster when there is no external source. With a VMC measurement, there is no VC12 (reverse transmission measurement), so there are only three background measurements. With correction OFF, the traces are updated as the data is measured. You can see this taking place by creating the following measurements.

### Create S11 Input and S22 Output Match

1. Right-click in the window and add new traces to the same channel. Add **S11** and **S22** measurement traces.
2. While the source is sweeping, watch the source port indicator on the front of the VNA. First, the port 1 indicator will light for two sweeps, then the port 2 indicator will light for 1 sweep while all 3 traces update.

3. Press **Cal > Main > Correction Off**. Notice that the relevant traces will update as the sweep is occurring.

The following image shows the corrected Conversion Loss (VC21), Input Match (S11), Output Match (S22) and the uncorrected Conversion Loss (VC21), which is a memory trace.
Embedded LO Measurements

The Embedded LO feature allows you to make VMC, SMC, IMDx, GCX, MODX and NFX measurements of mixers that have a FIXED LO inside the DUT. Learn how to make IMX Spectrum measurements on converters with an Embedded LO.

Note: This feature is available as Opt S9x084A/B, and must be enabled.

In this topic:

- Overview - How the VNA measures the embedded LO
- To measure a DUT with an Embedded LO (Procedure)
- How to Launch the Embedded LO Mode dialog box
- Embedded LO dialog box help
- Embedded LO Diagnostic dialog box help

Overview - How the VNA measures the embedded LO

Measurements of these devices are challenging for a couple of reasons:

1. The VMC measurement process requires the use of a reference mixer that has the same LO frequency as the DUT. A separate internal or external source must be used for the reference mixer LO. A VNA with an internal second source is much faster. This source (Internal or External) MUST be controlled by the VNA! Learn how.

2. All Embedded LO measurements require the VNA receivers to be tuned to the correct frequency to measure the mixer output, which is highly dependent on the exact LO frequency.

The nominal frequency of the embedded LO is input into the Mixer Setup dialog. This is used as a starting point for the measurement.

Before each DUT measurement sweep, background sweeps are made to determine the frequency of the embedded LO to a configurable degree of accuracy.

Background sweeps...

- **Broadband** Sweep - rough measurement of the embedded LO frequency, made around a selectable data point over a selectable frequency span. The input signal to the DUT is tuned to a selectable CW frequency. The B receiver is swept across a selectable span around the anticipated output frequency. The difference
between the frequency of the found signal and the desired output frequency is then applied as an adjustment.

- **Precise Sweep** The B receiver is measured at the selectable data point. Measurements of phase versus time are made, from which the exact offset frequency is computed, until either the tolerance value or maximum iterations are met.

- For VMC measurements, the reference mixer frequency is updated as the embedded LO frequency is determined.

**To measure a DUT with an Embedded LO:**

1. Create a VMC, SMC, IMDx, IMx Spectrum, GCX, or NFX measurement.
2. In the mixer setup dialog, enter the nominal frequency of the embedded LO as the LO frequency.
3. Perform a calibration as usual.
4. Launch and complete the Embedded LO Mode dialog box (below)

**For VMC with Embedded LO:**

(The other mixer applications do NOT require a reference mixer.)

The LO source for the Reference Mixer can be either:

- An **Internal** source when using a PNA-X that has two sources.
- An **External** source:
  - Must be configured as an External Device. Learn how.
  - Must be locked to the VNA using the 10 MHz reference.

This source (Internal or External) **MUST be controlled by the VNA**. To control the source, set the LO1 source in the Mixer Setup tab to whichever source (external, port 3, or port 4) that is driving the reference mixer.

**During Calibration** - The LO source is shared between the Reference Mixer and the Calibration Mixer/Filter. This requires a splitter when using an external source, as shown in the following image.
**During the Measurement** - Only the Reference Mixer uses the LO source. Terminate the LO source port that is no longer used by the Calibration Mixer/Filter to ensure that the match seen by the Reference Mixer LO port does not change after the calibration, as shown in the following image. This precaution is not necessary when using the internal second source (ports 3 and 4) of the PNA-X.
For SMC, IMDx, IMxSpectrum, and NFX measurements:

No unique setup is required for embedded LO measurements.

**How to Launch the Embedded LO Mode dialog box**

**Using Hardkey/SoftTab/Softkey**

1. Press **Sweep > Source Control > Embedded LO...**

**Using a mouse**

1. Click **Stimulus**
2. Select **Sweep**
3. Select **Source Control**
4. Select **Embedded LO...**
The Tuning Settings balance LO measurement speed versus accuracy. You can see that accuracy is becoming compromised when noise starts to appear on the measurement trace.

Scroll up to learn more about the Embedded LO measurement process.

Enable Embedded LO  Check to enable measurement of the Embedded LO.

Tuning Method  These settings determine the amount of time spent versus the degree of accuracy to which the LO Frequency is measured. You can see that accuracy is becoming compromised when noise starts to appear on the measurement trace.

Broadband and Precise  Does the entire tuning process for each background sweep. See the Overview for more information.

Precise only  Does NOT perform broadband tuning on each sweep. Use this setting when the embedded LO is stable. The signal (after broadband) must be within ½ the tuning IFBW. If the signal will always be within ½ the IFBW, broadband tuning is not needed. Most satellite components are within 3 kHz absolute so might not need broadband tuning.

Disable tuning  Only the previously measured LO Frequency Delta is applied to the reference mixer LO and VNA receivers.

Tuning Point  Select, or specify, the data point in the mixer sweep that will be used to find the
embedded LO frequency. If a marker is enabled, that data point can be used. For broadband and Precise sweeps, choose a point in the mixer sweep where noise is least likely to be found, such as the point of highest gain. This is generally the center of a sweep or the center of a filter if used.

**Tune every** Set the interval at which tuning is performed before a measurement sweep. 'Tune every 3 sweeps' means that every third measurement sweep is preceded by tuning sweeps. If the embedded LO drifts, or if regularly changing DUTs, use 'Tune every 1 sweep'.

**Broadband Search** - Set the frequency span over which to measure the embedded LO frequency.

**IFBW** IF Bandwidth used for Broadband and Precise tuning sweeps. This sets the resolution in the Broadband sweeps and sets the max error (1/2 IFBW) for precise tuning. The larger the IFBW, the faster the sweep, but the signal may not be found.

**Max Iterations** The maximum number of Precise sweeps to make. When this number is reached, the final measurement is used.

**Tolerance** When two consecutive Precise measurements are made within this value, the final measurement is used. If this is not achieved within the Max Iterations value, then the last measurement is used. This is the best of the 'Tunings settings' to change to improve accuracy.

**LO Frequency Delta** The absolute difference between the measured embedded LO frequency and the LO setting that is entered in the Mixer Setup dialog. This value is updated each time the embedded LO frequency is measured. Entering a value is a way to change the LO frequency on the mixer setup without invalidating the calibration.

**Find Now** The VNA finds and measures the actual LO frequency using the current dialog settings. This data is displayed in the Status box.

**Status window** Displays textual and graphical representation of the Embedded LO measurement sweeps.

**Default** Resets the LO Frequency Delta and Tuning parameters to their default settings.

**Graph...** Launches the graphical (spectrum analyzer type) display sweeps of the latest embedded LO measurement.
This dialog appears when **Graph** is clicked on the **Embedded LO** dialog.

Presents a graphical (spectrum analyzer type) display of the latest embedded LO measurement.

Click **Previous** and **Next** to view available Broadband and Precise sweeps. The LO Frequency is displayed in the Marker annotation.
Frequency Offset Mode

Frequency Offset Mode (FOM) provides the capability to have the VNA Sources tune to frequencies that are different (offset) from the VNA Receivers.

VNA Option 080 or S9x080A provides you with the hardware and basic software capability to make Frequency Offset Measurements. This topic discusses the VNA settings that are relevant to making these types of measurements. See Frequency Converting Device Measurements for more information on making specific device measurements.

- Frequency Offset Dialog Box
- Setup Examples
- RF Path Config (Reference Switch) Dialog Box

Other Frequency Offset topics

How to make Frequency Offset settings

Using Hardkey/SoftTab/Softkey

1. Press **Sweep > Source Control > Frequency Offset...**
The following are major changes to FOM:

- Stimulus and Response are now called Sources and Receivers.
- Sources and Receivers settings can be made in two ways:
  1. **By Coupling** to the Primary (Channel) settings. This is the only method used in previous releases.
  2. **By Uncoupling** and setting Sources and Receivers values independently. This is the new, simplified method.
- External sources appear here and can be controlled from this dialog. [Learn more]

**Note:** Source2 supplies power for ports 3 and 4. **Turn Source2 power ON** using the Power and Attenuators dialog. This (Frequency Offset) is the only dialog for controlling the frequency of Source 2. [Learn more about Source2]

**Note:** Source3 is an RF signal to the rear-panel connector. **Turn Source3 power ON** using the Power and Attenuators dialog. This (Frequency Offset) is the only dialog for controlling the frequency of Source 3. [Learn more about Source3]

**Frequency Offset (ON/OFF)** Enables Frequency Offset Mode on ALL measurements that are present in the active channel.

When FOM is NOT enabled, all frequencies are the same as the active channel.

**Tip:** First make other settings on this dialog box, then click **Frequency Offset ON**.

**Primary** The current Active Channel settings. When a Source or Receiver is coupled to the Primary settings, its Sweep Type is the same as that of the Primary. The frequency settings of the coupled range are mathematically derived from the Primary settings using the Multiplier, Divisor, and Offset values. With this approach, only the Primary settings need to be changed in order to affect change in the coupled Sources and Receivers. Changes to the Primary channel settings occur when Frequency Offset is checked ON. [See example using Primary and Coupled setting]

**Tip:** Primary settings are ONLY used when Sources and Receivers are Coupled. It is often easier to Uncouple, then set Sources and Receivers independently.

Source and Source2 if available. [Learn more about Internal Second Source]

**Receivers** All receivers that are used in the channel, including Reference receivers, are tuned to
the specified frequency settings.

**Mode**

**Coupled**  Source and Receiver settings are mathematically derived from the Primary settings using Multiplier, Divisor, and Offset values.  [Learn more.]

**Uncoupled**  Source and Receiver settings are entered independently, without reference to Primary settings. When Uncoupled, Source and Receiver Ranges can use separate sweep types.

**Sweep Type**  Click to change the type of sweep for each range. Only available for Primary and Uncoupled Sources and Receivers.

**Unsupported Sweep Type combinations**

- Power Sweep and Segment Sweep can NOT be used together.
- Uncoupled Log Sweep yields invalid data whenever the sources are offset from the receivers.
- Coupled Log Sweep is allowed only for the following two conditions:
  1. The offset = 0, the multiplier = 1, and the divisor = 1.
  2. The multiplier = 0

**Settings**  To change settings, click **IN** the appropriate Settings cell, then click **Edit**.

- If coupled, invokes the **Coupled dialog**.
- If uncoupled or Primary invokes the **Uncoupled settings dialog**.

**X-Axis Annotation**  Select the settings to be displayed on the X-Axis:

**X-Axis Point Spacing**  Only available when a Segment Sweep Type is selected as the X-Axis display.  [Learn more.]
**Note:** When Frequency Offset is enabled, ALL receivers on the channel, including the reference receivers, tune to the new offset frequencies. Therefore the source and reference receiver will be at different frequencies. Therefore, FOM measurements that include a reference receiver, which includes all S-parameters, display invalid data.

To measure and display measurements at both the source and receiver frequencies, you must use two channels. Use **Equation Editor** to calculate the conversion loss. See a calibrated FOM conversion loss example.

Learn how to calibrate frequency offset measurements.

---

**Coupled settings** dialog box help

**Coupled Formulas:**

\[
\text{Range Start} = \left[ \text{Primary Start} \times \left( \frac{\text{Multiplier}}{\text{Divisor}} \right) \right] + \text{Offset}
\]

\[
\text{Range Stop} = \left[ \text{Primary Stop} \times \left( \frac{\text{Multiplier}}{\text{Divisor}} \right) \right] + \text{Offset}
\]

Where:

**Offset**  Specifies an absolute offset frequency in Hz. For mixer measurements, this would be the LO frequency. Range is +/- 100 GHz. Offsets can be positive or negative.

**Multiplier**  Specifies (along with the divisor) the value to multiply by the stimulus. Range is +/- 100.

- Negative multipliers cause the stimulus to sweep in decreasing direction. For downconverter mixer measurements, this would be for setups requiring the Input frequency to be less than LO frequency. See an example.
- 0 (zero) as the multiplier nulls the Primary setting. Then the Offset value adds to zero.

**Divisor** Specifies (along with the multiplier) the value to multiply the stimulus. Range is 1 to 100.

---

**Primary and Uncoupled settings** dialog box help

![Primary and Uncoupled settings dialog](image)

This dialog will vary depending on the sweep type:

**Linear and Log frequency**

Uncoupled Log sweep yields **invalid data** whenever the sources are offset from the receivers.

- **Select Start/Stop or Center/Span**
- **Frequency** Enter values
- **Points** (Primary only) Enter number of data points for the sweep.

**Power**

- **CW Freq** Enter frequency in Hz.
- **Points** (Primary only) Enter number of data points for the power sweep.

**CW Time**

- **CW Freq** Enter frequency in Hz.
- **Sweep Time** Enter time to complete one sweep. Enter 0 for the fastest sweep.

**Segment Sweep** Edits are made exactly like the **standard segment table**.
**For Advanced Users:** Uncoupled Segment Sweep offers great flexibility in configuring measurements. In segment sweep mode:

- The **OK** button is NOT available until the total number of data points for all segments matches the number of Primary data points.

- **Independent IF Bandwidth** and **Independent Sweep Time** are available ONLY on the Primary (channel) and the Uncoupled **Receivers** - NOT Sources.

- **Independent Power** is available ONLY on the Primary (channel) and the Uncoupled **Sources** - NOT Receivers.

**Setup Examples**

Although the Frequency Offset settings can be used with many types of devices, these examples include mixer terminology.

See a Mixer Compression and Phase (AM-PM) Measurement using FOM.

See a calibrated FOM conversion loss example.

1. **Fixed LO - Upconverter**

   - **Swept Stimulus (Mixer Input):** 1000 MHz - 1200 MHz
   - **Fixed LO:** 1500
   - **Swept Response (Mixer Output):** 2500 MHz to 2700 MHz

   Make the following settings on the FOM dialog

   **Source:** Uncoupled

   - Sweep Type: Linear

   - Click Settings, then Edit. In the Source dialog:
     - Start Frequency = 1000 MHz
     - Stop Frequency = 1200 MHz

   **Receiver:** Uncoupled

   - Sweep Type: Linear
Click Settings, then Edit. In the Receiver dialog:

Start Frequency = 2500 MHz

Stop Frequency = 2700 MHz

**LO Settings**

Set external source to CW - 1500 MHz.

**Source2**: Uncoupled (Only with Second VNA Internal Source)

Sweep Type: CW Time

Click Settings, then Edit. In the Source2 dialog:

CW Frequency = 1500 MHz

2. **Fixed LO - Downconverter (Input < LO)**

- **Swept INCREASING Stimulus (Mixer Input)**: 1000 MHz to 1100 MHz
- **Fixed LO**: 2500 MHz
- **Swept DECREASING Response (Mixer Output)**: 1500 MHz to 1400 MHz

Make the following settings on the FOM dialog

**Primary**: Not used

**Source** (Input): Uncoupled

Sweep Type: Linear

Click Settings, then Edit. In the Source dialog:

Start Frequency = 1000 MHz

Stop Frequency = 1100 MHz

**Receiver** (Output): Coupled

Sweep Type: Linear

Click Settings, then Edit. In the Receiver dialog:

Offset: 2500 MHz
Multiplier: -1 (Minus one)

LO Settings

- If using external source, set to CW: 2500 MHz.
- If using Source2: Set to Uncoupled, then:
  
  Sweep Type: CW Time
  
  Click Settings, then Edit. In the Source2 dialog:
  
  CW Frequency = 2500 MHz

See a calibrated FOM conversion loss example.

3. **Swept LO - Fixed Output - Upconverter**

Swept External LO measurements in Frequency Offset Mode can be very difficult. The external LO source must be synchronized with the swept output or input (as in this case). See Synchronizing and External Source Control to see how this is done. The Frequency Converter Application Opt S9x083A/B performs makes these measurements easily.

- **Swept Stimulus (Mixer Input):** 1000 MHz to 1100 MHz
- **Swept LO:** 1500 MHz to 1400 MHz
- **Fixed Response (Mixer Output):** 2500 MHz

Make the following settings on the FOM dialog

**Source:** Uncoupled

Sweep Type: Linear

Click Settings, then Edit. In the Source dialog:

Start Frequency = 1000 MHz

Stop Frequency = 1100 MHz

**Receiver:** Uncoupled

Sweep Type: CW Time

Click Settings, then Edit. In the Receiver dialog:
CW Frequency = 2500 MHz

LO Settings

- If using external source, set to sweep from 1500 - 1400 MHz.
- If using **Source2 (Second Internal Source)**: set to Uncoupled, then:

  Sweep Type: Linear
  
  Click Settings, then Edit. In the Source2 dialog:

  Start Frequency = 1500 MHz
  
  Stop Frequency = 1400 MHz

4. **Power Sweep for Mixers**

   To measure the gain compression of a mixer, the input power to the mixer is swept. The input and output frequencies are fixed but offset from one another.

   This is a good use of Coupled settings because the same compression test can be performed at several different frequencies. With coupled Source and Receiver ranges, the Primary (channel) frequency can be easily changed from the front panel. The coupled source and receiver frequencies will update accordingly.

   - **Swept Input Power**: -10 dBm to 0 dBm
   - **Fixed Input Frequency**: 1500 MHz
   - **Fixed LO**: 500 MHz
   - **Fixed Output**: 2000 MHz

   Make the following settings on the FOM dialog

   **Primary**:

   Sweep Type: Power Sweep

   Click Settings, then Edit. In the Primary dialog:

   CW Frequency = 1500 MHz

   **Source**: Coupled
Default settings make CW Frequency: 1500 MHz (same as Primary)

**Receiver:** Coupled

Default settings make Sweep Type: CW Time

Click Settings, then Edit. In the Receiver dialog:

**Offset = 500 MHz**

**LO Settings**

- If using external source, set to CW: 500 MHz.
- If using **Source2 (Internal Second Source)**, set to Coupled, then:

  Sweep Type: Power Sweep

  Click Settings, then Edit. In the Source2 dialog:

  **CW Frequency = 500 MHz**

---

**Test Set Reference Switch**

PNA-X and N522x models have a switch in the test set that allows you to bypass the port 1 reference receiver through the front panel Reference 1 connectors. This switch lets you easily switch between standard S-Parameter measurements and measurements using a reference mixer. You could use this feature to make standard S11 measurements and converter transmission measurements relative to a reference ("golden") mixer.

**Note:** The Frequency Converter Application Option S93083A/B simplifies the task of making extremely accurate phase measurements on MOST frequency converting devices.

---

**How to access the RF Path Config dialog box**

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Setup &gt; Internal Hardware &gt; RF Path Config...</strong></td>
<td>1. Click <strong>Instrument</strong></td>
</tr>
<tr>
<td>2.</td>
<td>2. Select <strong>Setup</strong></td>
</tr>
<tr>
<td>3.</td>
<td>3. Select <strong>Internal Hardware</strong></td>
</tr>
<tr>
<td>4.</td>
<td>4. Select <strong>RF Path Config</strong></td>
</tr>
</tbody>
</table>

---

**Programming Commands**
**RF Path Config dialog box help**

![RF Path Config](image)

**Note:** This feature is available on PNA-X and N522x models.

**R1 Input Path**

- **Internal**  Connects the port 1 source directly to the R1 receiver.
- **External**  Allows direct access to the R1 receiver through the Reference 1 front-panel connectors.

See specifications which include a block diagram of reference switch.
Frequency Converting Device Measurements

Many frequency offset measurements can be made using the VNA with S93080A. The following is a list of some of those measurements and how they are made.

- Conversion Loss
- Conversion Compression
- Return Loss and VSWR
- Isolation
- Harmonic Distortion

See Also: Frequency Offset Measurement Accuracy
Frequency Offset Measurement Accuracy

This topic discusses methods that can be used to make accurate frequency offset measurements.

- Calibrations
- Mismatch Errors
- Accurate and Stable LO

See other Mixer Measurement topics

Calibrations

With Frequency Offset measurements, the stimulus and response frequencies are different. Standard calibration error terms are calculated using reference measurements. Therefore, traditional calibration methods such as full 2-port SOLT cannot be used with frequency offset.

Source and Receiver Power calibrations can be used to calibrate your Frequency Offset measurements.

Frequency Converter Application offers fully calibrated scalar and vector frequency offset measurements.

Source Power calibration:

- Sets accurate power level at stimulus frequencies regardless of the receiver that will be used in the measurement.
- Can be copied to other channels with copy channels feature.
- Can be interpolated.

Receiver Power Cal:

- Requires a source cal to have already been performed and applied.
- Cannot be copied to other channels.

Therefore:

- Start by performing a source power cal over the combined stimulus and response frequencies.
• Copy the channel to other needed channels and the source power cal is copied.

• Change the frequency range of the copied channel to response frequencies.

• Perform a receiver cal at the response frequencies on individual channels.

• Change the frequency range to stimulus frequency and switch frequency offset ON.

• On Status Bar, ensure that source and receiver cals are ON (source cal will be interpolated).

See Frequency Offset Conversion Loss Measurements to see a step-by-step example.

Mismatch Errors

Mismatch errors result when there is a connection between two ports that have different impedances. With S-parameter measurements, these mismatches are measured and mathematically removed during a full 2-port calibration. This is much more difficult with frequency offset measurements. A much easier solution is to use high-quality attenuators on the input and output of the mixer.

By adding a high-quality attenuator to a port, the effective port match can be improved by up to twice the value of the attenuation. For example, a 10-dB attenuator, with a port match of 32 dB, can transform an original port match of 10 dB into an effective match of 25 dB. However, as the match of the attenuator approaches the match of the original source, the improvement diminishes.

Note: The Frequency Converter Application (option S93083A/B) uses calibration techniques that correct for mismatch errors.

The larger the attenuation, the more nearly the resulting match approaches that of the attenuator, as shown in the following graphic. However, excessive attenuation is not desired because that will decrease the dynamic range of the measurement system.
Accurate and Stable LO

When using frequency offset mode, if the LO signal is not accurate and stable, the output signal will not be at the expected response frequency. As a result, the output signal can fall on the skirts of the VNA receiver IF filter, or fall completely outside of the receiver filter passband.

Also, the LO power level is critical in mixer measurements. Be sure to monitor these power levels closely.
Conversion Loss (or Gain)

- What is Conversion Loss?
- Why Measure Conversion Loss?
- How to Measure Conversion Loss

See other Frequency Converting Device Measurements

What is Conversion Loss?

Conversion loss is defined as the ratio of the power at the output frequency to the power at the input frequency with a given LO (local oscillator) power. This is illustrated in the graphic below. A specified LO power is necessary because conversion loss varies with the level of the LO, as the impedance of the mixer diode changes.

![Conversion Loss Diagram](image)

Why Measure Conversion Loss?

Conversion loss (or gain in the case of many converters and tuners) is a measure of how efficiently a mixer converts energy from the input frequency to the output frequency. If the conversion loss response of a mixer or converter is not flat over the frequency span of intended operation, valuable information may be lost from the resulting output signal.
How to Measure Conversion Loss

Conversion loss is a transmission measurement. It is measured by applying an input signal (stimulus) and an LO signal at specific known power levels, and measuring the resulting output signal level. Because the output frequency is different from the input frequency, frequency offset mode (option S93080A) must be used for this measurement.

Note: This measurement is made much easier if your VNA has the Frequency Converter Application

Equipment Setup

Example: A calibrated Conversion Loss (Down-converter) measurement

Swept Input with Fixed LO = Swept Output

- RF Input: 3.1 - 3.3 GHz
- LO: 2.2 GHz
- IF Output: 900 - 1100 MHz

VNA setup and calibrate on channel 1

1. On channel 1 create an unratioed R measurement over the ENTIRE input and output frequency span (.9 - 3.3 GHz). This will be the base source power cal that will be copied to the R and B channel measurements.

2. Perform a source calibration using a power meter. This makes the power level at the input of the mixer very accurate.

Setup Reference measurement on channel 2
1. **Copy channel 1** to channel 2 which will display the reference input to the mixer. The channel 1 source power cal is copied with the other channel settings.

2. Change measurement to R1 unratioed.

3. Change RF Input frequency to 3.1 - 3.3 GHz. The source power cal becomes interpolated.

4. Perform receiver power cal. Do not need to make physical connections. The VNA source is internally connected to the R1 receiver. Makes the R receiver read the source power level.

### Setup B measurement on channel 3

1. Copy channel 1 to channel 3. This channel will display the output of the mixer. The channel 1 source power cal is copied with the other channel settings.

2. Change measurement to B unratioed.

3. Change IF Output frequency to .9 - 1.1 GHz. This causes the source power cal becomes interpolated.

4. Connect thru line from port 1 to port 2.

5. Perform receiver power cal. This makes the B receiver read the source power at the IF Output frequencies.

6. **Turn OFF receiver power cal.** This prevents an error when changing to input frequencies (next step).

7. Change RF Input frequency to 3.1 - 3.3 GHz. This changes the channel back to the mixer RF Input frequencies.

8. **Enable Frequency Offset.**

9. Change Offset to (-2.2 GHz). This tunes the B receiver to the IF Output frequencies .9 to 1.1 GHz. **Note:** The minus sign indicates a down-converter measurement.

10. Turn ON receiver power cal.

### Measure the Mixer

1. Connect the mixer.

2. Adjust **scale** to suit your needs.

3. Enable **markers** to read power levels for each trace.

The display below shows:

- Ch3 B receiver (bottom trace) absolute output power.
Ch2 R1 receiver measurement (top trace) absolute input power to the mixer.

With this method, the conversion loss math (B/R1) can be performed with Equation Editor (not shown). The B/R1 ratio measurement is not supported with receiver power Cal turned on. However, conversion loss (C21) measurements can be made directly and are much easier using the Frequency Converter Application, FCA (Opt S93083A/B).
What is Conversion Compression?

Conversion compression is a measure of the maximum input signal level for which a mixer will produce linear operation. It is very similar to the gain compression experienced in amplifiers.

To understand conversion compression, you must first understand conversion loss. This is the ratio of the mixer output level to the mixer input level. This value remains constant over a specified input power range. When the input power level exceeds a certain maximum level, the constant ratio between input and output power levels begins to change. The point at which the ratio has decreased 1 dB is called the 1-dB compression point. This is illustrated in the graphic below.

Why Measure Conversion Compression?

...
Conversion compression is an indicator of the dynamic range of a device. Dynamic range is generally defined as the difference between the noise floor and the 1-dB compression point.

**How to Measure Conversion Compression**

The equipment and setup used to measure conversion compression are essentially the same as for measuring conversion loss and is illustrated in the following graphic.

The VNA performs a power sweep using **frequency-offset mode** and the resulting display shows the mixer's output power as a function of its input power. The 1-dB compression point (or others such as 3-dB) can be determined using markers.

![Diagram of VNA setup](image)

**Measurement Accuracy Considerations**

**Equipment Setup Considerations**

- The couplers in the VNA have very good directivity. If the return loss of the DUT is bad, the reflected signal gets sampled by the VNA and can result in errors. This relates to error in DUT gain. To increase the accuracy, an attenuator can be added between the VNA's source port and the DUT's input port. Normally a 6- to 10-dB attenuator is sufficient. Addition of this attenuator, however, decreases the available drive to the DUT.

- With high drive levels the VNA can be driven into compression resulting in measurement error. With excessive drive levels, the VNA can be damaged. Add an attenuator between the output of the DUT and the receiver input of the VNA to avoid these problems.

**Calibration Considerations**
Source power calibration can be used to provide a high level of accuracy for this measurement.
Isolation Measurements of Frequency Converting Devices

- What is Isolation?
- Why Measure Isolation?
- How to Measure Isolation

See other Frequency Converting Device Measurements

What is Isolation?

Isolation is a measure of the leakage, or feedthrough, from one port to another. The more isolation a mixer provides, the lower the amount of feedthrough. Isolation is measured at the same frequency as the stimulus, not the converted or shifted frequency. Therefore, Frequency Offset capability is not necessary for these measurements.

Three main isolation terms are of interest for mixer measurements:

- LO-to-OUT isolation ($V_{LO}$)
- LO-to-IN isolation ($V_{LO}$)
- IN-to-OUT feedthrough ($V_{IN}$)

Why Measure Isolation?

Any unwanted signal "leaking" through the device will mix with the desired output signal creating intermodulation products, adding to intermodulation distortion. These unwanted signals may be...
difficult to filter out.

**How to Measure Isolation**

Use the following setups to measure the isolation of a mixer:

Note the following:

- The Input to Output isolation is very dependent on the LO power level. Isolation should be measured with the LO power at its normal operating level.

- Each of the ports not being tested should be terminated with an impedance typical of actual operation. This may not always be the characteristic impedance, Z0 (usually 50 or 75 ohms). For example, if the OUT port of a mixer is intended to be directly connected to a filter, then this filter should be used when measuring the LO-to-IN feedthrough.

![LO-TO-IN ISOLATION](image)

**LO-TO-OUT ISOLATION**
Measuring Converters vs. Mixers

Measuring IN-to-OUT feedthrough of a converter is identical to that of a mixer. The IN-to-OUT feedthrough is generally very small for a converter due to the inclusion of an IF filter in the device. Because of this, the measurement may require the VNA to have increased dynamic range.

Measuring LO leakage (LO-to-OUT and LO-to-IN) of a converter requires a different technique because the LO port is typically not accessible:

- The VNA can be tuned to the frequency of the LO signal and either the OUT or IN port connected to the VNA receiver port. The VNA source port is not connected.

- A spectrum analyzer can be connected to either the OUT or IN port and tuned to the frequency of the LO
signal.
Harmonic Distortion

- What is Harmonic Distortion?
- Why Measure Harmonic Distortion?
- How to Measure Harmonic Distortion
- Measurement and Accuracy Considerations

See other Frequency Converting Device Measurements

What is Harmonic Distortion?

Harmonics are multiples of any signal appearing at the mixer input and also multiples of the LO input. The distortion of the mixer's output characteristics caused by these harmonics is referred to as harmonic distortion. Harmonic distortion is caused by non-linearities in the device.

Harmonics are NOT signals created by two or more signals interacting (mixing); these signals are known as intermodulation products, which result in intermodulation distortion.

Why Measure Harmonic Distortion?

- It can degrade the performance of devices connected to the output of the mixer.
- The harmonics can also mix with other signals present in the mixer, adding to the intermodulation distortion of the mixer.

How to measure Harmonic Distortion

The harmonics can be measured using the VNA with Frequency Offset (option 80). The frequency of the LO to the mixer is set to zero and multiplier of the RF input is used to set the IF frequency (the harmonic). The equipment setup is shown below.

Since harmonics are specified in dBc, the fundamental RF and both the second and third harmonics are measured and the differences calculated. Multiple channels can be used to do this.

1. Connect the equipment.
2. Setup the measurement for calibration. See also Measurement and Accuracy Considerations.
Use three channels and frequency offset mode:

Channel 1 = F1 to F2

Channel 2 = F1 to 2F2 (frequency offset mode, multiplier = 1)

Channel 3 = F1 to 3F2 (frequency offset mode, multiplier = 1)

- Perform a source power calibration and receiver power calibration over the entire frequency range. See Measurement and Accuracy Considerations.

- Reduce the frequency span and increase the frequency offset multiplier on Channels 2 and 3:
  Channel 2 = F1 to F2 (frequency offset mode, multiplier = 2)
  Channel 3 = F1 to F2 (frequency offset mode, multiplier = 3)

  **Note:** Because the frequency span has been changed from that used for calibration, the source and receiver calibrations will be interpolated.

- Connect the DUT, make the measurement, and calculate the harmonic response:
  Set up markers on Channels 1, 2 and 3, and determine the difference between the marker values to get the dBc value of each harmonic.

  Channel 1 - Channel 2 = 2nd harmonic (dBc)

  Channel 1 - Channel 3 = 3rd harmonic (dBc)

  **Note:** Be sure to set the markers to the appropriate stimulus. Channel 2 markers should be set to twice the frequency of Channel 1 markers. Channel 3 markers should be set to three times the frequency of Channel 1 markers.

---

**Measurement and Accuracy Considerations**

**Equipment Setup Considerations**

- A filter must be used at the input of the mixer to remove the VNA source harmonics.
Return Loss and VSWR

- What are Return Loss and VSWR?
- Why Measure Return Loss and VSWR?
- How to Measure Return Loss and VSWR

What is Return Loss and VSWR?

Return loss and VSWR are both linear reflection measurements, even when testing frequency conversion devices, because the reflected frequency is not converted. These measurements are essentially the same as for filters and amplifiers. Learn more about Reflection Measurements.

Why Measure Return Loss and VSWR?

Devices which have poor return loss and VSWR result in loss of signal power or degradation of signal information.

How to Measure Return Loss and VSWR

Setup the VNA measure return loss and VSWR as you would any two-port device. Connect your frequency converting device as shown in the following diagrams:

RETURN LOSS AND VSWR OF MIXER INPUT PORT
RETURN LOSS AND VSWR OF MIXER OUTPUT PORT

RETURN LOSS AND VSWR OF MIXER LO PORT
Gain Compression for Amplifiers GCA (Opt S9x086A 086)

- Features, Requirements, and Limitations
- Gain Compression Concepts
- Understanding the GCA Displayed Traces
- Gain Compression Parameters
- Compression Methods
- Acquisition Modes
- Using Gain Compression App
  - Frequency tab
  - Power tab
  - Compression tab
  - Safe Sweep Mode dialog
- Compression Analysis
- Saving GCA Data
- GCA Measurement Tips
- Macros

See Also

- Gain Compression for Converters
- GCA Calibration
- Programming commands
- **App Note** Amplifier Linear and Gain Compression Measurements

Other VNA Applications
Features, Requirements, and Limitations

**Features**

- Fast, easy, and complete Gain Compression measurements for amplifiers.
- Many compression parameters to choose from, including gain, input power at compression, output power at compression, input match, and compression level.
- Several compression methods to choose from, including deviation from linear gain, deviation from max gain, back-off, and X/Y, and compression from saturation.
- Three acquisition methods to choose from: Power per Freq, Freq per Power, and SMART Sweep
- SMARTCal Calibration Wizard to guide you through Full 2-Port or Enhanced Response calibration, plus Source Power calibration.
- Compression Analysis allows traditional power sweep view at a selected frequency.
- Receiver Leveling provides continuous source power accuracy.
- Supports Wideband (NOT Narrowband) Pulse measurements using the new integrated Pulse setup dialogs.

**Requirements**

- Option S9x086A (software option only) must be enabled.
- When performing an optional calibration:
  - ECal module or Calibration Kit
  - Power meter/sensor

**Limitations with GCA**

- Number of points limited to 100,001 for two-dimensional acquisitions, 50,000 points for SMART Sweep.
- Standard CW power sweep is NOT supported in a Gain Compression channel.
- Independent IFBW, Power Levels, Shift LO, or Sweep Time in a segment table is NOT supported.
- Stepped sweep mode only.
- Linear, Log, and Segment frequency sweep modes only.

The following VNA Features are **NOT** Available in a Gain Compression channel:
• Unratioed receiver measurements (A, B, R)
• ECAl User Characterization
• Some Fixturing Features
• FOM or FCA
• External Test Set Control (Option S93551A/B)
• IF Path Configuration
• CW Time sweep
• Balanced measurements
• Save Auto Formatted Citifile data.
• Narrowband Pulse measurements using the Integrated Pulse App

Gain Compression Application Concepts

What is Gain Compression

An amplifier has a region of linear gain, where the gain is independent of the input power level. This gain is commonly referred to as small signal gain. As the input power is increased to a level that causes the amplifier to approach saturation, the gain will decrease. The 1 dB gain compression is defined as the input power level that causes amplifier gain to drop 1 dB relative to the linear gain.

You can quickly measure the gain compression using a compression marker on a power sweep trace.

Terms used in GCA

Linear Power Level  The specified input power that yields linear gain (also known as 'small-signal gain') in the amplifier.

Reference gain  The measured gain that is used as a reference for determining compression level. The
Compression Method that is used could cause this value to be different.

Compression level The specified amount of gain reduction from the reference gain.

Target gain The gain at the specified compression level. Although this term does not appear in GCA, it is important to understand when discussing the various compression parameters.

For example, when using Compression from Linear Gain method with the following settings:

- Linear gain (measured at Linear Input power) = 10.2 dB
- Compression level (specified) = 1 dB
- Target gain = 9.2 dB

This is called 'Target' gain because GCA will search for the closest measured gain to 9.2000 dB. It may not measure this gain exactly.

Compression point The operating point at which the measured gain is closest to the Target Gain. All compression parameters report data for this operating point.

Understanding the GCA Displayed Traces

One of the most important concepts to remember with GCA is that, each frequency data point represents many measurements using different input power levels.

Some things to notice about how GCA displays compression data:
1. The X-axis values are ALWAYS frequency. Imagine behind each frequency data point, a traditional power sweep curve with corresponding measurements and calculations to find the specified compression point.

2. The Y-axis values are always reported at the compression point. The value that is displayed depends on the compression parameter that you choose. The S-parameters that are displayed in a GCA channel are always measured at the linear and reverse power level.

Example: Five of the six GCA compression parameters are displayed in the above image. The missing trace, DeltaGain21 is discussed below.

- Markers are placed at 4.549 GHz for all of the parameters.
- Tr 3 CompIn21 (Input power at the compression point) shows the marker value to be \(-5.4117\) dBm. This is the power into the DUT that was required to achieve the compression point. Notice that this is about the same input power required to achieve the specified compression at ALL frequencies.
- Tr 5 CompGain21 (Gain at the compression point) shows the marker value \(9.6443\) dB. This is the measured gain at the compression point.
- To see the gain at a different input power at this frequency, use the Compression Analysis feature.

Create a GCA Measurement

1. Press Meas > S-Param > Meas Class....

2. Select Gain Compression, then either:
   - OK delete the existing measurement, or
   - New Channel to create the measurement in a new channel.

Gain Compression Parameters

There are several Gain Compression parameters, as well as standard S-parameters and ADC parameters, that can be measured in a GCA channel.
How to add GCA Parameters

Using **Hardkey**/**SoftTab**/**Softkey**

1. Press **Trace**, then select trace.
2. Press **Meas**, then select a parameter.

Using a mouse

1. Click **Instrument, Trace, Add Trace**.
2. Click **Response, Meas**, then select a parameter.

### Programming Commands

#### Linear S-Parameters

For convenience, the standard S-parameters are offered in a GCA channel. S11 and S21 are measured at the specified Linear Input level. S22 and S12 are measured at the specified Reverse power level.

**Note:** When a **DC meter** is added, it will be displayed in the New Trace dialog and **SMART Sweep Safe Mode** dialog (in DC Parameters pull down menu).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>When Measured</th>
</tr>
</thead>
<tbody>
<tr>
<td>S11</td>
<td>Input Match</td>
<td>Always</td>
</tr>
<tr>
<td>S21</td>
<td>Gain</td>
<td>Always</td>
</tr>
<tr>
<td>S22</td>
<td>Output Match</td>
<td>See Reverse</td>
</tr>
<tr>
<td>S12</td>
<td>Reverse Isolation</td>
<td>See Reverse</td>
</tr>
<tr>
<td>Al1</td>
<td>Linear Al1</td>
<td></td>
</tr>
<tr>
<td>Al2</td>
<td>Linear Al2</td>
<td></td>
</tr>
</tbody>
</table>

#### ADC Parameters

Four ADC analog-to-digital converter measurements are offered in a GCA channel:

- **Al1** and **Al2** are measured at the specified Linear Input level.
- **CompAl1** and **CompAl2** are measured at the specified compression point.

These DC measurements, along with **Equation Editor**, allow you to make PAE measurements at the Linear Input level and compression point.

These measurements are made at pins 7 and 8 of the VNA **Power IO connector**.

Learn more about ADC measurements.

#### Compression Parameters
The following table assumes: DUT Input = VNA port 1 and DUT Output = VNA port 2.

When the Port mapping is different, the parameters in GCA are updated accordingly. For example, with Input = port 2 and Output = port 1, then "CompIn12" would be displayed.

The raw data for these parameters are always measured.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CompIn21</td>
<td>Input power at the compression point.</td>
</tr>
<tr>
<td>CompOut21</td>
<td>Output power at the compression point.</td>
</tr>
<tr>
<td>CompGain21</td>
<td>Gain at the compression point.</td>
</tr>
<tr>
<td>CompS11</td>
<td>Input Match at the compression point.</td>
</tr>
<tr>
<td>RefS21</td>
<td>Linear Gain value used to calculate the compression level. This is calculated differently depending on the compression method.</td>
</tr>
<tr>
<td>DeltaGain21</td>
<td>CompGain21 MINUS Linear Gain (in Log Mag format). This trace can be used to learn a lot about the DUT compression point. Learn more.</td>
</tr>
</tbody>
</table>

**Compression Methods**

GCA offers the following methods to find the compression point of an amplifier using GCA:

**Compression from Linear Gain**

The Reference Gain is measured using the specified Linear (Input) Power Level. The Target Gain is calculated as the Linear Gain minus the specified Compression Level. For example 8.3 dB - 1 dB = 7.3 dB.

**Compression from Max Gain**

The linear region of an amplifier gain may not be perfectly linear. The highest gain value that is found at each frequency is used as the Reference (S21) Gain. The Target Gain is found in the same way as Compression from Linear Gain.

**Compression from Saturation**

This method is used to better find the compression point when measuring amplifiers with non-linear gain as shown in the following image:
The Max power out value * is found at each frequency. Then input power is lowered until the output power decreases by the specified 'From Max Pout' value. This is the compression point.^

**Backoff and X/Y method**

These two compression methods are very similar.

- Both methods specify a difference in input power (X axis) between the linear region and compression point.
- For the Y-axis difference:
  - **Backoff method** specifies Compression Level which is a difference in **Gain**.
  - **X/Y method** specifies Delta Y which is a difference in **Output Power**.

GCA searches for these points differently for **2D sweeps** and **SMART sweep**.

The following images show how Backoff and X/Y method is calculated at ONE frequency.
The compression point (yellow circle) is where 10 dB more input power yields 1 dB less gain than at the reference point (blue circle).

---

Acquisition Modes

The GCA offers three modes for data acquisition: Two 2D sweep modes, and SMART sweep.

To see a traditional power sweep at a single frequency, use the Compression Analysis feature. Learn more.

2D (two-dimensional) Sweeps

This is the easiest method to understand, and the least efficient for finding the compression point. Both 2D sweep modes work as follows:

1. All GCA measurements begin by measuring S-parameters at the specified Linear Power level. Reverse parameters are measured ONLY if Full 2-port calibration is applied or if a reverse parameter is displayed. Learn more about Cal choices.

2. Gain measurements are then made at ALL of the specified frequency and power values. Although these are conceptually 2-Dimensional sweeps, a single sweep is constructed in firmware. See Data Points Limit.

3. After data has been measured, a search is performed to find the compression point. You can choose to interpolate between the two measured points closest to the target gain. Learn more.
As each sweep is performed, dots are plotted next to the Ch indicator in the lower left corner of the display to indicate progress for the current sweep.

**Note:** For Backoff and X/Y compression method, GCA does not verify that the specified Start - Stop power range is at least the size of the specified Backoff or X value. The closest compression point is always reported.

**Note:** SMU Hardware List trigger mode is NOT supported in GCA 2D sweeps.

### 2D Sweep Modes

- **2D Sweep Power per Frequency** - Input power is stepped from Start to Stop at each specified frequency. From the following example you can see that the device is exposed to the highest power level (p3) at the first frequency (f1). This could heat the device early in the measurement and affect compression results.

The following examples show (frequency, power) values for three frequency points and three power points, resulting in a total of 9 measurements:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>f1,p1</td>
<td>f1,p2</td>
<td>f1,p3</td>
<td>f2,p1</td>
<td>f2,p2</td>
<td>f2,p3</td>
<td>f3,p1</td>
<td>f3,p2</td>
<td>f3,p3</td>
</tr>
</tbody>
</table>

- **2D Sweep Frequency per Power** - Frequency is swept from start to stop at each specified power level as follows:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>f1,p1</td>
<td>f2,p1</td>
<td>f3,p1</td>
<td>f1,p2</td>
<td>f2,p2</td>
<td>f3,p2</td>
<td>f1,p3</td>
<td>f2,p3</td>
<td>f3,p3</td>
</tr>
</tbody>
</table>

### Viewing and Saving 2D Data

It is NOT possible to plot ALL of the 2D measurement data on the VNA display. However, it can be saved to a *.csv file and then read into an Excel spreadsheet. The initial S-parameter measurement data is not saved to this file. Learn more.

You can also view on the VNA all power sweep information at a selected frequency using the Compression Analysis feature.

### SMART Sweep

SMART Sweep is usually the fastest and most accurate method to measure Gain Compression. Unlike the 2D acquisition modes which measure all of the specified frequency / power points, SMART Sweep performs a series of power search iterations. At each frequency, an 'intelligent guess' of input power is made to find the compression level that is within tolerance. This guess is further refined with each
successive power search iteration sweep.

SMART Sweep continues to iterate until one of the following conditions occur:

1. ALL data points are within tolerance. When the compression level for a data point achieves the specified tolerance, it continues to be measured and input power changed to improve the measurement within tolerance.

2. The specified compression level can NOT be achieved for the remaining frequencies that are not in tolerance. Either the Start power is too high or the Stop power is too low.

3. Maximum iterations have been achieved. If a measured gain is not within the specified tolerance before the specified Max number of Iterations has been reached, then the last power reading is used as the compression point.

The Iteration Counter, Dots, and Bangs(!)

Next to the Ch indicator, in the lower left corner of a GCA window, the following annotation appears:

- An iteration counter is incremented each time input power is adjusted.
- A dot appears when another 10% of the frequency points are within tolerance.
- ! (bangs) are displayed after the last iteration. Each bang represents 10% of the data points that are NOT within tolerance.

SMART Sweep and Compression Method

The intelligent guess process works differently depending on the compression method. This is important because Backoff and X/Y compression methods subject the DUT to significant changes in input power during an iteration sweep. This can affect the DUT and the measurement results.

Learn all about Backoff and X/Y compression methods.

ALL GCA measurements begin by measuring S-parameters at the specified Linear Power level. Reverse parameters are measured ONLY if Full 2-port calibration is applied or if a reverse parameter is displayed. Learn more about Cal choices.

- Backoff and XY Because both compression methods specify the separation between the "linear" region and the "compressed" region, each iteration requires a single sweep at two dramatically different power levels over the same frequency range. The first half of the sweep measures the DUT at the Backoff or X power level. The second half of the sweep measures the DUT at the compressed power level, specified by the Start
and Stop power range. At the beginning of the second half, the power level rises by the Backoff or X value. The specified Settling Time is applied at this point to allow the DUT time to react to this significant change in power level. Safe Sweep does NOT minimize this change in input power. However, Safe Sweep with Backoff and XY methods DOES prevent the DUT from being exposed to too much input power.

- **Compression From Linear Gain** After the reference gain is measured at the linear input power, the next iteration measures the DUT at a higher power level which attempts to push the DUT well into compression. Subsequent sweeps, depending upon the compression level of the DUT, either increases or decreases the power in order to reach the desired compression level. Usually, by the third iteration sweep, a curve-fit algorithm is utilized to precisely find the compression point.

**Note:** The DUT can be subject to significant changes in power from one iteration sweep to the next. This can be minimized by the use of SAFE Sweep and careful selection of the corresponding settings.

- **Compression from Max Gain** The maximum gain that is found at each frequency is stored and used to calculate the compression point. SMART Sweep does NOT perform extra iterations to search for the maximum possible gain of the amplifier at each frequency.

- **Compression from Saturation** The maximum power out that is found at each frequency is stored and used to calculate the compression point. SMART Sweep does NOT perform extra iterations to search for the maximum possible power out of the amplifier at each frequency.

**Using the Gain Compression Application**

The following is a general procedure for performing a GCA measurement. The challenge with GCA is configuring a measurement that yields the true compression performance of YOUR DUT. This requires knowledge of the Gain Compression settings and knowledge of the DUT.

See specific dialog boxes below.

1. Disconnect the DUT if preset or default power levels may damage the VNA or DUT.
2. **Preset** the VNA, or configure a suitable **User Preset** that will be safe in case the DUT is connected.
3. Create a GCA channel. Learn how. The default trace is S21.
4. Start **GCA Setup dialog** and configure the measurement settings based on the DUT, adapters, attenuators, booster amplifiers, and fixtures to be used in the measurement.
5. Save the **instrument state** (optional).
6. Connect DUT and apply bias and RF power as appropriate. The default measurement for a GCA channel is S21 (amplifier gain). Inspect the gain measurement to ensure the DUT is operating as expected.
7. Add GCA compression parameter traces. Learn how.

8. Adjust the measurement settings to yield satisfactory compression parameters. See GCA Measurement Tips.

9. Start and complete the GCA Calibration wizard.

How to start the Gain Compression Setup dialog

Using Hardkey/SoftTab/Softkey

1. Press Freq > Main > GCA Setup....

Using a mouse

1. Click Stimulus
2. Select GCA Setup...

Frequency tab - Gain Compression - dialog box help

Configures the frequency settings over which Gain compression is to be measured, as well as the measurement method.

Sweep Type

Choose a method in which to sweep frequency: Linear, Log, and Segment Sweeps. This setting applies to all data acquisition modes.
**Notes**

- Log and Segment Sweep are NOT available on GCX.
- CW Sweep is NOT available in GCA. However, to see a traditional power sweep at a single frequency, use the **Compression Analysis** feature.
- To use CW Sweep in GCX, set all ranges to **Fixed** on the **Mixer Frequency tab**. This has the same effect as setting all ranges to Start = Stop Frequency.

**Segment Sweep Notes (GCA ONLY)**

- The segment table shown on the dialog is **READ-ONLY**.
- Learn how to **Create and edit the Segment Sweep table**.
- **Independent IFBW** and **Power** are NOT available.
- **X-axis point spacing** is available beginning with A.09.10.

**Data Acquisition Mode**

Specifies HOW the gain compression data is collected.

**SMART Sweep**

- At each frequency, input power is 'intelligently' adjusted to find a measured gain equal to the target gain.
- Faster and more accurate than 2D sweeps to measure Gain Compression point at a number of frequencies.
- Learn ALL about SMART Sweep

**2D (two-dimensional) Sweeps**

- **Sweep Power per Frequency** Performs a series of power sweeps at each successive frequency.
- **Sweep Frequency per Power** Performs a series of frequency sweeps at each successive power level.
- Learn ALL about 2D sweeps

**Sweep Settings**

Click each to learn more about these settings.
- **Number of points** Number of frequency points to measure. The Frequency points may be limited due to the number of specified Power points. See Data Points Limit.

- **IF Bandwidth** Set this value to yield acceptable trace noise when measuring gain at the linear power level. This level of noise contributes directly to the accuracy of compression point. A lower value (narrower IFBW) allows for more accurate, but slower, measurements. See GCA Measurement Tips to see how to best set IFBW.

- **Start / Stop, Center / Span** frequencies. Set the frequency range over which to measure Gain compression.

### Data Points Limit

The maximum number of measurement data points depends on Acquisition method and Compression method as follows:

<table>
<thead>
<tr>
<th>Compression method</th>
<th>SMART sweep</th>
<th>2D sweep</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of frequency points is reduced to ensure the total number of data points does not exceed the specified limit. Frequency points = maxpoints/2.</td>
<td>Number of power points is reduced to ensure the total number of data points does not exceed the specified limit.</td>
<td></td>
</tr>
<tr>
<td>Data points = freq points</td>
<td>Data points = freq points</td>
<td>Data points = (freq. points) * (power points)</td>
</tr>
<tr>
<td>Max  = 100,001</td>
<td>Max  = 100,001</td>
<td>Max power points = 2,001</td>
</tr>
<tr>
<td>Data points = 2 * freq points</td>
<td>Max  = 100,001</td>
<td>Max data points = 100,000</td>
</tr>
<tr>
<td>Data points = freq points</td>
<td>Max  = 100,001</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** Although the dialog box will allow you to enter any number of frequency or power points, the values are checked when **OK** or **Apply** is pressed. If a limit is exceeded, the relevant data points are reduced to the maximum allowable number without warning.
Configures RF power and Power Sweep settings for Gain Compression measurement.

**Power ON (All channels)** Check to turn RF Power ON or clear to turn power OFF for all channels.

**Input Port**

Select the VNA port that is connected to the DUT Input.

**Linear Power Level** The input power that yields the linear gain of the DUT. The linear gain is used as the reference gain when calculating the **Compression from Linear Gain**. Input match is also measured at this power level.

**Source Attenuator** Specifies the attenuator setting associated with the port connected to the input of the DUT. This attenuator will affect the range of available power into the DUT. Learn more about Source Attenuation.

All VNA channels in continuous sweep must have the same attenuation value. Learn more.

**Receiver Attenuator** Specifies the attenuator setting for the receiver associated with the output of the DUT. When the power into the receiver test port is around +10 dBm, the VNA receiver may be in compression. However, with receiver attenuation, lower input power levels may become too noisy to make accurate power measurements. In this case, lower IFBW to reduce noise. Learn more about Receiver Attenuation. (This only applicable for PNA and M9485A)

**Source Leveling Mode** Specifies the leveling mode. Choose Internal or Receiver R1. Learn how to configure Receiver Leveling. Open Loop should only be used when doing Wideband Pulse
Output Port

Select the VNA port that is connected to the DUT Output.

Reverse Output Power  Sets power level into the output of the DUT for reverse sweeps. Port power is automatically uncoupled.

Reverse power is applied to the DUT ONLY under the following conditions. Otherwise, this setting is ignored.

- When Linear Output Match or Linear Reverse Isolation parameters are requested.
- When Full 2-port correction is used. You can perform a full 2-port cal and downgrade to an Enhanced Response Cal to prevent reverse power from being applied to the DUT. Learn more.

Source Attenuator  Specifies the attenuator setting for the port connected to the DUT output. This setting will affect the range of available power at the DUT output port.

  Auto  - Selects the proper attenuation setting automatically.

Receiver Attenuator  Specifies the attenuator setting for the receiver associated with the DUT output port. (This only applicable for PNA and M9485A)

Source Leveling Mode  Specifies the leveling mode. Choose from: Internal (normal operation) or Open Loop (used only for Wideband Pulse measurements).

Power Sweep

Power Points  Number of power points to measure for 2D acquisition modes. The Power Points may be limited due to the number of frequency data points. See Data Points Limit. This setting is NOT available in SMART Sweep, which uses only enough power points to find the specified compression level.

Start and Stop Power

- 2D sweep  In Backoff, X/Y, and Compression from Max Gain methods, sets the range of power levels that are applied to the DUT to find BOTH the Reference Gain and Compression point. Make sure this range is wide enough to include both. For example, if the Backoff level is 10 dB, then the power range must be greater than 10dB. Otherwise, GCA will report a compression value using the closest reference gain and compression point, which may be inaccurate. In Compression from Linear Gain, the reference gain is measured at the Linear...
Power Level, so the Start and Stop power levels are used to find the compression point.

- **SMART sweep** Sets the range of power over which GCA will search for the compression point. The reference gain is found using the Linear Power Level, Backoff, and X values, depending on the Compression Method. To reduce the number of iterations that are required to find the compression point, limit the Start / Stop power range to the input levels that will achieve compression. Do not include the linear region.

**Note:** If your DUT requires more input power to achieve compression below 3.2 GHz, use the PNA-X Hi-power mode, available from the RF Path Configuration dialog. The disadvantage to this is higher harmonic content.

**Power Step (Size)** Calculated value from current Start, Stop, and Points settings. This setting can NOT be changed directly.

**Path Configuration** click to launch the RF Path Configuration dialog.

---

**Compression tab - Gain Compression** dialog box help

**Compression Method**

Learn ALL about these Compression Methods

- **Compression from Linear Gain** The specified compression **Magnitude Level** is measured from the linear gain. The linear gain is measured using the **Linear Input Power** that is specified on the **Power tab**.
• **Compression from Max Gain** The specified compression level is measured from the maximum gain level. In SMART sweep, the Max Gain value is updated as each iteration occurs. To increase the chances of measuring the actual maximum gain of the amplifier, **Safe Sweep** should be invoked using low Coarse and Fine increments.

• **Compression from Back Off** This compression method uses the Compression Level and Back Off values for finding the compression point.

• **X/Y Compression** This compression method uses the specified parameters (X and Y) as the criterion for finding the compression point.

• **Compression from Saturation** Similar to Compression from Max Gain, except the specified compression level is measured from the maximum power out level. Use this method to better find the compression point when measuring amplifiers with non-monotonic gain. In SMART sweep, the Max power out value is updated as each iteration occurs. To increase the chances of measuring the actual maximum power out of the amplifier, **Safe Sweep** should be invoked using low Coarse and Fine increments.

**Phase**

When **Compression from Linear Gain** is selected and the **Data Acquisition Mode** in the Frequency tab is set to **Sweep Power Per Frequency (2D)** or **Sweep Frequency Per Power (2D)**, the following Phase functions are enabled.

• **Magnitude Only** Measures compression against magnitude.

• **Phase Only** Measures compression against phase.

• **Magnitude or Phase** Measures compression against magnitude or phase.

• **Phase Level** Specifies the phase to interpolate compression point.

• **Phase Details...** Accesses the Advanced Phase Settings dialog.

**Gain Compression Measurement Class Dialog**

![Advanced Phase Settings](image)
Gain Compression Converters Measurement Class Dialog

Compute Linear Power from Percent of Span Specifies the aperture as a percentage of span to compute linear input power.

Smooth Power Sweep Using Enables the power sweep to be smoothed.

Aperture Aperture used for smoothing the power sweep.

Use Reference Mixer Enables a reference mixer. To improve the noise of the phase measurements in Gain Compression Converters (GCX), an optional user supplied external reference can be added to the R1 loop. The LO for the reference mixer should be common with the LO for the DUT mixer. After this hardware is added, and this function is enabled, the GCX application will use the reference mixer to improve the phase measurements.

Force Source Power Out Port 1 This function is used when GCX channels with a reference mixer and one GCA channel without a reference mixer are set up simultaneously and you want to use the same path configuration for all channels. This remaps the source port to Port 1.

SMART Sweep

Learn ALL about Smart Sweep.

Tolerance Specifies an acceptable range for measuring the compression level. Reducing this value can significantly increase the number of iterations that are required to find the compression point.

Maximum Iterations Specifies the maximum number of power search iterations SMART Sweep
is allowed. Reducing this value can cause SMART sweep to terminate before all compression levels are found to within the specified tolerance.

**Show Iterations**  When checked, the compression parameter traces are updated at the completion of each power search iteration. When cleared, compression parameter traces are updated when SMART Sweep completes the power search iteration process.

**Read DC at Compression Point**  When checked, only the DC readings at the compression point in the last iteration of a smart sweep will be taken. By default, the DC traces are read at each point in the sweep. In some cases, the user may only want to read the DC meter at the compression point to improve measurement speed.

---

**2D Sweep - Compression Point Interpolation**

When a 2D Sweep is selected (on the Frequency tab), check this box to calculate and display interpolated compression traces.

The **Target gain** is calculated using a complex linear ratio between the two closest measured values. All compression parameters are then interpolated using this same ratio.

Clear the box to display compression parameters for the closest compression point, either high or low, to the level specified in the Compression Method setting.

**End of Sweep Condition**  Specifies the power level applied to the DUT at the completion of a GCA measurement.

GCA performs numerous power and frequency sweeps on the DUT during the overall measurement process. This setting has no affect on these intermediate sweeps. This setting only applies at the end of the very last sweep in the GCA channel.

In addition, this setting applies ONLY to the GCA channel. All other channels operate independently of this setting. Therefore, the power applied to the DUT after all channels have been measured may be different from this setting.

Choose from:

- **Default**  Use the default VNA method.  Learn more.
- **RF OFF**  RF power is turned off when GCA completes a measurement cycle.
- **Start Power**  RF power is set to the start power level.
- **Stop Power**  RF power stays at the stop power level.

**Settling Time**
Used ONLY in SMART Sweep when Back Off or X/Y compression algorithms are selected.

This setting allows additional dwell time when the input power changes from the back-off level to the compression level. Learn more.

**SMART Sweep Safe Mode** dialog box help

For use with SMART Sweep ONLY.

When enabled, Safe Sweep increases the input power to the DUT by the specified amounts, allowing the compression point to be achieved gradually. While this will increase the number of iterations required to achieve compression, it also minimizes the possibility of driving the DUT too far into compression.

**Note:** Safe Sweep does **NOT** minimize the dramatic change in input power with Backoff and XY method. However, Safe Sweep with Backoff and XY methods **DOES** prevent the DUT from being exposed to too much input power. Learn more.

**Safe Mode (Enable)** Check to enable Safe Sweep.

**Coarse Increment** Sets the maximum change in input power, up or down, which will be applied to the DUT from one iteration to the next. Default = 3.0 dB.

Without Safe Sweep, the maximum change in input power can be the entire Backoff or X value when using these compression methods.

**Fine Increment** Once the Fine Threshold has been achieved, this becomes the maximum change in input power, up or down, which will be applied to the DUT. Default = 1.00 dB

**Fine Threshold** Specifies the compression level in which Safe Sweep changes from the COARSE to the FINE increment. Default = 0.5 dB. This means that, by default, the VNA uses the Fine Increment adjustment when compression reaches 0.5 dB.
**Max Output Power** To protect the VNA from damage, when the VNA port that is connected to the DUT Output measures the specified value, the input power to the DUT is no longer incremented at that frequency. In these cases, the compression point would probably not be achieved.

**DC Parameters** Select a DC device from the pulldown list. (Modification Distortion Only)

*Note:* When a DC meter is added, it will be displayed in the New Trace dialog and SMART Sweep Safe Mode dialog (in DC Parameters pull down menu).

**Max DC Power** Enter the limit value of the DC device. When the Max DC Power is above the limit value, the power sweep is clipped and returns to the previous safe value. The units are changed automatically per the Type setting in the DC meter dialog. (Modification Distortion Only)

---

**Compression Analysis**

Compression Analysis changes the current trace into a power sweep trace at a specified CW frequency. The current parameter and acquisition method is unchanged. For example, with a CompGain21 trace displayed and SMART Sweep selected, enable Compression Analysis. The trace becomes a power sweep trace at the specified CW frequency. The Y-axis displays S21 Gain at each X-axis power point.

When Smart sweep is used, a complete power sweep is not performed, but only the data points that are required to find the compression point. To see a traditional power IN vs power OUT compression sweep, use one of the 2-D acquisition methods.

You can create PNOP or PSAT markers on a CompOut trace with Compression Analysis mode ON. Learn more.

---

**How to perform Compression Analysis**

With any compression parameter (such as CompGainS21) displayed:

**Using Hardkey/SoftTab/Softkey**

1. Click **Math > Analysis > Compression Analysis**....

**Using a mouse**

1. Click **Response**
2. Select **Math**
3. Select **Compression Analysis**
Notes: When an S21 or S11 trace is active, any compression parameter (such as CompGainS21) must also be displayed.

Compression Analysis is NOT allowed for S12 or S22 traces.

Scroll up to learn more about Compression Analysis.

Analysis Frequency: CW  Enter a frequency to use for the compression analysis trace.

Compression Analysis  Check to perform compression analysis. A compression trace is displayed at the Analysis (CW) Frequency.

Use Discrete Frequencies  Check to allow Analysis Frequencies at only the discrete points where data is measured. Clear to allow Analysis CW Frequencies that are interpolated from the data points. Then select ANY CW frequency between the start and stop frequencies of the GCA channel.

X-Axis

- Use Measured Pin  The X-axis displays the actual power that is applied to the DUT after match correction and R-channel drift correction.

- Use Source Pwr Settings  The X-axis displays the power level of the stimulus.

Saving GCA Data

Beginning with VNA release A.08.20, GCA data can be saved to a *.csv file in both 2D and SMART Sweep modes (previously only 2D modes). Also, a Delta Gain, AI1, and AI2 columns have been added.
to the data. Learn about ADC parameters.

How to save GCA data

With a GCA Compression trace active:

Using Hardkey/SoftTab/Softkey

1. Press Save Recall > Save Other > Save Data...
2. File Type= CSV Formatted Data (*.csv).

Using a mouse

1. Click File
2. Select Save Data
3. File Type= CSV Formatted Data (*.csv).

Notes

- This data type can be read by spreadsheet programs, such as Microsoft Excel.
- Data from the last complete sweep is saved to the specified *.csv file.
- If calibration is turned ON when the file is saved, then all data is calibrated. Otherwise, raw data is saved.
- All *.csv data saves include a reference power level sweep at the beginning of each frequency data.

SMART Sweep data with 5 iterations and 3 frequency points. The yellow highlight is added here for readability.

When saving or recalling 2D data:
- When Linear Input Power EQUALS Start Power, then the number of data points (rows) / freq = num power points.

- When Linear Input Power does NOT EQUAL Start Power, the number of data points (rows) / freq = num power points + 1.

- Make these selections on the GCA/GCX Power tab dialog.

GCA Measurement Tips

There are many settings in the Gain Compression Application. Here are a few tips when using GCA to learn as much as possible about the compression characteristics of your DUT in the most efficient manner.

DUT Compression Characteristics and GCA

Although GCA provides excellent results with a wide variety of amplifiers, it works best with amplifiers which have a monotonic compression curve. In some cases where the compression curve is not monotonic, for example if the amplifier gain expands before it compresses, the correct compression level may not be found.

To help a SMART sweep find the correct compression point, limit the Start and Stop power levels around the anticipated compression point. Learn more.

The following two power-sweep traces are examples of non-monotonic gain:

![DeltaGain Graph](image-url)
A DeltaGain trace is the best way to see how closely GCA is actually measuring to the desired compression level. In addition, you can view the phase of DeltaGain to see the phase deviation between the compressed gain and the reference gain. DeltaGain is calculated as:

- \[ \text{DeltaGain} = \frac{\text{Measured Gain (watts)}}{\text{Ref Gain (watts)}} \]
- In LogMag format: \[ \text{DeltaGain} = \text{(Measured Gain)} - \text{(Ref Gain)} \]

With SMART Sweep, DeltaGain (in LogMag format) shows how soon certain frequencies achieve the specified tolerance. Learn more.

Some other settings which may be helpful:

- Trigger source: Manual allows you to analyze data and make adjustments while allowing the device to cool.
- Construct Limit Lines around the compression point at the tolerance level.

The following image shows a DeltaGain21 trace using SMART Sweep. The Limit Lines were added manually.

In the above image:
### Relevant Settings

<table>
<thead>
<tr>
<th>Method = Compression From Linear Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compression level = 1</td>
</tr>
<tr>
<td>Iteration Tolerance = 0.05 dB.</td>
</tr>
<tr>
<td>Maximum Iterations = 10</td>
</tr>
</tbody>
</table>

### Displayed Results

- A data point on -1.00 indicates that, at that frequency, the exact compression level (1 dB) was measured.
- Several frequencies did not achieve the specified tolerance (0.05 dB) before the Max Iterations (10) was reached.
  - **FAIL** and red data points outside the limit lines.
  - Nine dots (....) indicate that 90% of the data points achieved the specified compression level.
  - **one !** indicates that 10% of the data points did not achieve compression.
  - Learn more about the Iteration Counter and annotation.

### SMART Sweep Tips

- Compression from Linear Gain is the easiest compression method to understand and control in SMART Sweep. Learn more.
- If SMART Sweep requires more than twenty iterations, this is an indication that something is wrong. Try changing the Tolerance setting, Frequency Range, Start / Stop power range, IF bandwidth, or Dwell Time.
- If the number of iterations required to achieve the desired compression level changes significantly from one set of measurements to the next, this could be due to other effects, such as heating. Try increasing the dwell time or using a wideband pulse measurement configuration.
- If the DUT should not be significantly overloaded into compression, or the changes in the input power should be limited, use Safe Sweep mode with Deviation from Linear Gain compression method.

### Single Frequency Macros

**Note:** Beginning with VNA rev. A.09.00, the Compression Analysis feature provides an easier method of viewing a traditional power sweep at a single frequency than the GCA macros. However, the Macros are still maintained on the VNA hard drive.

The macros perform a single power sweep on the DUT using a standard channel with corresponding stimulus settings. The macro can show measurement differences from the compression analysis traces.
due to bias/thermal/settling effects of the DUT. So, the macro can help confirm a DUT is exhibiting some type of settling behavior which will need to be handled in some way.

Also, the macro is a great GCA programming example.

With a 2D sweep (NOT SMART Sweep) a script that is stored on the VNA hard drive automatically creates a traditional power sweep measurement in a standard channel using the same stimulus setting as the GCA channel. Use a marker in the GCA channel to specify the frequency for the measurement.

The script has two modes of operation:

1. **View Mode** displays all of the previous 2D sweep data at that frequency.

2. **Measure Mode** performs a new measurement at that frequency.

Both modes create a new S-Parameter channel using the same stimulus settings as the GCA channel, including port power, attenuator, IF Bandwidth, and dwell settings. The new channel does not support calibration or pulse characteristics.

To see noise on a measurement, use the **Measure** macro in continuous sweep. Adjust the IFBW and averaging until the noise versus sweep speed meets your needs.

To see other effects of your DUT at a specific frequency, use the **View** macro and the **Measure** macro with 2D sweep mode. Both macros present data using a standard channel. The View macro shows 2D data at a specific frequency, while the Measure macro shows freshly-measured data at the same frequency. Ideally, the data from these two would be identical. However, changes in your DUT behavior due to heating or other effects can cause these to be different. If significant differences exist, try:

- Using the 2D Frequency per Power setting rather than Power per Frequency
- Adjusting the dwell time
- Adjusting IFBW
- Use a wideband pulse configuration

**How to setup the Macros**

Each macro must be setup separately.

1. Press **Macro** > **Key Setup** > **Macro Setup**.

2. Select a blank line, then click **Edit**.
3. In **Macro Title**, type a short description such as Meas GCA or View GCA.

4. Click **Browse**, then navigate to C:\Program Files(x86)\Keysight\Network Analyzer\Applications\GCA\GCA.vbs

5. In Macro run string parameters:

   1. Type **M** for the Measure macro or **V** for View macro.
   
   2. Optional: Supply the following additional parameters in any order:

      - To run the program from a remote computer, specify the full computer name of the VNA.
      
      - Channel in which to create the measurement. If not specified, Measure is created in Ch30 and View is created in Ch31.

      - Example: Run string parameters for the Measure macro run from a remote computer in Channel 5.----M MyVNA 5.

6. Click **OK**.

**How to run the Macros**

On a GCA channel:

1. Create a 2D sweep. Either Power per Freq or Freq per Power. Both macros always create a power sweep at the frequency of interest.

2. Create a CompIn trace.

3. On the CompIn trace, right-click and select **Add Marker**. Drag the marker to the frequency of interest.

4. Press **Macro**, then select either by the short description your provided in Step 3.
Gain Compression is measured on Converters in the same manner as it is measured in Amplifiers. Also, the Mixer/Converter setup is very similar to that of SMC (Scalar Mixer Converter application).

In this topic (unique for GCX):

- Requirements and Limitations
- Using GCX
- Create a GCX Measurement
- Valid Mixer Configuration / Sweep Type Combinations
- Measurement Parameters offered in GCX
- GCX Calibration

The following Gain Compression for Amplifiers information is relevant for learning about GCX:

- Gain Compression Concepts
- Understanding the GCA Displayed Traces
- Compression Methods
- Acquisition Modes
- Compression Analysis
- Saving GCA Data
- GCA Measurement Tips

The following dialog setup tabs are shared with other applications:

- Frequency tab (GCA topic)
- Power tab (GCA topic)
- Compression tab (GCA topic)
  - Safe Sweep Mode dialog (GCA topic)
Requirements and Limitations

GCX requires Option S93086A/B (Gain Compression).

Limitations:

- Number of points limited to 100,001 for two-dimensional acquisitions, 50,000 points for SMART Sweep.
- Linear and CW sweep ONLY - No Power, Log, or Segment sweep. Learn more.
- GCX does NOT provide any built-in image rejection techniques. You should provide image rejection hardware if necessary.
- Stepped sweep mode only.
- Does NOT support Narrowband Pulse measurements using the integrated Pulse setup dialogs.

The following VNA features are NOT available with Gain Compression on Converters:

- ECal User Characterization
- Time Domain
- Balanced measurements
- Save Formatted Citifile data.
- Save SnP data.
- Port extensions
- Some Fixturing Features
- External Test Set Control (Option S93551A/B)
- Integrated Narrowband or Narrowband Pulse App
- Independent IFBW, Power Levels, or Sweep Time in a segment table is NOT supported.
**Using GCX**

The following is a general procedure for performing a GCX measurement. The challenge with GCX is configuring a measurement that yields the true compression performance of YOUR DUT. This requires knowledge of the Gain Compression and Mixer settings, and knowledge of the DUT.

See specific dialog boxes below for details.

1. Disconnect the DUT if preset or default power levels may damage the VNA or DUT.
2. **Preset** the VNA, or configure a suitable **User Preset** that will be safe in case the DUT is connected.
3. Create a GCX channel. **Learn how**. The default trace is SC21.
4. Start the GCX Setup dialog and configure the measurement settings based on the DUT, adapters, attenuators, booster amplifiers, and fixtures to be used in the measurement. To start the dialog, click **Stimulus**, then **Frequency**, then **GCX Setup**. **Learn about the setup dialogs**.
5. Save the **instrument state** (optional).
6. Connect the DUT. Inspect the measurement to ensure the DUT is operating as expected.
7. Add GCX compression and mixer parameter traces. **Learn more**.
8. Adjust the measurement settings to yield satisfactory compression results. **See GCA Measurement Tips**.
9. Start and complete the **GCX Calibration wizard**.

**Create a GCX Measurement**

1. Press **Meas** > **S-Param** > **Meas Class**....
2. Select **Gain Compression Converters**, then either:
   
   - **OK** delete the existing measurement, or
   
   - **New Channel** to create the measurement in a new channel.

3. A default SC21 measurement is displayed. To select additional parameters to display, press **Trace**, select a trace, press **Meas**, then select a parameter. **Learn more about GCX Parameters**.
How to start the Gain Compression for Converters Setup dialog

Using **Hardkey/SoftTab/Softkey**

1. Press **Freq > Main > GCX Setup...**

Using a mouse

1. Click **Stimulus**
2. Select **GCX Setup...**

**Programming Commands**

Valid Mixer Configuration / Sweep Type Combinations

The following are the **Valid Sweep Type / Mixer Configurations**.

<table>
<thead>
<tr>
<th>Sweep Type</th>
<th>Input</th>
<th>LO</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>Swept</td>
<td>Fixed</td>
<td>Swept</td>
</tr>
<tr>
<td></td>
<td>Swept</td>
<td>Swept</td>
<td>Fixed</td>
</tr>
<tr>
<td>CW</td>
<td>Fixed</td>
<td>Fixed</td>
<td>Fixed</td>
</tr>
<tr>
<td></td>
<td>Fixed</td>
<td>Swept</td>
<td>Swept</td>
</tr>
</tbody>
</table>

For determining a valid mixer configuration with 2 LOs, one Fixed LO and one Swept is equivalent to having a single-stage Swept LO.

If you create an invalid Sweep Type / Mixer Configuration, a red message appears like the following:

**ERROR: Input range must be swept in Linear sweep mode.**

If this occurs, change the **Sweep Type** on the **Frequency tab**.

See other rules for configuring a mixer.

**GCX Measurement Parameters**

**Note:** The following table assumes: DUT **Input** = VNA **port 1** and DUT **Output** = VNA **port 2**.

When the Port mapping is different, the parameters in GCX are updated accordingly. For example, with Input = port 2 and Output = port 1, then "CompIn12" would be displayed.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mixer Parameters</strong></td>
<td></td>
</tr>
<tr>
<td>SC21</td>
<td>Linear Conversion Gain</td>
</tr>
<tr>
<td>SC12</td>
<td>Reverse Conversion Gain</td>
</tr>
<tr>
<td>S11</td>
<td>Input Match</td>
</tr>
<tr>
<td>S22</td>
<td>Output Match</td>
</tr>
<tr>
<td><strong>Compression Parameters</strong></td>
<td></td>
</tr>
<tr>
<td>CompIn21</td>
<td>Input power at the compression point.</td>
</tr>
<tr>
<td>CompOut21</td>
<td>Output power at the compression point.</td>
</tr>
<tr>
<td>CompGain21</td>
<td>Gain at the compression point</td>
</tr>
<tr>
<td>CompS11</td>
<td>Input Match at the compression point.</td>
</tr>
<tr>
<td>RefS21</td>
<td>Linear Gain value used to calculate the compression level. This is calculated differently depending on the compression method.</td>
</tr>
<tr>
<td>DeltaGain21</td>
<td>CompGain21 MINUS Linear Gain (in Log Mag format). This trace can be used to learn a lot about the DUT compression point. Learn more.</td>
</tr>
<tr>
<td><strong>Unratioed</strong> - Absolute test port receiver measurements. Learn more.</td>
<td></td>
</tr>
<tr>
<td>IPwr</td>
<td>Input power measured at DUT-IN @ Input frequency</td>
</tr>
<tr>
<td>OPwr</td>
<td>Output power measured at DUT-OUT @ Output frequency</td>
</tr>
<tr>
<td>RevIPwr</td>
<td>Input power measured at DUT-OUT @ Output frequency</td>
</tr>
<tr>
<td>RevOPwr</td>
<td>Output power measured at DUT-IN @ Input frequency</td>
</tr>
<tr>
<td><strong>ADC Parameters</strong> - Learn more.</td>
<td></td>
</tr>
<tr>
<td>AI1</td>
<td>Measured at the specified Linear Input level.</td>
</tr>
<tr>
<td>AI2</td>
<td>Measured at the specified Linear Input level.</td>
</tr>
<tr>
<td>CompAI1</td>
<td>AI1 at Compression</td>
</tr>
<tr>
<td>CompAI2</td>
<td>AI2 at Compression</td>
</tr>
</tbody>
</table>

**GCX Setup Dialogs**

All of the GCX Setup tabs are shared with other applications.

- Frequency tab
- Power tab
- Compression tab
  - Safe Sweep Mode dialog
A GCX Cal is conceptually the same as a Gain Compression Calibration. This includes the ability to perform or downgrade to an Enhanced Response Cal. Learn how.

The following Guided Cal Wizard pages are unique to GCX:

**GCX Calibration Setup** dialog box help

- **Waveguide/In-fixture/On-Wafer Setup** Starts the following dialog box.

- **Independent power cals for input and output ports (no thru)** Check if a Thru standard is NOT available. During the power cal, you will be prompted to connect the power sensor to the Input, then the Output port.

**Additional Power Cal Steps**

- **Enable LO1 / LO2 Power Cal** Check when LO1 / LO2 is controlled (on the Mixer Setup tab) to perform a Power Cal on the source.

**Note:** Phase Correction is NOT allowed for GCX measurements.
Waveguide/In-fixture/On-Wafer Setup dialog box help

This dialog box appears ONLY if you checked the Waveguide/In-fixture/On-Wafer Setup box in the previous Cal Setup dialog.

Allows you to embed or de-embed circuit networks on the input and output of your mixer under test.

For Network1 (Input) and Network2 (Output) select Embed, De-embed, or None.

Browse  Click to navigate to the .S2P file that models the network to embed or de-embed.

To Embed or De-embed

- **De-Embed** when there is a 2-port network that needs to be connected during the measurement, but it is NOT present during the calibration. An example might be when you do not have standards for a characterized test fixture, so you calibrate without the fixture, and make measurements with the DUT in the fixture. De-Embedding during the calibration extends the calibration reference plane to include the 2-port network.

- **Embed** when there is a 2-port network that to be disconnected during the measurement, but is present during the calibration. An example might be when a characterized adapter is required during the calibration but NOT present during measurements of the DUT. Embedding during the calibration retracts the calibration reference plane to exclude the 2-port network during the measurement.

Notes

- The S2P file for Network1 (on the input of the mixer), must cover the Input frequency range. The S2P file for Network2 (on the output of the mixer), must cover the Output frequency range.

- The frequency range of the S2P file must be the same, or larger than, the frequency range of the measurement. If more frequencies are included in the file, and the data points do not exactly match those of the measurement, interpolation will be performed.

- As in the image on the dialog (above), in all cases:
  - Port 1 of each network is assumed to be connected to the VNA.
  - Port 2 of each network is assumed to be connected to the DUT.
Select DUT Connectors and Cal Kits dialog box help

Allows you to specify the connector type and Cal Kit for each DUT port.

**Port n**  For each listed VNA port, specify the DUT connector type and gender, and the Cal Kit to use.

**input pwr sensor**  Specify the connector type of the power sensor. Select **Ignored** to not compensate for the effects of the adapter that may be necessary to connect the power sensor to the input reference plane.

**output pwr sensor**  Available when **Independent power cals for input and output ports** is checked on the GCX Calibration Setup dialog. Specify the connector type of the power sensor. Select **Ignored** to NOT compensate for the effects of an adapter that may be necessary to connect the power sensor to the output reference plane.

**De-embed input power sensor adapter**  Check to measure, then remove the effects of the adapter that is used to connect the power sensor to the calibration reference plane.

**Source Cal Settings**  Click to start the Source Cal Settings dialog. These settings allow you change ALL Source Cal and Power Meter settings.

**Note:** If your DUT connectors are:

- **Waveguide**  Change the system impedance to 1 ohm before performing a calibration. See **Setting System Impedance**.
- **Not listed** (male and female)  Select **Type A** as the connector type. Type A requires a calibration kit file containing the electrical properties of the standards used for calibration (see **Calibration kits**).
- **Unspecified** (like a packaged device)  Select **Type B** as the connector type. Type B requires a calibration kit file containing the electrical properties of the standards used for calibration (see **Calibration kits**).

**Modify Cal**  Check, then click Next, to start the Modify Frequency Cal dialog.
Gain Compression Calibration

The GCA Calibration Wizard guides you through a calibration of GCA or GCX channel. The procedure is the same regardless of the Gain Compression Settings. Option 086 or S9x086A/B is required.

- A Source Power Calibration is performed first.
- Then, your choice of a Full 2-port Cal or an Enhanced Response Cal.

See Also

Gain Compression Application
Gain Compression for Converters
Calibration Programming commands

How to start a GCA Calibration

Using Hardkey/SoftTab/Softkey

1. Press CAL > Main > Smart Cal....

Using a mouse

1. Click Response
2. Select Cal
3. Select Smart Cal...

Overview - GCA Source Power Cal

The GCA Calibration Wizard first performs a Source Power Cal. The GCA Source Power Cal is a little different from a standard Source Power Cal. Although GCA measurements are performed at many power levels, the GCA source power cal is performed at a single power level over the specified frequency span of your GCA measurement. The required source correction from that single power level is applied to ALL power levels. This method ensures that the 'absolute' power level being applied to the DUT is within the PNA-X source power linearity specification.

Although it is important for GCA to be able to set the absolute power level to the DUT, it is MOST important to be able to exactly measure the actual incident power. Therefore, during the GCA Source Power Cal, a receiver calibration is applied to the port 1 reference receiver, and indirectly to both test port receivers during the S-parameter calibration, correcting for impedance mismatch between the
power meter and the VNA source, and the DUT and the VNA source.

Although the cal process is also at a single power level, the dynamic accuracy of the PNA-X receivers is typically about +/- .05 dB, which is comparable to the accuracy of Keysight's best power sensors. This allows GCA to very accurately measure and report ALL power levels that are actually applied to the DUT.

**Full 2-port or Enhanced Response (ER) Cal**

By default, a full 2-port calibration is performed as part of a GCA and GCX calibration. However, you can change to an Enhanced Response Cal. The following issues may help you decide between these two Cal types:

- **Accuracy** A full 2-port correction is more accurate than ER when GCA measures linear gain. However, for non-linear measurements, ER yields identical compression values as a full 2-port cal, so this may not be a significant factor.

- **Measurement speed** An ER correction only requires measurements in the forward direction. The reverse parameters (usually S22 and S12) are not measured unless requested. With a full 2-port cal applied, all four S-parameters are measured, which requires an additional reverse sweep. Learn more.

- **Ease** A full 2-port cal is easiest with an ECal module. An ER Cal requires a Defined Thru or a Flush Thru Cal method. If these are possible, then an ER cal is easiest when using a mechanical Cal Kit.

- **High power** The test port damage level of a standard PNA-X is +30 dBm. Therefore, external attenuation may be required on the output of high power amplifiers, which degrades calibration accuracy for reverse (full 2-port) measurements. In addition, the external attenuation improves the DUT output / load match error, which allows a better uncorrected response and makes an Enhanced Response Cal the better choice.

- **DUT limitations** With an ER Cal applied, reverse measurements on the DUT are not performed unless requested.

**How to select Enhanced Response Cal**

At the Select DUT Connectors page of the GCA Cal Wizard:

1. Check **Modify Cal**, then click **Next**.
2. A Defined Thru or a Flush Thru Cal method must be selected.
3. Click **Cal Type/Std**s
4. Under Calibration type, select **EnhResp** (2 <= 1 refers to the receive port 2 and source port 1).

**Downgrade a Full 2-port Cal to Enhanced Response Cal** if you prefer to perform a Full 2-port cal, but not perform reverse sweeps on the DUT.
To change the correction on the channel from Full 2-port to Enhanced Response:

1. Press **CAL > Main > Correction Methods.**
2. Select **GCA EnhResp**, then **OK.**

![Select Calibration Type]

**GCA Cal Wizard**

![Select DUT Connectors and Cal Kits - GCA Cal dialog box help]

**Select DUT Connectors and Cal Kits - GCA Cal dialog box help**

This is a **standard Cal Wizard** page except for the following:

**Power Sensor** Specify the connector type and gender of the power sensor. When the power sensor connector is not the same type and gender as the DUT Port 1 connector, then an adapter is required to connect the power sensor to the port 1 reference plane during the Source Power Cal. An extra 1-port cal is performed to measure and correct for the adapter. No characterization S2P files are required.

- Select **Ignored** (at the bottom of the DUT Connectors list) to NOT compensate for the adapter.
- Select the Cal Kit that will be used for that process.

**Modify Cal** Check, then click **Next**, to Modify Cal (Standards AND Thru Method).
Source Cal Settings  Click to launch the Source Cal Settings dialog.

Learn more about GCA Source Power Calibration

---

**Gain Compression Calibration Step 1** dialog box help

**Power Level** at which to perform the Source Power Cal.

It is usually best to perform the Source Power Cal at 0 dBm because the power sensor is calibrated at that level.

However, if the Gain Compression measurement is performed entirely below or above 0 dBm, then perform the Source Power Cal at the **Stop** power which probably has the lowest level of measurement noise.

Learn more about GCA Source Power Calibration

The remaining Gain Compression Cal dialogs are the same as the standard SmartCal dialogs.

Return to [Gain Compression Application](#).
**iTMSA (Integrated True Mode Stimulus Application)**

This application is integrated into the VNA firmware as **S9x460A/B**.

- Features, Requirements, and Limitations
- True Mode Stimulus Concepts
- Using iTMSA
  - Create a Measurement
  - Set Power Level
  - Calibrate
  - Ratioed Receiver Measurements
  - Differential Input and Output Power Measurement

### Other VNA Applications

**Important Note**

When the first iTMSA measurement is created, an IFMUX Cal is performed, which takes a couple of minutes.

When finished, **Exit** then **restart** the VNA App.

If an IF board is replaced, this Cal should be performed again. To make this happen, delete `c:\users\public\network analyzer\IFMUX.txt`. When the next iTMSA measurement is created, the file will be recreated with new data.

### Features, Requirements, and Limitations

**Features**

- A seamless extension of existing VNA Balanced Measurement, but with True Mode stimulus.
- True Mode Stimulus measurements are performed in a standard S-parameter channel.
Requirements

- PNA-X and VNA (N522x) 4-port models
- Opt S9x460A/B (software option only); must be enabled.

Limitations with iTMSA

- Direct Receiver Access configuration of the front-panel loops is NOT allowed. This bypasses the coupler making it no longer possible to perform return loss measurement of the DUT (needed to align the two sources). However, inserting amplifiers or other components into the standard source path configuration at the front-panel loops IS allowed.
- Reversing the port couplers is NOT recommended due to the increased noise on the initial sweep for match correction. Good reflection measurements on all four ports is necessary to ensure adequate accuracy.
- Does not support 1-port measurements.
- Using true mode and measuring ADC inputs simultaneously may not result in accurate ADC measurements. In this case, adding two more ADC measurements to the display will result in accurate ADC measurements.

The following standard VNA features are **NOT** available with iTMSA measurements:

- **External Multiport Test Set Control** (Option S9x551A/B)
- **Time Domain Pulse measurements** in Wideband Pulse are NOT supported.
- **Segment Sweep**
- **Frequency Offset Measurements** (opt S9x080A/B)
- **mmWave Measurements** are allowed but with differential and S-Parameter measurements only. Absolute power measurements are not possible.

True Mode Stimulus Concepts

A balanced device is designed to receive input simultaneously across two ports. Standard VNA Balanced measurements apply stimulus to one port at a time, measures the responses, and calculates the theoretical balanced responses. Learn more about balanced measurements.

True Mode Stimulus uses two VNA sources to apply either truly differential (180 degree out-of-phase) or truly common (in-phase) signals across the input of a balanced device. VNA receivers measure the single-ended response at the output of the device and calculate the balanced response.

When operating in non-linear regions, a device may respond differently to single-ended stimulus than
to True Mode Stimulus. Thus, True Mode Stimulus capability allows you to understand when, and if, True Mode Stimulus is required.

For more detailed information on this measurement technique, see the following white papers (internet connection required):

- New Methods & Non-Linear Measurements for Active Differential Devices
- New Measurement Results and Models for Non-linear Differential Amplifier Characterization

**How iTMSA Works - Overview**

The following is an overview of how iTMSA gathers and displays True Mode Stimulus data:

1. **Initial sweep** is performed to gather the initial phase deviation between the two sources at the reference plane.

2. **Measurement** The phase deviation between the two sources is set to the required value and the response is measured. For a differential sweep, the phase deviation of the two sources is set to 180° at the reference plane. For a common sweep, the phase deviation is set to 0° at the reference plane.

A forward direction measurement requires 2 sweeps for differential and 2 sweeps for common. A reverse direction measurement requires the same. Therefore, a complete measurement in both directions requires 8 sweeps. The iTMSA traces are updated when all of the necessary data is gathered in each direction.

iTMSA computes the raw S parameters with the following matrix:

\[
\begin{bmatrix}
\frac{a_1}{d_1} & \frac{a_1}{d_2} & \frac{a_1}{c_1} & \frac{a_1}{c_2} \\
\frac{a_2}{d_1} & \frac{a_2}{d_2} & \frac{a_2}{c_1} & \frac{a_2}{c_2} \\
\frac{a_3}{d_1} & \frac{a_3}{d_2} & \frac{a_3}{c_1} & \frac{a_3}{c_2} \\
\frac{a_4}{d_1} & \frac{a_4}{d_2} & \frac{a_4}{c_1} & \frac{a_4}{c_2}
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
a_4
\end{bmatrix}^{-1} =
\begin{bmatrix}
S_{11} & S_{12} & S_{13} & S_{14} \\
S_{21} & S_{22} & S_{23} & S_{24} \\
S_{31} & S_{32} & S_{33} & S_{34} \\
S_{41} & S_{42} & S_{43} & S_{44}
\end{bmatrix}
\]

**Legend:**

- d1: differential stimulus in balanced port 1.
- d2: differential stimulus in balanced port 2
c1: common stimulus in balanced port 1.

c2: common stimulus in balanced port 2.

**Using iTMSA**

A Standard (default) S-Parameter channel is used for iTMSA measurements.

**How to create an iTMSA trace:**

1. Preset the VNA

2. Then do the following:

**Using Hardkey/SoftTab/Softkey**

1. Press **Meas > Balanced > Topology...**

**Balanced DUT Topology / Logical Port Mapping (with iTMSA Option S9x460A/B) dialog box**

Learn about iTMSA.

Topology
The following topologies are available in True Mode Stimulus.

- **BAL** DUT has a single balanced port.
- **BAL-BAL** DUT has two balanced ports.
- **BAL-SE** DUT has one balanced port and one single-ended port.
- **BAL-SE-SE** - DUT has one balanced port and two single-ended ports.
- **SE-BAL** DUT has one single-ended port and one balanced port.
- **SE-SE-BAL** - DUT has two single-ended ports and one balanced port.

- **All SE**  
  (Changes all entries to Single-ended)

- **All BAL**  
  (Changes all entries to Balanced)

- **All Unused** (Changes all entries to Unused) This is a convenience feature to help set up custom topologies, but it is not possible to have only Unused ports. If the **OK** button is clicked and all ports are listed as Unused, the topology will be set up to have 1 SE Port.
- **Custom**
  (Maps the physical VNA ports to a device of balanced and single-ended logical ports for multi-port systems with greater than 4 ports.)

A balanced port can be any one of four physical port combinations:

1 - 3
1 - 4
2 - 3
2 - 4

**Port Z**

Provide an enable for both Common and Differential Conversion and SE Port Z Conversion.

**Balanced Port**  Shows all ports defined on the balanced topology page.

**Default Z**  Shows the default impedances that will be applied if the port Z conversions are not enabled. The SE Default Z always equals the System Zo defined for the VNA. The Differential and Common Default Z will display values calculated from the single-ended port impedances.
**Converted Z** User may enter the real or imaginary component of the impedance.

**Warning Dialog** The dialog is displayed if "Conversion" is enabled and "Apply Fixtures" is currently disabled.

**Offset**

**Balanced Port** Set the phase or power offset for the balanced port.

**True Mode** Shows whether True Mode is On or Off.
**Power Offset**  Specify power offset. Range is +/- 20 dB. This is in addition to the power level that is specified using the [Power and Attenuators dialog](#). Offset Power is NOT reflected on the power dialog nor on the X-Axis during a power sweep. A power offset may only be applied to a True Mode port.

**Apply Power Offset as Fixture**

When unchecked, output calculations are performed and displayed as though there is no stimulus power offset.

When checked, output calculations are performed and displayed using the power offset that is applied to the DUT. Use this setting to compensate for a component or fixture that may present a magnitude loss before the DUT.

**Phase Offset**  Specify offset for the balanced INPUT port. This is in addition to the standard offsets, which are 180° offset for the differential stimulus sweeps, and 0° for the common stimulus sweeps. A phase offset may only be applied to a True Mode port.

**Apply Phase Offset as Fixture**

When unchecked, output calculations are performed and displayed as though there is only the standard offset. Although additional Phase Offset is applied to the stimulus, it is ignored in the calculations of balanced differential and common mode output signals.

When checked, output calculations are performed and displayed using the actual phase offset that is applied to the DUT. Use this setting to compensate for a component or fixture that may present a phase delay before the DUT.

**Sweep**
Available only when **Sweep type = CW**.

Available ONLY for balanced pairs.

The phase of the selected active source port is swept relative to the phase of the other source port in the balanced pair. The active source port uses the logical port number.

Phase Sweep is similar to phase offset, except that for each data point, the phase 'offset' is incremented.

For example, with the topology that is selected in the above image: (Bal-Bal) Logical port 1 = VNA ports 1 and 3. For a phase sweep with 7 data points, from 0° to 180°, the phase difference between port 1 and port 3 increments 30° with each data point:

<table>
<thead>
<tr>
<th>Data point</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delta phase</td>
<td>0°</td>
<td>30°</td>
<td>60°</td>
<td>90°</td>
<td>120°</td>
<td>150°</td>
<td>180°</td>
</tr>
</tbody>
</table>

**Enable Balanced Phase Sweep**  Check to enable phase sweep for the specified port. If enabled and the VNA sweep type is not CW, the following dialog is displayed to turn on CW Sweep Type.
Start / Stop Phase  Enter phase values in degrees. Each sweep will start and stop at these settings.

Offset as Fixture

In the following image, the VNA phase sweep is shown as a phase shifter that is "virtually" located EITHER before the reference receiver (top) OR after the reference receiver (bottom) - NOT BOTH. This image is useful as a model ONLY.

| NOT checked | The phase shift occurs at each data point and the reference receiver measurement sees the same phase shift as the DUT. The response is measured as though a true phase sweep is stimulating the differential input to the DUT. |
| Checked     | The phase shift occurs at each data point, but the reference receiver measurement does NOT see the shift. Use this setting to compensate for a component or fixture that...

5940
shift occurs at each data point, but the reference receiver measurement does NOT see the shift.

Use this setting to compensate for a component or fixture that may present a phase shift before or after the DUT.

**Note:** Beginning with firmware version A.14.00, the **Enable Differential Source Only Mode** feature has been removed.

**One use for Phase Sweep**

Do the following to find the best operating point:

1. Phase sweep port 3, from 0° to 180°. Note the peak of Sdd21. This finds the phase offset of the input differential port.

2. Set port 3 **fixed offset** to the phase that has the highest Sdd21.

3. Then, phase sweep port 4 (balanced port 2) to find the phase offset of the output differential port.

4. Then set the port 4 to fixed phase offset found.

This measures the best possible case Sdd21 for the DUT.

**Power and Attenuators Settings**

Set stimulus power levels using the standard VNA Power and Attenuators dialog.

To set power, press **Power > Main > Power and Attenuators**...
Power and Attenuation (with True Mode Stimulus) dialog box help

When a True Mode Stimulus is selected in the Balanced DUT Topology dialog, the balanced ports are shown on the top rows of the dialog. The individual VNA port settings are displayed but can not be changed directly.

**Port Powers Coupled** Check to couple all power settings for Balanced Port 1 and Balanced Port 2. Clear this box to make independent power settings for these logical ports.

**State** Should be left in Auto for iTMSA measurements. Learn more about this setting.

**Port Power** Set power for the balanced port. Power for VNA ports, shown below balanced ports, are set to 3 dB less, or half power. Power Offset is made in addition to this setting, and on only VNA port 3 and port 4. Offset Power is NOT reflected on this dialog nor on the X-Axis.

**Start and Stop Power** Available when sweep type is set to power sweep.

**Auto Range** Check to allow the VNA to select the optimum attenuation value to achieve the specified test port power. Clear to manually set the attenuation for each port. Type or select the attenuation value in the adjacent Attenuator Control box. When the attenuator setting of a logical port is changed, then the attenuators of the VNA ports associated with that logical port change to the same value.

**Leveling Mode** Open Loop leveling is available only on ports 1 and 3. Learn more.

**Note:** The range of leveled power (ALC range) for balanced ports is 3 dB higher than it is for each individual VNA port. For example, if a VNA source port with 0 dB attenuation will supply leveled power from -30dBm to +15dBm, then the balanced logical port has a range of -27dBm to +18dBm. Learn more about Leveled Power.

---

Calibration with iTMSA

**Note:** Uncalibrated True Mode Stimulus are NOT at all accurate.

Perform or recall a SMART Cal exactly like any other 3 or 4-port cal.

Press **Cal > Main > Other Cals > Smart Cal...**

- All Cals are performed as single-ended.
- Supports all Fixturing and Port Extension features.
- Supports Guided Power Cal.
- Supports Enhanced Response Cal and Source Power Cal.
Source Power Cal

Perform a Source Power Cal on:

- Port 3 for BAL Port 1
- Port 4 for BAL Port 2

Port 1 and Port 2 power levels are adjusted to these ports in the initial sweep.

Receiver Measurements

Any pair of receivers can be viewed as a ratio using the following dialog.

To select these measurements, press Meas > Receivers.

Create / Change Receiver Measurements (with True Mode Stimulus) dialog box help

Click Activate

- For RATIOED measurements, Select a receiver for the Numerator and select a receiver for the Denominator.
- For UNRATIOED measurements, Select a test port receiver (A, B, C, or D) for the Numerator. Reference receiver measurements are NOT accurate. Select 1 for the Denominator.

For example, with a Bal-Bal topology, the above selections show a R3/R1 measurement. R (reference) receivers measure the stimulus to the DUT. An R3/ R1 trace shows the difference between the two sources that comprise logical port 1. With Log format, a power offset between the two sources is visible. With Phase format, a phase offset is visible.

Source Port  Specifies whether a Differential stimulus or Common mode stimulus is used for the measurement.

- With Source Port 1 or 2 selected, then Differential stimulus is used for the unratioed measurement which causes 180° offset between the sources.
  - Source Port 1 = Differential stimulus on BAL1 port
  - Source Port 2 = Differential stimulus on BAL2 port
With Source Port 3 or 4 selected, then **Common** mode stimulus is used for the unratioed measurement which causes **0° offset** between the sources.

- Source Port 3 = Common-mode stimulus on BAL1 port
- Source Port 4 = Common-mode stimulus on BAL2 port

### Differential Input and Output Power Measurement

Equation Editor is used to calculate and display the Differential Input and Output power levels.

This procedure assumes a Bal-Bal topology which requires the following port assignments:

- Balanced Port 1 = VNA ports 1 and 3
- Balanced Port 2 = VNA ports 2 and 4

### Procedure

1. **Preset** the VNA
2. Perform a **4-port** Guided Power Calibration if desired.
3. Then press **Trace**, select **Trace Setup**, then select **Receivers**.
4. Complete the dialog as follows, then click OK.

<table>
<thead>
<tr>
<th>S-Parameter</th>
<th>Balanced</th>
<th>Receivers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activate:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5. The above FIVE traces and the default (S11) trace should be visible on the VNA screen.
6. Click **Tr1** (S11). This is the 'dummy' trace that will become an equation trace.
7. Then press **Math > Analysis > Equation Editor**.

8. In the **Equation** field, enter `(Tr2-Tr4) / sqrt(2)`

9. Press **OK** to close Equation Editor. The **Tr1** trace is now showing Differential Output power.

10. For Differential Input power, activate **Tr3** (C receiver).

11. In Equation Editor, enter `(Tr5-Tr6) / sqrt(2)`. Press **OK**.

12. The **Tr3** trace is now showing Differential Output power.

*The traces used to calculate and display Differential Input and Output Power*
The S93070xB Modulation Distortion application characterizes the nonlinear distortion of RF microwave amplifiers and converters under a modulated signal.

In this topic:

- Features and Requirements
- Hardware Setup
- Compact Modulated Waveform
- Measuring Spectrum Using PNA-X Receivers
- Spectrum Decomposition
- Computing ACPR and EVM
- Optimizing Modulation Distortion Measurements
- Modulation Distortion Graphical User Interface

See Also

- Starting and Exiting Modulation Distortion
- Configuring Distortion Measurements
- Creating Modulation Files
- Displaying Distortion Parameters
- Calibration
- Setup Examples
- Modulation Distortion Measurement
- Programming Examples:
  
  - Measurement Setup
  
  - Measurement Setup Converters
  
  - Create Modulation File
  
  - Display Data Setup
  
  - Source Modulation Calibration
  
  - Spectrum Analyzer Settings
• Unprecedented dynamic range through coherent averaging (repetitive waveforms are used)
• Corrects for port match and establishes calibration reference plane (VNA vector error correction is applied)
• No need to demodulate the response signal
• Accurate measurement of modulated signal with VNA receivers
• Analysis of nonlinear behavior of the device under modulated signal stimulus
• Measure input and output signals coherently
• Delivers figure of merits of the device commonly used in the industry (ACP, EVM, NPR)

Requirements

• Modulation Distortion Option S93070xB
• 2-Port or 4-Port PNA/PNA-X with option 22x or 42x

• Supported external sources:
  • E8267D PSG Vector Signal Generator
  • M8190A with E8267D PSG Vector Signal Generator
  • M9383A/B, M9384B (wideband vector, 44 GHz, with enhanced phase noise)

  **Note:** When setting up an M9383A, M9383B, or M9384B source in the External Device Configuration dialog, select `MXG_Vector` as the driver (they are code compatible).

  • N5182B MXG RF Vector Signal Generator
  • N5192A and N5194A UXG Vector Adapter

• Option S93083A/B provides Frequency Converter Application (FCA) for converter measurements (includes Scalar Mixer (SMC) and Vector Mixer (VMC) measurements).

• Option S9x084A/B provides the ability to measure frequency converters that have an embedded LO.

• Windows 7 or Windows 10 operating system
A Vector Signal Generator is used to generate a repetitive signal with a given CCDF (Complementary-Cumulative-Distribution-Function) and PSD (Power Spectral Density).

**Compact Modulated Waveform**

The modulation distortion application uses a short waveform period to perform an accurate measurement within a relatively short time frame. This process is known as compacting the waveform.

For example, use this process if you want to determine the response of the DUT using a waveform under a specific modulation scheme. The VNA firmware helps to create a slice of the parent waveform. The waveform inherits the frequency signature and statistical characteristics. The sliced waveform is called the compact test signal.

The parent waveform is created using the Keysight Signal Studio Software and is a .wfm file. (The parent waveform is also referred to as the original waveform.) In addition, you can use a .csv file format that has a timestamp, I, and Q that uses a comma to separate the values.
Note: Encrypted .wfm files created using the N5182B MXG RF Vector Signal Generator are not supported.

The following parent waveform is a 5G NR 100 MHz bandwidth signal:

The PNA-X firmware uses a unique algorithm to determine the most statistically representative slice from the parent signal. The results are from the parameters selected in the Create Modulation dialog. The firmware then applies a brick wall filter to remove spectral leakage when the compact test signal plays.

The following shows the CDF of a parent signal and subset slice:

Depending on the parameters selected in the Create Modulation dialog, the compact test signal
displays different characteristics, which can affect the measurement result.

For example, if you are using a 5G NR 100 MHz bandwidth waveform as a parent waveform to create a compact test signal, the plots below illustrate two different compact test signal characteristics when using different parameters for the same parent waveform.

The yellow trace indicates the parent waveform, and the blue trace shows the compact test signal. The plots on the left display a compact test signal consisting of 1,001 tones with a 100 kHz tone spacing. The plots on the right display a compact test signal consisting of 10,001 tones with a 10 kHz tone spacing.

The following plots represent the spectrum of the waveform. The results show what the waveform looks like in the frequency domain by applying the Fast Fourier Transform (FFT) to each waveform. The frequency signature of signals in both the left and right plots have the same bandwidth as the original signal. The original signal has a higher out-of-band spectrum. The compact test signal has a low out-of-band spectrum that uses the brick wall filter for cleaning the signal.

CTS #1: 1,001 Tones  
CTS #2: 10,001 Tones

The following plots represent the position of a compact test signal within the parent signal in the time domain. The numbers shown represent the waveform length (reciprocal of the tone spacing). The left compact test signal has a waveform length of 10 us; the waveform length of the right compact test signal is 100 us. A finer tone spacing waveform results in an extended period of the compact test signal waveform.

CTS #1: 1,001 Tones  
CTS #2: 10,001 Tones
The following plots represent the complementary cumulative distribution function (CCDF) curve of the parent signal, compact test signal, as well as Gaussian distribution (pink trace). The CCDF of this specific parent waveform is in alignment with the Gaussian distribution. The CCDF of a compact test signal that consists of 10,001 tones aligns with the parent waveform across the entire probability. The CCDF of a compact test signal that includes 1,001 tones aligns with the parent waveform, but only until it is approximately a 0.1% probability.

<table>
<thead>
<tr>
<th>CTS #1: 1,001 Tones</th>
<th>CTS #2: 10,001 Tones</th>
</tr>
</thead>
</table>

**Measuring Spectrum Using VNA Receivers**

The following is a simplified block diagram of a modulation distortion setup showing the VNA receivers and VSG.
The VSG replays the compact test signal without any interruptions. Three VNA receivers capture the spectrum at the reference plane at the input and output of the DUT. The instantaneous bandwidth of the VNA ADC (Analog-to-Digital Converter) is at 30 MHz. When the modulation distortion application measures the spectrum of the signal with a bandwidth wider than 30 MHz, it moves the local frequency of the VNA to measure the spectrum for each instantaneous bandwidth. It then combines the captured partial spectrum to obtain a complete spectrum response.

When the modulation distortion application measures the spectrum for each section, it uses multiple receivers coherently and applies linear calibration terms. The modulation distortion application offers a measuring technique where the VNA completes accurate vector corrected measurements at the reference plane.

The block diagram above shows how the VNA measures the input signal and the output signal spectrum:

- Signal generator generates a repetitive compact test signal.
- VNA receiver measures the input signal spectrum and output signal spectrum in the frequency domain.
- Output signal spectrum has spectral regrowth created by the nonlinear response of the DUT.
Spectrum Decomposition

The DUT is stimulated with a low-level signal to measure its linear parameters and a high-level modulated signal to measure its distortion characteristics. The modulation distortion application processes the data and compares the input spectrum $X(f)$ and the output spectrum $Y(f)$ using a process known as spectral correlation. As a result, the modulation distortion application decomposes the output signal spectrum into two parts: $H(f) \cdot X(f)$ and $D(f)$. $H(f) \cdot X(f)$ linearly correlates to the input while $D(f)$ represents the distortion which does not correlate to the input.

Computing ACPR and EVM

Computing ACPR is similar to the traditional signal generator and signal analyzer approach. The channel power of the in-band channel of interest and the channel power of the adjacent channel band is evaluated. The ratio between the BAND and AC is then computed.

For the EVM computation, the compact test signal plays continuously in the signal generator following the measurement of the input and output response of the DUT in the frequency domain. Spectrum correlation is then performed to compute the EVM.

When viewed in the frequency domain, this signal is a mult-tone "grid" of frequencies. The data is measured on this multi-tone grid.
Optimizing Modulation Distortion Measurements

Signal-to-noise ratio of the spectrum measurement

Poor EVM occurs when the random noise is the dominant factor of the EVM, where it likely happens at the left side of the bathtub curve. Improving the SNR of the measurement can improve the accuracy of the measurement.

Noise bandwidth

The resolution bandwidth and the number of coherent averaging of the measurement determine the noise bandwidth. The default value of the noise bandwidth is 1 kHz. Noise bandwidth determines the signal-to-noise ratio (SNR) of the measurement system and measurement time.

As noise bandwidth decreases, modulation distortion increases the underlying coherent average. The result is a wider signal noise ratio and longer measurement time. The resolution bandwidth is set automatically with firmware from the compact test signal waveform length; it is a discrete number. It is the closest value to the one you entered.

Signal noise ratio of the vector signal generator

The SNR of the VSG differs depending on multiple factors. A key factor is the number of tones of the compact test signal. As you increase the number of tones, the SNR of the VSG degrades. For example, compare a compact test signal with 1,000 tones and a compact test signal with 10,000 tones given the same channel power: -10dBm.

The tone power level of the compact test signal with 1,000 tones is -40 dBm, while the tone power of the compact test signal with 10,000 tones is -50 dBm. If the measurement has the same noise floor for each compact test signal, the compact test signal with 1,000 tones is 10 dB better than the compact test signal with 10,000 tones.

Nonlinearity of the test receiver
If there is a nonlinear response in the test receiver, it is unable to distinguish nonlinearity that comes either from the DUT or the receiver of the test system. The PNA-X test receiver needs to be in linear when measuring the signal. When it measures the subtle nonlinearity of a DUT, such as the EVM level of 1%, the recommendation is to keep the power level less than -5 dBm at the test port of the PNA-X. Use the receiver attenuator to adjust the power level if the test port power is higher than -5 dBm.

**Nonideal compact test signal**

The linear error due to the test system can be corrected to have the desired compact test signal at the reference plane. You can do this by adjusting the channel power and linear flatness response using the modulated correction feature. Also, out-of-band spectral regrowth is suppressible by adjusting the ACPR.

**Compact test signal statistical characteristics**

It is critical to understand that nonlinear characteristics of the DUT under modulated signal condition is highly dependent on the stimulus signal. It is also important to create the compact test signal that can stimulate the DUT with the most representative statistical characteristics to the practical usage of the DUT.

The best practice is to align the complementary cumulative distribution function (CCDF) of the compact test signal with the parent waveform. However, the CCDF is not the same since the compact test signal is a slice of the parent waveform. The recommendation is to match the CCDF until it is 0.1% of probability. By choosing more than 3,000 tones gives you a good match of the CCDF - up to 0.1%.

**Measurement throughput**

Faster measurement time gives faster throughput to complete the evaluation. You can determine measurement time using these parameters:

- signal analyzer span
- noise bandwidth
- compact test signal number of tones

Measurement time and measurement accuracy is generally a trade-off. It is essential to have the right balance of speed and accuracy to modify the parameter depending on the target measurement value.

**Modulation Distortion Graphical User Interface**
The GUI consists of setup dialogs accessed by clicking on their corresponding tabs. In this way, configurations can be set up quickly. See Configuring Distortion Measurements for information about these dialogs.
The Create... button in the Modulate tab accesses the Create Modulation dialog used to set up the compact test signal. See Creating Modulation Files for information about this dialog.
**Create Modulation**

**Modulation Type** Compact  
**Source Name** None

**Original Signal**
- **Filename**: VR_256QAM_120kHz_SCS_100MHz_122x88MHz5R.wfm
- **Sample Rate**: 122.880 MHz  
- **Signal Span**: 65.0768 MHz  
- **Number of Samples**: 1228800  
- **Carrier Offset**: -75.450 kHz  
- **Waveform Period**: 10.000 ms

**Compact Signal**
- **Signal Span**: 05.076800000 MHz
- **Waveform Period**: 10.51818 usec
- **Number of Tones**: 1001
- **Peak-to-Avg**: 11.487 dB
- **Carrier Offset**: -75.450 kHz
- **DAC Scaling**: 85.00 %
- **Signal Start Time**: 0 psec
- **Number of Files**: 5

**Priority**
- Desired  
- Calculated

**Optimize Signal**
- **Enable Optimizer**
- **Frequency Tolerance**: 1.00 %

**Calculated Result**

The calculation is complete, ready to save.
Starting and Exiting Modulation Distortion

In this topic:

- Starting Modulation Distortion Application
- Exiting Modulation Distortion Application

Starting Modulation Distortion Application

How to start Modulation Distortion

**Using Hardkey/SoftTab/Softkey**

1. On the VNA, press `Meas > S-Param > Meas Class.....`
2. Select Modulation Distortion, then click OK.
3. In the Confirm Measurement Class Change dialog, click OK to proceed or Cancel to exit.
4. The Modulation Distortion application is displayed (shown below).

**Using a mouse**

1. Click Instrument.
2. Select Meas Class....
3. Select Modulation Distortion, then click OK.
4. In the Confirm Measurement Class Change dialog, click OK to proceed or Cancel to exit.
5. The Modulation Distortion application is displayed (shown below).
Exiting Modulation Distortion Application

1. Select a different Measurement Class for the currently active channel.
Configuring Distortion Measurements

The Modulation Distortion dialog settings are contained within four tabs: **Sweep**, **RF Path**, **Modulate**, and **Measure**. A fifth tab called **Mixer** contains settings for converter measurements (Option S93083A/B FCA required).

For convenience, the Spectrum Analyzer tabs can be displayed as follows:

1. Right-click in a Modulation Distortion dialog.
2. Select **Show Expert Tabs (on|OFF)**.
3. The tabs are displayed next to the Modulation Distortion tabs:

   ![Modulation Distortion Converter Setup](image)

4. To display the Spectrum Analyzer Advanced tabs, right-click again in a Modulation Distortion dialog then select **Show SA Advanced (on|OFF)**:
5. The Spectrum Analyzer Advanced tabs are displayed:

In this topic:

- Create a Modulation Distortion Channel
- Opening the Modulation Distortion Dialog
- Create a Modulation Distortion Converter Channel
- Opening the Modulation Distortion Mixer Setup Dialog
- Sweep Tab in Dialog
- RF Path Tab in Dialog
- Mixer Tab in Dialog
- Modulate Tab in Dialog
  - Create Modulation Dialog (separate topic)
LO Feedthu Monitor Dialog

- Measure Tab in Dialog
- Source Modulation Calibration (separate topic)
- X-axis Type (Modulation Distortion/Modulation Distortion Converters only)

See Also

- Displaying Distortion Parameters
- Modulation Distortion Measurement
- Programming Examples:
  - Measurement Setup
  - Measurement Setup Converters
  - Create Modulation File
  - Display Data Setup
  - Source Modulation Calibration
- Spectrum Analyzer Settings

Create a Modulation Distortion Channel

1. On the VNA front panel, press Meas > S-Param > Meas Class...

2. Select Modulation Distortion, then either:
   - OK delete the existing measurement, or
   - New Channel to create the measurement in a new channel.

3. The Modulation Distortion dialog is displayed.

Opening the Modulation Distortion Dialog

Accessing Modulation Distortion Settings

Using Hardkey / SoftTab / Softkey

1. Press Setup > Main > MOD Setup...

Using a mouse

1. Click Stimulus.

2. Select MOD Setup...
Create a Modulation Distortion Converter Channel

1. On the VNA front panel, press **Meas > S-Param > Meas Class...**

2. Select **Modulation Distortion Converters**, then either:
   - **OK** to delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.

3. The **Modulation Distortion Mixer Setup** dialog is displayed.

Opening the Modulation Distortion Mixer Setup Dialog

<table>
<thead>
<tr>
<th>Accessing Modulation Distortion Mixer Settings</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Using Hardkey /SoftTab /Softkey</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press <strong>Setup &gt; Main &gt; MODX Setup...</strong></td>
<td>1. Click <strong>Stimulus</strong></td>
</tr>
<tr>
<td></td>
<td>2. Select <strong>MODX Setup...</strong></td>
</tr>
</tbody>
</table>

**Sweep Tab in dialog help**

Fixed and Power Sweep Type Dialogs for Modulation Distortion Measurement Class (MOD)
Fixed and Power Sweep Type Dialogs for Modulation Distortion Converters Measurement Class (MODX)
Sweep Type

**Fixed** - Fixed sweep measures a modulated signal with a fixed carrier LO frequency and power level.

**Power** - Power sweep sweeps the total power of the modulated signal at multiple power levels defined by a start/stop power level. This power may be defined at the input or output of the
DUT, depending on the leveling defined in the RF Path tab. The current start/stop power settings are displayed next to the Power Sweep button.

**Carrier Frequency** - (Modulation Distortion only) Sets the carrier LO frequency.

**Carrier Freq In** - (Modulation Distortion Converters only) Sets the carrier frequency at the input of the mixer.

**SA Center/SA Start** - (Modulation Distortion only) Sets the Spectrum Analyzer display center or start frequency.

**SA Center In/SA Start In** - (Modulation Distortion Converters only) Displays the Spectrum Analyzer center or start frequency at the input of the mixer.

**SA Span/SA Stop** - (Modulation Distortion only) Sets the Spectrum Analyzer display span or stop frequency.

**Note:** The SA displayed frequency range is limited to the measurement band frequencies

**SA Span In** - (Modulation Distortion Converters only) Sets the Spectrum Analyzer span or stop frequency at the input of the mixer.

**SA Stop In** - (Modulation Distortion Converters only) Displays the Spectrum Analyzer span or stop frequency at the input of the mixer.

**Noise BW** - Noise BW is equal to the Resolution BW divided by the Vector Average factor. If a Power Sweep Type of List is selected and Custom Noise BW values are set, the word Swept is displayed in the **Noise BW** box. Learn more.

**Set Power At** - Sets power level used for the distortion test at either the input or output of the DUT.

**Buttons**

**Power Sweep...** button - Accesses the **Power Sweep** dialog.

**Power Sweep Type - Ramp**

**Ramp** - Selects a ramp type power sweep and displays the following information:
Start Power At - Sets the start power value for the power sweep.

Stop Power At - Sets the stop power value for the power sweep.

Number of Powers - Sets the number of power points to measure.

Noise BW - This is the same value as shown in the Sweep tab dialog.

Auto-Increase NBW at High Powers - The Noise BW setting will be used for the minimum power level. As the power level increases, the Noise BW will increase automatically. This results in faster measurements and ensures that the noise error is approximately the same for each power level.

Power Sweep Type - List

List - Selects a list of power values to define a power sweep and displays the following information:
Index - Each index entry defines a power level, source and receiver attenuation, and noise bandwidth for the power sweep.

Power - Sets the power level for the specific index number.

Src Atten -

- Fixed Atten - Setting one attenuation value in the column sets them all to the same value.

- Custom - Settings for each index may be different. Note that the Src Atten entry on the RF Path tab will indicate Swept under this condition.

Rcvr Atten -

- Fixed Atten - Setting one attenuation value in the column sets them all to the same value.

- Custom - Settings for each index may be different. Note that the Rcvr Atten entry on the RF Path tab will indicate Swept under this condition.

Noise BW - This is the same value as shown in the Sweep tab dialog if the Noise BW column is set to One NBW or Auto-Increase.

- Fixed NBW - Setting one noise bandwidth value in the column sets them all to the same value.

- Custom - Settings for each index may be different. Note that the Noise BW entry on the Sweep tab will indicate Swept under this condition and will be grayed-out.
Auto-Increase - The Noise BW setting will be used for the minimum power level. As the power level increases, the Noise BW will increase automatically. This results in faster measurements and ensures that the noise error is approximately the same for each power level.

Add Row button - Adds a row after the currently selected row in the table.

Delete Row button - Deletes the currently selected row.

Load... button - Loads a .csv file into the table.

Save... button - Saves the table to a .csv file.

DC Sources... button - Accesses the standard dialog for controlling power supplies. Learn more.

Sweep Details... button - Accesses the following dialog:

**Dialog for Modulation Distortion Measurement Class (MOD)**

![Sweep Details dialog](image)

**Dialog for Modulation Distortion Converters Measurement Class (MODX)**
Force RF Power OFF at the End of Sweep - System Preference - If enabled, RF power is turned off at the end of a sweep.

Note: This setting is a system-wide preference; the state of this setting will affect all measurement channels and the state will not change if the VNA is preset or restarted.

Delay Before Start of Sweep - Same as Sweep Delay in a standard channel.

Delay Before Distortion Measurement - Adds delay after the linear S-parameter sweep and before the distortion measurement to allow the RF source to settle.

Linear S-Parameter Measurement Sweep (Modulation Distortion only)

Enable S-Param Sweep - Enables S-parameters to be measured at the low linear power level. If disabled (default), curve fitting (smoothing) is used instead of the S-parameter sweep. Curve fitting can be adjusted by varying the window aperture using the Distortion Measurement Correlation Aperture setting in the Measurement Details dialog.

Note: When disabled, some measurements, such as S-parameters, will display -200 dB because they will not be measured.

S-Param Source - Selects source and signal used to measure S-parameters.
**External Vector - Chirp** - Selects a chirp signal from the external signal generator for measuring S-parameters.

**S-Param Power** - Sets the power level used at the DUT input to measure the DUT gain when operating in its linear region. The linear gain is used to calculate distortion.

**S-Param Freq Step** - Frequency step size setting used when measuring S-parameters.

**S-Param IFBW** - Bandwidth setting when measuring S-parameters.

**Re-use Previous S-Parameter Measurements If Available** - If disabled, S-parameters will be measured at the low linear power level. If enabled and compatible measurements exist from a previous sweep, those measurements will be used and a Linear S-parameter sweep will not be performed. If enabled and compatible measurements are not available, then a Linear S-parameter sweep will be performed.

**Apply** button - Applies any changes to the settings in this dialog.
Dialog for Modulation Distortion Converters Measurement Class (MODX)

Set Power At DUT In - Displays the DUT power set in the Sweep tab.

VNA Source Attenuator - Sets the VNA's internal source attenuator (PXI/USB VNA does not support this)

- **Include** - When selected (default), the external modulated source passes through the attenuator of the specified PNA source and the attenuator setting is used to calculate the power levels. If a Power Sweep Type of List is selected and Custom Src Atten values are set, the word Swept is displayed in the VNA Source Attenuator box.

Nominal Source Amp - Sets the nominal gain (positive number) from an amplifier or loss (negative number) due to an attenuation, cable loss, etc. This value is used by the Set Power At function, receiver leveling, and calibration. This value is equal to the Power Offset setting found on the Offsets and Limits dialog.

DUT Input - Sets the VNA port number that is connected to the DUT input.

Nominal DUT Gain - Sets the nominal DUT gain. This value is used by Set Power At DUT Out, receiver leveling, and calibration. This function is also displayed on the calibration dialog.
**DUT Output** - Sets the VNA port number that is connected to the DUT output.

**Receiver Attenuator** - Sets the VNA's internal receiver attenuator for the currently selected output port.

**Buttons**

**RF Path Config...** button - Accesses the RF Path Configuration dialog.

**Offsets and Limits...** button - Accesses the Offsets and Limits dialog.

**Apply** button - Applies any changes to the settings in this dialog.

**Mixer Tab in dialog help**

**Converter Stages** - Selects between 1- or 2-stage mixer configuration.

**Enable Embedded LO** - Check to enable measurements of mixers that have a fixed LO inside the DUT.

**Note:** Embedded LO does not support signals with a tone spacing of less than 1 kHz.

**Setup...** - Opens the Embedded LO dialog.
Enable Embedded LO - Check to enable measurements of mixers that have a fixed LO inside the DUT.

Tuning Method - These settings determine the amount of time spent versus the degree of accuracy to which the LO Frequency is measured. Accuracy is compromised when noise starts to appear on the measurement trace.

- **Broadband and Precise** Does the entire tuning process for each background sweep.

- **Precise Only** Does NOT perform broadband tuning on each sweep. Use this setting when the embedded LO is stable. The signal (after broadband) must be within ½ the tuning IFBW. If the signal will always be within ½ the IFBW, broadband tuning is not needed. Most satellite components are within 3 kHz absolute so might not need broadband tuning.

- **Disable Tuning** Only the previously measured LO Frequency Delta is applied
to the reference mixer LO and VNA receivers.

**Tune every** - Set the interval at which tuning is performed before a measurement sweep. 'Tune every 3 sweeps' means that every third measurement sweep is preceded by tuning sweeps. If the embedded LO drifts, or if regularly changing DUTs, use 'Tune every 1 sweep'.

**Broadband Search** - Set the frequency span over which to measure the embedded LO frequency.

**Noise BW** - Noise Bandwidth used for Broadband and Precise tuning sweeps. This sets the resolution in the Broadband sweeps.

**Max Iterations** - The maximum number of Precise sweeps to make. When this number is reached, the final measurement is used.

**Tolerance** - When two consecutive Precise measurements are made within this value, the final measurement is used. If this is not achieved within the Max Iterations value, then the last measurement is used. This is the best of the 'Tunings settings' to change to improve accuracy.

**LO Frequency Delta** - The absolute difference between the measured embedded LO frequency and the LO setting that is entered in the Mixer Tab dialog.

**Find Now** - The VNA finds and measures the actual LO frequency using the current dialog settings. This data is displayed in the **Status** box.

**Default** - Resets the LO Frequency Delta and Tuning parameters to their default settings.

**Input** - Input frequency defined on the Sweep tab.

**IF** - Intermediate frequency between stage 1 and stage 2 for a 2-stage mixer configuration calculated from the mixer equation selection.

Choose from the following mixer equations:

- IF=Input+LO1
- IF=Input-LO1
- IF=LO1-Input

**Output** - Mixer output frequency calculated from the mixer equation selection.

Choose from the following mixer equations for a 1-stage mixer configuration:
Choose from the following mixer equations for a 2-stage mixer configuration:

- Output=IF+LO2
- Output=IF-LO2
- Output=LO2-IF

X - The combination of numerator / denominator forms a fractional value that is multiplied by the input and LO frequency ranges. Learn more.

**LO/LO1/LO2** - Sets the LO frequency.

**Source Name** - Select Not Controlled to allow an external source to provide a Fixed LO Frequency at all times. Otherwise, select an internal VNA source or External source to be used as the LO. Learn how to Configure an External Device (Source).

**Power** - Sets the LO power.

**Leveling**

- **Open Loop** - No ALC or receiver leveling. No leveling is used in setting the power.
- **Internal** - ALC leveling. Power level within an attenuator setting is limited to the ALC range.

**Attenuator** - Selects the receiver attenuation.

**Save... button** - Saves the mixer configuration to a .mrx or .mxrx file.

**Load... button** - Loads an existing mixer configuration file.

**Apply** button - Applies any changes to the settings in this dialog.
**Dialog for Modulation Distortion Measurement Class (MOD)**

Source - Selects sources that have been defined in the External Device Configuration dialog or selects Source3 (requires Option S93072B, Option XSB, Option S93070xB or Sx090A/B, and either Option 422 or Option 423).

**Dialog for Modulation Distortion Converters Measurement Class (MODX)**
Add Source... - Adds a new external source using the External Device Configuration dialog.

Source3 - With Option S93072B Arbitrary Waveform Generation on XSB Port (requires Option S93072B, Option XSB, Option S93070xB or Sx090A/B, and either Option 422 or Option 423), an internal third RF source can be selected from the SRC3 connector on the rear panel. Refer to Set Up Source3 Modulation Source - Option S93072B for important setup information.

**Note:** If the source selection is changed, and if a modulation file is selected and Enable Modulation is checked, then the modulation file will be automatically loaded into the source.

Modulation File - Displays the currently loaded modulation file. Supported file types include *.mdx, *.csv, and *.wfm. After selecting a modulation file, the following dialog is displayed to allow you to Autofill the measurement bands:

![Modulation Properties](image)

**Note:** Once a source is selected and a file is selected, the file will be automatically loaded into the source and the Enable Modulation box is checked.

Enable Modulation - Check to enable modulation.

Enable Source Correction - Check to enable source correction after a source modulation calibration is performed.

Enable LO Feedthru Monitor - Check to enable LO Feedthru monitor.

Enable Pulse - Check to enable standard pulse modulation.

Buttons

Load File... button - Loads an existing modulation file. Supported file types include *.mdx, *.csv, and *.wfm.

Create... button - Set up a modulation file. See Create Modulation dialog description.
Edit... button - Edit the currently loaded modulation file. See Create Modulation dialog description.

Properties... button

**Waveform Tab**

Displays the properties of the currently active modulation file, which was selected in the Modulation settings dialog. This information cannot be edited.

**Calibration Tab**

The Calibration tab displays the calibration files and their properties stored in the .mdx file. Each calibration displayed in the list is for one power level. Calibrations may have been performed on multiple power levels during a single calibration. In this case, multiple calibrations will be saved in the .mdx file. Any of these calibrations can be deleted by selecting the calibration name then clicking on the **Delete Cal** button.

Source Cal... button - Accesses the Modulation Cal - Setup dialog for performing source modulation calibration.

LO Monitor... button - See **LO Feedthru Monitor** dialog help below.

Pulse Setup... button - Opens the Pulse Setup dialog to set up a pulse measurement.
When the **OK** button is clicked, the following warning is displayed if a source has not been added or the Modulation file name has not been specified:

![Warning dialog]

Click **Exit with Errors** to exit the setup dialogs with the errors or click **Go Back** to return to the setup dialogs to fix the error.

**Apply** button - Applies any changes to the settings in this dialog.

---

**LO Feedthru Monitor** dialog help

The SA multitone NPR and the Modulation Distortion measurement use a vector signal source. It may have LO feedthrough leakage and the level may drift after calibration due to hardware imperfection. The LO feedthru calibration, as a part of source modulation calibration, reduces the LO feedthru by adjusting I/Q offset voltage. The LO Feedthru Monitor feature notifies you when you should take the calibration again.

You need to setup them depends on your measurement requirements.

- Monitoring receiver: It’ll be a reference receiver in most cases.
- Tolerance: Relative to Band Power (in dBc), Relative to Average Tone Power (in dBc) or Absolute level (dBm).

The firmware checks the LO feedthru level on every measurement. Exceeding the tolerance make a warning message of "LO Feedthru is out of tolerance on channel <n>. Perform Source Modulation Cal > LO Feedthru Calibration".

Supported measurements are Modulation Distortion measurements (MOD, MODX) and SA multitone NPR measurement. Non-multitone SA is not supported.
Enable LO Feedthru Monitor - Check to enable LO Feedthru Monitor. LO feedthru level is checked on every measurement.

Threshold Type -

- Relative to Band Power (in dBc): Compare with the band power.
- Relative to Average Tone Power (in dBc): Compare with the average tone power.
- Absolute level (dBm) - Check the LO feedthru absolute power level.

Threshold Level - Set the threshold level.

Measure LO at - Select the port to monitor the LO feedthru signal. Select the reference receiver in most cases.
Measurement Type - Selects the measurement type:

   **ACP** - Performs ACP measurement.

   **ACP+EVM** - Performs both ACP and EVM measurements.

   **Band Power** - Measures total power within a specified frequency span.

   **EVM** - Performs EVM measurement.

   **NPR** - Performs NPR measurement.

   **Multi-Band** - Displays the full Measurement Band Table.
Edit Table

Append Band - Adds a new band after the row with a currently selected active cell. Each band is assigned to a measurement type. Distortion parameters are calculated over the measurement bands.

Delete Band - Deletes the row with a currently selected active cell.

Edit Band Name - Change the Band Name.

Autofill Current Table from Mod File - Automatically sets up appropriate bands from the currently active modulation file loaded in the source.

Create New Table from Mod File - Automatically deletes the current table, then sets up appropriate bands for making measurements with information from the currently active modulation file loaded in the source.

Band Name - Right-click in the Band Name field to display a keyboard for editing the name.
**Carrier Offset Freq** - Offsets the Carrier integration bandwidth relative to the Carrier LO used to generate the modulation signal. Use the up/down arrows or double-click in the field to display the keypad for entering frequency.

**Carrier Integ BW** - Sets the Carrier integration bandwidth for the distortion measurement. The IBW is used to determine total signal power within a specified frequency span. For example, to calculate the total power of a signal composed of 100 tones spaced 1 MHz apart over a 100 MHz span, the signal power would be integrated over an IBW of 100 MHz.

**Notch Offset Freq** - (NPR measurement only) Sets the notch center relative to the carrier LO used to generate the modulation signal. Use the up/down arrows or double-click in the field to display the keypad for entering frequency.

**Notch Integ BW** - (NPR measurements only) Sets the integration bandwidth of the NPR notch measurement. The IBW is used to determine total signal power within a specified frequency span. For example, to calculate the total power of a notch composed of 100 tones spaced 1 MHz apart over a 100 MHz span, the notch power would be integrated over an IBW of 100 MHz.

**ACPLo Offset Freq** - (ACP measurements only) Offsets the lower ACP integration bandwidth relative to the LO used to generate the modulated signal. Use the up/down arrows or double-click in the field to display the keypad for entering frequency.

**ACPLo Integ BW** - (ACP measurements only) Sets the integration bandwidth of the lower ACP measurement. The IBW is used to determine total signal power within a specified frequency span. For example, to calculate the total power of a signal composed of 100 tones spaced 1 MHz apart over a 100 MHz span, the signal power would be integrated over an IBW of 100 MHz.

**ACPUp Offset Freq** - (ACP measurements only) Offsets the upper ACP integration bandwidth relative to the LO used to generate the modulated signal. Use the up/down arrows or double-click in the field to display the keypad for entering frequency.
**ACPUp Integ BW** - (ACP measurements only) Sets the integration bandwidth of the upper ACP measurement. The IBW is used to determine total signal power within a specified frequency span. For example, to calculate the total power of a signal composed of 100 tones spaced 1 MHz apart over a 100 MHz span, the signal power would be integrated over an IBW of 100 MHz.

**Buttons**

**Autofill** - Automatically fills in the measurement settings for all bands from the currently active modulation file loaded in the source:

<table>
<thead>
<tr>
<th>Modulation File Band(s)</th>
<th>Currently Defined Measurement Band(s)</th>
<th>Autofill Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-band</td>
<td>Single-band</td>
<td>Use single-band Modulation File</td>
</tr>
<tr>
<td>Single-band</td>
<td>Multi-band</td>
<td>All bands set using single-band Modulation File</td>
</tr>
<tr>
<td>Multi-band</td>
<td>Multi-band but with fewer or same number of bands than Modulation File</td>
<td>First band set using first Modulation File band</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Second band set using second Modulation File band, etc.</td>
</tr>
<tr>
<td>Multi-band</td>
<td>Multi-band but with more bands than Modulation File</td>
<td>First band set using first Modulation File band</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Second band set using second Modulation File band, etc.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>When bands exceed number of bands in Modulation File, set them all to last Modulation File band settings</td>
</tr>
</tbody>
</table>

**Measurement Details...** button - Accesses the Measurement Details dialog.
**Distortion Measurement Correlation Aperture** - Sets the frequency span window used for modeling the DUT's gain and distortion.

- **Auto** - Check to automatically set **Distortion Measurement Correlation Aperture** to window size.

**ADC Anti-alias Filter** - Selects between auto and wide IF filter anti-aliasing path.

- **Auto** - Check to automatically set the ADC Filter setting based on the ADC Sampling Frequency. If the currently selected modulation waveform was created with Nyquist Rejection = OFF, then the VNA will measure the signal using the Narrow anti-alias filter in the receiver.

- **Wide** - Selects the ADC 38 MHz IF filter path.

- **Narrow** - Selects the ADC 11 MHz IF filter path. A warning message will appear if the Narrow IF filter path is selected and the Resolution Bandwidth is > 1 MHz.

**Modulation Filter** - (EVM and ACP measurements only) Sets the measurement filter to either **None** (default) or **RRC** (root-raised-cosine filter).

- **Alpha** - Sets Alpha factor of the filter.

- **Symbol Rate** - Sets the Symbol Rate of the filter. If **Auto** is selected, the symbol rate from the file is used. If no Symbol Rate is indicated in the file, then the Symbol Rate will be approximated from the bandwidth of the signal.

**EVM Normalization** - Identifies the scaling factor applied to the EVM measurements. Enter
Nominal DUT NF - Sets the DUT noise figure. This value is used by the EVM measurement. This setting adds noise to the measurement for more realistic results. The default is 0 dB. Setting the noise figure to a low value (-200 dB) will make EVM measurements without the effects of noise.

The following is the formula for computing EVM for a given NF value:

$$DEVM = \frac{k T^{1/2}}{F^{1/2}} \int_{f_0}^{f_1} X(f) \sqrt{f_0 - f_1} df$$

where:

- $DEVM$ : Measured Distortion EVM
- $k$ : Boltzmann constant
- $T$ : 290k
- $F$ : Noise Factor of DUT ($F = 10^{(NF/10)}$)
- $X(f)$ : input spectrum
- $f_0 , f_1$ : given frequency band to be computed

Apply button - Applies any changes to the settings in this dialog.
Accessing X-axis Type Settings

Using Hardkey / SoftTab / Softkey

1. Set up a Modulation Distortion Fixed or Power sweep measurement (see above).

2. Press **Trace** > **Trace 1-7** > **New Traces**.

3. Click on the **Distortion** tab then select a measurement parameter.

4. Press **Sweep** > **Main** > **X-axis Type**.

Using a mouse

1. Set up a Modulation Distortion Fixed or Power sweep measurement (see above).

2. Click **Instrument**.

3. Select **Trace**.

4. Select **New Traces**.

5. Click on the **Distortion** tab then select a measurement parameter.

6. Click **Stimulus**.

7. Select **Sweep**.

8. Select **X-axis Type**.

---

**X-axis Type dialog help**

The X-axis Type dialog is used to set the displayed x-axis annotation for a selected trace in Modulation Distortion or Modulation Distortion Converter measurements.

**Dialog for Modulation Distortion Measurement Class (MOD)**
Dialog for Modulation Distortion Converters Measurement Class (MODX)
Select

**Trace N** - Selects active trace.

**All Traces** - Selects all available traces.

**Sweep Type** - Based on the currently active sweep type for the selected channel.
**Fixed** - Fixed sweep measures a modulated signal with a fixed carrier LO frequency and power level. The Sweep Type selection determines the available X-axis Type. This selection can only be changed from the Sweep dialog.

**Power** - Power sweep sweeps the total power of the modulated signal.

**X-axis Type**

The X-axis Type setting defines the x-axis annotation for the displayed spectrum analyzer data. For Modulation Distortion Converter measurements, the x-axis annotation may be set to one of two frequency ranges: mixer input range and mixer output range.

For example, if a mixer has an input frequency of 1 GHz and an output frequency of 10 GHz, the X-axis Type is set to SA Freq In, and Pln and POut are measured, Pln and POut will both indicate they are measured at 1 GHz, even though POut is actually at 10 GHz. If the X-axis Type is set to SA Freq Out, Pln and POut will both indicate they are measured at 10 GHz, even though Pln is actually at 1 GHz.

For Modulation Distortion Channels (MOD)

**SA Frequency** - SA display showing the SA frequency settings.

**Power In** - Displays input power sweep.

**Power Out** - Displays output power sweep.

**Measured CarrIn1** - Measured input band power.

**Measured CarrOut2** - Measured output band power.

For Modulation Distortion Converters Channels (MODX)

**SA Freq In** - SA display showing mixer input range.

**SA Freq Out** - SA display showing mixer output range.

**Power In** - Displays input power sweep.

**Power Out** - Displays output power sweep.

**Measured CarrIn1** - Measured input band power.

**Measured CarrOut2** - Measured output band power.
### Fixed Parameters (Power Sweep Type only)

**Value** - Displays the spectrum analyzer frequency set in the Sweep dialog for a power sweep.

**Parameter** - (Modulation Distortion Converters Channels only) Selects either **SA Freq In** to display mixer input range or **SA Freq Out** to display mixer output range.
Creating Modulation Files

The **Create Modulation** dialog is used to create modulation files which can then be loaded into the source. The types of modulation files that can be created include **NPR Notch**, **Flat Tones**, and a shortened version of a modulation file called **Compact**.

### Original (Parent) Waveform

**Keysight Signal Studio Waveform (.wfm)**

Use a .wfm file, which is created using the Keysight Signal Studio Software.

In order to use a compact file (.mdx) which is created from .wfm, the corresponding license is required on the Signal Generator to play the .mdx.

When the waveform pack license is used, the original waveform file must be assigned / locked to the signal generator in order to use the .mdx created from the original waveform file.

**Note:** Encrypted .wfm files created using the N5182B MXG RF Vector Signal Generator are not supported.

**IQ file (.csv)**

Use a .csv file format that has a timestamp, I, and Q that uses a comma to separate the values.

No license is required on the signal generator to play the .mdx created from the .csv.

**Modulation Type** - Selects the modulation type (note that these define signals, not measurements):

- **Compact** - Compact signals cut a slice of the IQ data from an original waveform.
- **Flat Tones** - This signal is a set of constant amplitude tones over a defined signal span.
- **NPR Notch** - This signal is a set of constant amplitude tones over a defined signal span where a subset of those tones are set to zero over a notch span.
Compact Modulation Type

Compact signals cut a slice of the IQ data from an original waveform. The slice of IQ data is chosen that best fits in terms of statistical distribution (CCDF), then conditions the waveform so that it has the same spectrum signature as the original signal.

Original signal:

Compact signal:
Source Name - Selects the modulation source. Learn more.

Sample Rate - Sets the source sample rate.

  Auto - Automatically selects a sample rate for the active source depending on the modulation requested. If the modulation is invalid (for example, 0 Hz span), then Auto will default the sample rate to 100 MHz.

Filename - Select the filename of the original (parent) modulation file to be compacted.

  Sample Rate - Displays the sample rate of the original modulation file.

  Number of Samples - Displays the number of samples in the original modulation file.

  Tone Spacing - Displays the tone spacing of the original modulation file.

  Waveform Period - Displays the waveform period of the original modulation file.

  Signal Span - Displays the signal span value of the original modulation file.

  Carrier Offset - Displays the carrier offset value of the original modulation file.

Signal Span - Sets the frequency span of the modulated carrier.

Tone Spacing/Waveform Period - Sets the distance between each tone.

Number of Tones - Sets the desired number of tones. This setting is related to the span and tone spacing: (Number of Tones) = (Signal Span)/(Tone Spacing) +1.

Peak-to-Avg - Displays the peak-to-average value of the modulated signal.

Carrier Offset - Sets the carrier offset value relative to the carrier LO frequency.

Nmbr of Subcarriers - Allows selection of multiple carriers when defining a multicarrier signal. Select None, 2, 3, 4, 5, 6, 7, 8, or 9 from the pull-down menu. The default is None.
**Sub1 Span** - Sets the span of the subcarrier selected from the pull-down menu.

**Sub1 Offset** - Sets the offset of the subcarrier selected from the pull-down menu.

**DAC Scaling** - Sets the scaling factor used for the waveform (full scale = 100%). This ensures that the DAC filter does not output a signal that is larger than the DAC’s maximum output level, which can cause distortion in the system. Setting the scaling factor to 100% will usually cause excessive distortion.

**Signal Start Time** - Sets where to start the compact signal within the original signal. The compact signal is a slice of the original signal.

**Note:** The Priority checkbox must be checked to use this value.

**Number of Files** - Sets the number of modulation files to create. This function is useful to create several signals, compare them, then save the best signal.

**Priority** - Check to attempt to calculate values closer to the desired values.

**Calculated** - Displays the settings that were used to create the modulated signal.

**Enable Optimizer** - When enabled, the calculated modulated signal will be optimized according to the constraints defined in this group box.

If **Enable Optimizer** is disabled:

- Always creates a signal that can be measured by the PNA.

- If the Priority box for **Waveform Period** or **Tone Spacing** is checked, then this will attempt to create the exact desired period if possible.

- If the Priority box for **Number of Tones** is checked, then this will attempt to create the exact desired number of tones if possible.

- In some cases, the resulting file will be too large, or the frequencies will be inconsistent with the ADC frequencies. In these cases, the desired values will not be used.

**Setup...** button - Accesses the **Optimizer Setup** dialog.
Reject up to Harmonic Number - Set the number of test signal harmonics you want to be protected against. This adds constraints to the list of LOs used to cover the span.

Reject Nyquist Frequencies - Ensure that Nyquist images of the signal tones in the IF bandwidth are not falling back on top of real signal frequencies.

Enable Brick-wall Filter For Spectral Leakage - (Compact signals only) The brick-wall filter is applied to the band-power span calculated for the signal. The brick-wall filter cuts off signals outside this span. Uncheck this box to retain signals outside the calculated band power span.

Optimize Signal pull-down menu:

Frequency Tolerance - Set the allowed tolerance for tone spacing when calculating the modulation signal. Wider tolerance results in selection of tone spacing which require smaller files and less measurement time. If the tolerance to too small to provide a solution, then a solution is calculated with the minimum possible tolerance and the tolerance value will be changed.

Min Waveform Period - Minimizes the period of the waveform greater than or equal to the value (seconds).

Min Number of Tones - Minimizes the number of tones greater than or equal to the value. This will ignore the Number of Tones selection.

Max Tone Spacing - Maximizes the tone spacing less than or equal to the value (Hz). This will ignore the Tone Spacing selection.

Display - Select from the following:

Spectrum-Ideal - Displays the spectrum represented by floating point numbers, which results in a very low noise floor.

Spectrum-16bit - Displays the spectrum represented by 16-bit numbers, which results in more distortion.
**Time** - Displays the signal in the time domain.

**CCDF** - Displays the complementary cumulative distribution function.

**CCDF Error** - (Compact Modulation Type only) Displays the difference between the parent signal and created signal. The Y-axis is displayed in linear %.

**File** - Allows you to switch between multiple created signals for comparison, then save the best signal. Use **Number of Files** to specify the number of files to create for comparison.

**Number of Samples** - Displays the calculated file size. If there are no calculated results, **None** is displayed.

**Calculated Sample Rate** - Displays the calculated sample rate. If there are no calculated results, **None** is displayed.

**Measurement Time** - Displays the minimum measurement time for the calculated signal. If there are no calculated results, **None** is displayed.

**Filename** - Displays the name of the modulation file. If the calculated result has not been saved, **None** is displayed.

**Display Data Features**

Click-and-drag over a part of the display to zoom in on data.

![Graph with zoom feature](image-url)
Right-click in the display area to access the following menu:

- **Autoscale** - Automatically scales the data to fit vertically within the display grid area.

- **Display marker annotation** - Select to display marker annotation in the top-right of the display.

- **Show graticule** - Select to display graticules.

- **Add marker to:** - Select to add a marker to the displayed data trace. When a selection is made, the mouse pointer changes to a "+". Click in the display area and the marker will appear. Drag the marker to the desired position. Each time this selection is made, a new marker will be added to the data.

  Choose from:

  - **Original/Compact** - Add marker to the original data and/or the calculated compact
data when the Modulation Type selection is **Compact**.

**Flat Tones** - Add marker to flat tone data when the Modulation Type selection is **Flat Tones**.

**NPR Notch** - Add marker to notch data when the Modulation Type selection is **NPR Notch**.

**Copy to Clipboard** - Copies a bitmap of the trace control (Display) to the clipboard. It can then be pasted into any document that accepts bitmaps.

![Graph](image)

**Print...** - Prints the displayed data.

**Scale properties...** - Accesses the following dialog:
**Stimulus** - Sets the **Begin** and **End** frequency displayed on the X-axis.

**Response** - Sets the Reference level in the center of the Y-axis and sets the scale per division.

### Flat Tones Modulation Type

**Source Name** - See above.

**Sample Rate** - See above.

**Signal Span** - See above.

**Tone Spacing/Waveform Period** - See above.

**Number of Tones** - Select between odd or even number of tones and sets the desired number of tones. This setting is related to the span and tone spacing: \((\text{Number of Tones}) = \frac{\text{Signal Span}}{\text{Tone Spacing}} + 1\).

**Nmbr Tones, Odd** - This forces the optimizer to choose an odd number of tones. The optimizer will also choose an offset such that the carrier lands on one of the tones.
Nmbr Tones, Even - This forces the optimizer to choose an even number of tones. The optimizer will also choose an offset such that the carrier either lands on one of the tones, or exactly halfway between the tones.

Peak-to-Avg - See above.

Carrier Offset - See above.

Phase Type - Select Random, Fixed, or Parabolic.

Random Phase Seed - Sets the phase seed when Random phase is the Phase Type.

DAC Scaling - See above.

Priority - See above.

Calculated - See above.

Enable Optimizer - See above.

Optimize Signal pull-down menu - See above.

Display - See above.

NPR Notch Modulation Type
Source Name - See above.
Sample Rate - See above.
Signal Span - See above.
Tone Spacing/Waveform Period - See above.
Number of Tones - See above.
Peak-to-Avg - See above.
Carrier Offset - See above.
Phase Type - See above.
Random Phase Seed - See above.
Nmbr of Notches - Sets the number of notches in the modulated signal.

Notch Location - Select from the following:

  Symmetric - Locates the notch in the center of the signal span.

  Avoid Carrier - Locates the notch near the center of the signal span but will be shifted to
avoid the LO carrier feedthrough.

**Custom** - Allows the user to define the offset of the notch.

**Notch N Span** - Sets the span of the selected notch. The notch can be up to 10% of the **Signal Span**.

**Notch N Offset** - The offset frequency is the center frequency of the selected notch relative to the LO carrier frequency. Typically, the notch will have a 0 Hz offset, meaning it is centered on the LO carrier. If you have more than one notch, you can offset some of the notches from the carrier. For example, if you have three notches 1 MHz wide, you might set their offsets to -10 MHz, 0 MHz and +10 MHz so that they are spaced across the wideband carrier.

**DAC Scaling** - See above.

**Priority** - See above.

**Calculated** - See above.

**Enable Optimizer** - See above.

**Optimize Signal** pull-down menu - See above.

**Display** - See above.

**Buttons**

**Calculate** button- Calculates the result from the current settings. If calculated results exist then data is plotted on the display. If there are no calculated results then there is no plotted data and **No Data** is displayed. The calculated result is erased if you click on the **Defaults** button or **Recall...** button and recall a previously saved file.

**Save...** button - Saves the current modulation settings.

**Recall...** button - Recalls a previously saved modulation file.

**Edit...** button - Accesses the Edit Multitone dialog:
**Tone** - Tone number from 1 to N. Cannot be edited.

**Frequency** - Frequency of tone in Hz relative to the carrier. Cannot be edited.

**Power (dBm)** - Tone power in dBm. Click in the cell to edit.

**Phase (deg)** - Tone phase in degrees. Click in the cell to edit.

**State** - Click in the cell to turn the tone on or off.

**Set All Tones** - Set all tone states to on or off.

**Display** - Select which tones to display in the table (All Tones, On-Tones, or Off-Tones).

**Go To Row** - Jumps to the specified row. Enter the row number then press Enter.

**Save...** button - Saves the current multitone settings as a .csv file.

**Load...** button - Loads a previously saved multitone .csv file.

**Defaults** button - Restores default modulation settings for the current Modulation Type.
Distortion Measurement Parameters

In this topic:

- Spectrum Decomposition
- Distortion Model
- Accessing the Distortion Measurement Parameters
- Main and Distortion Tab Measurement Parameters
- Distortion Table

Spectrum Decomposition

The DUT is stimulated with a low-level signal to measure its linear parameters and a high-level modulated signal to measure its distortion characteristics. The modulation distortion application processes the data and compares the input spectrum \( X(f) \) and the output spectrum \( Y(f) \) using a process known as spectral correlation. As a result, the modulation distortion application decomposes the output signal spectrum into two parts: \( H(f)X(f) \) and \( D(f) \). \( H(f)X(f) \) linearly correlates to the input while \( D(f) \) represents the distortion which does not correlate to the input.

![Spectrum Decomposition Diagram](image)

Distortion Model

The following Distortion Model describes noise and distortion in an amplifier and will help in understanding the Distortion Measurement Parameters (shown below):

![Distortion Model Diagram](image)
Y(f) = C(f) * S21(f) * X(f) + D(f)

[H(f) = C(f) * S21(f)]

where:

X(f) = Measured input signal

Y(f) = Measured output signal

C(f) = Compression

S21(f) = Linear flatness and gain

D(f) = Output modulation distortion

Accessing the Distortion Measurement Parameters

How to select and configure Measurement Parameters

Using Hardkey / SoftTab / Softkey

1. Select a trace by pressing Trace > Trace N > Trace N.
2. Press Trace > Trace Setup > Measure....
3. Select a parameter.

Using a mouse

1. Right-click on a trace.
2. Select a parameter.

Main and Distortion Tab Measurement Parameters
Select from the parameters to measure power, S-parameters, or distortion characteristics.

**New Trace** - Add a new trace.

**Channel N** - Select a channel number for the new modulation distortion trace.

**Window N** - Select to create the new trace in an existing window or new window.

**Select All** - Select all measurement parameters.

The following tables describe the available Distortion Measurement Parameters for each DUT type and for the **Main** and **Distortion** tabs.

**Note:** The parameter names in the table assume we are testing a DUT with Port 1 = input and Port 2 = output.

**Main Tab Measurement Parameters**
<table>
<thead>
<tr>
<th>DUT Type</th>
<th>Description</th>
<th>Default Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amplifier</td>
<td>Mixer</td>
<td>Description</td>
</tr>
<tr>
<td>PI1</td>
<td>PI1</td>
<td>Input power incident on the DUT</td>
</tr>
<tr>
<td>PO1</td>
<td>PO1</td>
<td>Power reflected from the DUT input</td>
</tr>
<tr>
<td>PO2</td>
<td>PO2</td>
<td>Output Power incident on the VNA test port</td>
</tr>
<tr>
<td>MSig2</td>
<td>MSig2</td>
<td>Output modulation signal - C(f) * S21(f) * X(f)</td>
</tr>
<tr>
<td>MDist2</td>
<td>MDist2</td>
<td>Output modulation distortion - D(f)</td>
</tr>
<tr>
<td>MGain21</td>
<td>MGain21</td>
<td>DUT modulation gain - C(f) * S21(f) = H(f)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This gain is a smooth complex function, with smooth referring to the mathematical property that it has continuous derivatives of infinite order</td>
</tr>
<tr>
<td>MComp21</td>
<td>DUT modulation compression (not calculated if there is no S21) - C(f)</td>
<td>dB</td>
</tr>
<tr>
<td>PGain21</td>
<td>PGain21</td>
<td>Power gain - Y(f) / X(f)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This gain is calculated tone-by-tone and includes the nonlinear distortion which appears as random noise in the frequency domain. As a result the PGain trace looks “noisy”.</td>
</tr>
<tr>
<td>S11</td>
<td>Linear input match measured at linear power level</td>
<td>dB</td>
</tr>
<tr>
<td>S21</td>
<td>Linear gain measured at linear power level</td>
<td>dB</td>
</tr>
<tr>
<td>LPIn1</td>
<td>Input power level used for linear S-parameter measurement</td>
<td>dBm</td>
</tr>
<tr>
<td>LPOut1</td>
<td>Reflected power level at input during linear measurement</td>
<td>dBm</td>
</tr>
<tr>
<td>LPOut2</td>
<td>Output power level measured for linear measurement</td>
<td>dBm</td>
</tr>
</tbody>
</table>

**Distortion Tab Measurement Parameters**

<table>
<thead>
<tr>
<th>DUT Type</th>
<th>Description</th>
<th>Default Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amplifier</td>
<td>Mixer</td>
<td>Description</td>
</tr>
<tr>
<td>CarrIn1</td>
<td>CarrIn1</td>
<td>Input band power</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td>Unit</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>CarrOut2</td>
<td>Output band power</td>
<td>dBm/Hz</td>
</tr>
<tr>
<td>CarrGain21</td>
<td>Band power gain</td>
<td>dB</td>
</tr>
<tr>
<td>ACPIn1</td>
<td>Ratio of in-band channel power to channel power of the adjacent channel band at the input</td>
<td>dBc</td>
</tr>
<tr>
<td>ACPOut2</td>
<td>Ratio of in-band channel power to channel power of the adjacent channel band at the output</td>
<td>dBc</td>
</tr>
<tr>
<td>ACPDist21</td>
<td>ACP distortion added by DUT</td>
<td>dBc</td>
</tr>
<tr>
<td>ACPPwrIn1</td>
<td>ACP input band power</td>
<td>dBm/Hz</td>
</tr>
<tr>
<td>ACPPwrOut2</td>
<td>ACP output band power</td>
<td>dBm/Hz</td>
</tr>
<tr>
<td>EVMDistEq21</td>
<td>EVM equalized distortion added by DUT (non-linear contribution)</td>
<td>%</td>
</tr>
<tr>
<td>EVMDistUn21</td>
<td>EVM unequalized distortion added by DUT (includes non-linear and linear distortion due to frequency dispersion)</td>
<td>%</td>
</tr>
<tr>
<td>EVMPwrIn1</td>
<td>EVM input band power</td>
<td>dBm/Hz</td>
</tr>
<tr>
<td>EVMPwrOut2</td>
<td>EVM output band power</td>
<td>dBm/Hz</td>
</tr>
<tr>
<td>NPRIn1</td>
<td>NPR at input</td>
<td>dBc</td>
</tr>
<tr>
<td>NPROut2</td>
<td>NPR at output</td>
<td>dBc</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Unit</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>NPRDist21</td>
<td>NPR distortion added by DUT</td>
<td>dBC</td>
</tr>
<tr>
<td>NPRPwrIn1</td>
<td>NPR input band power</td>
<td>dBm/Hz</td>
</tr>
<tr>
<td>NPRPwrOut2</td>
<td>NPR output band power</td>
<td>dBm/Hz</td>
</tr>
<tr>
<td>ModFilter</td>
<td>Modulation filter used during measurement</td>
<td>dB</td>
</tr>
</tbody>
</table>

Right-clicking on the dialog accesses a pop up menu. Select **Show Expert Tabs** to display the **Receivers** tab for selecting one of the VNA receivers:
Distortion Table

The Distortion Table is displayed below the measurement area. Each row represents a measurement band. Each column represents a measurement parameter.

<table>
<thead>
<tr>
<th>Type</th>
<th>Carrier 1 dBm</th>
<th>Carrier 3 dBm</th>
<th>Carrier 5 dBm</th>
<th>ACP 1 dBm</th>
<th>ACP 3 dBm</th>
<th>ACP 5 dBm</th>
<th>ACP Up 1 dBm</th>
<th>ACP Up 3 dBm</th>
<th>EVM %</th>
</tr>
</thead>
</table>

Total EVM

If there is more than one EVM band, then an extra band labeled **Total EVM** is added at the bottom of the table. The band power column sums the band power. The EVM and power density columns are a weighted average. The columns that do not apply are blank (for example, Carrier Offset Frequency).

Saving the Distortion Table

The Distortion Table can be saved to a .csv file in one of three ways:

1. Using a mouse, right-click in the Distortion Table area, select **Save Table As...**, then select **Distortion Table (*.csv)** from the **Save as type** pulldown menu in the **Save Data As** dialog.
2. Press **Save Recall > Save Other > Save Data...**, then select **Distortion Table (*.csv)** from the **Save as**
1. Type pulldown menu in the **Save Data As** dialog.

3. Using a mouse, select **Utility**, **Save**, **Save Data**, then select **Distortion Table (*.csv)** from the **Save as type** pulldown menu in the **Save Data As** dialog.

### Accessing Distortion Table

**Using Hardkey /SoftTab /Softkey**

| 1. Press **Meas > Main** > click the small button to the left of the **Distortion Table...** button. |

**Using a mouse**

| 1. Click **Instrument**. |
| 2. Select **Display**. |
| 3. Select **Show Table...**. |
| 4. In the **Customize Display** dialog, select the **Show Table** pull-down. |
| 5. Select **Distortion**. |
| 6. Click **OK**. |

### Distortion Table help

**Save Table As...**

Specifies the Modulation Distortion Table path and file name (*.csv).

| **Edit Columns...** |
| **Edit Rows...** |
| **Hide Table** |
| **Save Table As...** |

### Editing Columns

The columns may be edited by clicking on **Meas > Main > Distortion Table...** button or by right-clicking in the distortion table at the bottom of the measurement area to access the pop up menu then selecting **Edit Columns...**.
The Distortion Table Setup dialog is displayed with tabs for adding/removing measurement parameters for each measurement type.

**Show Table**

Check to display the Distortion Table at the bottom of the VNA display area.

**Sort Table By**

Select Band to sort the Distortion Table by band or Power to sort the Distortion Table by power.

**Font Size**

Select Small or Medium font size for the Distortion Table.

**Carrier tab**

![Carrier tab interface](image)

**EVM tab**
Editing Rows

The rows may be edited by right-clicking in the distortion table at the bottom of the measurement area to access the pop up menu then selecting **Edit Rows...**.

The **Modulation Distortion Setup Measure** tab dialog is displayed. Click on the **Measurement Type** pulldown menu then select **Multi-Band** to access the table for adding, editing, and deleting band rows. Learn more.

Buttons

**Defaults** - Resets the dialog to its default selections.

**MOD Setup...** - Accesses the **Modulation Distortion Setup Measure** tab dialog. Learn more.
Calibration Overview

Calibrating a Modulation Distortion channel requires the following types of calibrations:

- Phase Reference Wizard
- S-Parameter Calibration
- Receiver IF Cal
- Source Modulation Calibration

See Also

- Phase Reference Wizard (separate topic)
- S-Parameter Calibration (separate topic)
- Receiver IF Cal (separate topic)
- Source Modulation Calibration (separate topic)

Phase Reference Wizard

A Phase Reference Cal is performed, saved, and later recalled during a S-parameter calibration. This is sometimes referred to as a 'tier 1" calibration. Due to stability of the VNA, the Phase Reference Cal can be performed infrequently. It is typically performed over the full frequency range of the VNA or Phase Reference so that it can be applied to all modulation distortion calibrations that will be needed in the future.

S-Parameter Calibration

The S-parameter calibration with power correction is the traditional method used to compute linear error terms. Enhanced response terms are applied for correction since only the forward direction is measured. The a1, b1, and b2 waves for each tone are accurately measured with enhanced response correction. The calibration frequency range should cover all of the analysis bandwidth required for the Modulation Distortion channel.
**Receiver IF Cal**

Receiver IF Calibration is an additional process for S-parameter calibration in a Modulation Distortion channel to improve the error term in the IF chain of the VNA. Receiver IF Cal measures the S21 of the thru device at different IF frequencies to compute the correction coefficient.

**Source Modulation Calibration**

The source modulation calibration compensates the IQ data to achieve a flat frequency response in gain and phase at the reference plane. Before this calibration, the receivers must be calibrated by performing the S-parameter calibration. The DUT is connected at the reference plane when performing the source modulation calibration to achieve the best accuracy.
S-Parameter Calibration

Note: Perform this calibration before the Source Modulation Calibration.

The example procedure in this topic describes how to perform a typical calibration using the Cal All Calibration Wizard method, which allows multiple channel calibrations in a single session. For more information about this calibration method, refer to Calibrate All Channels.

1. Ensure that the Modulation Distortion measurement class to be calibrated is currently active.

2. Press Cal > Main > Cal All... to launch the following dialog showing the active measurement class:

![Calibration Dialog Example](image)

3. Select the ports.

4. Click Next, then confirm or change the calibration properties in the Measurement Class Cal Properties dialog:

![Measurement Class Properties Example](image)

5. Click Next to access the Calibration Attenuator Settings dialog:
6. In the **Calibration Attenuator Settings** dialog, perform the following:

   a. Set the attenuator settings. [Learn more.](#)

   b. Click on the **Noise Reduction** button to improve measurement accuracy. [Learn more.](#)

   c. Click on the **Mechanical Devices** button to view all switches and attenuators in the VNA. [Learn more.](#)

7. Click **Next**, then select the DUT connectors and calibration kits in the **Select DUT Connectors and Cal Kits** dialog.

8. Click **Next** to access the **Power Cal Settings** dialog.

9. In the **Power Cal Settings** dialog, set the desired Tolerance and Max Readings. Learn more about [Accuracy Tolerance](#) and [Max Number of Readings](#).
10. Click **Next** and follow the calibration process until completed.
Receiver IF Calibration is an additional process for S-parameter calibration in a Modulation Distortion channel to improve the error term of the IF chain of the VNA. Receiver IF Cal measures the S21 of the thru device at different IF frequencies to compute the correction coefficient. The S-parameter Cal Set contains all of these error terms. The point count used for the receiver IF cal is determined by the tone spacing of the modulated signal and the user’s measurement span.

The receiver IF calibration uses the internal PNA CW source to make the measurement.

Receiver IF Cal Procedure

Perform the Receiver IF Cal if it is inaccurate or missing. This will be indicated in the Receiver IF Cal dialog as a warning message. An inaccurate Receiver IF Cal occurs when the Start, Stop, Center, or Span setting frequencies do not match the active S-parameter Cal Set.

*Note:* The following procedure assumes the Receiver IF Cal has never been performed and is therefore not part of the active Cal Set.

1. Ensure that the Modulation Distortion measurement class to be calibrated is currently active.
2. Press **Cal** > **Cal Sets & Cal Kits** > **Cal Set...** then select a Cal Set to apply to the active channel.
3. Press **Cal** > **Main** > **Other Cals**.
4. Select **Receiver IF Cal...**. The following dialog is displayed:

The **Close** button aborts the calibration.

If **Do not show this dialog again** is checked, the Receiver IF Cal dialog will not be displayed when a calibration is required.

5. Click on the **Calibrate IF** button. The following dialog is displayed:
6. Connect a thru device between the input port and the output port (use a low loss, passive component).

7. Click on the **OK** button.

8. If the calibration was successful, the status message will indicate a valid IF calibration.
Source Modulation Calibration

**Note:** The calibration information in this topic applies to both the Option S93070xB Modulation Distortion and Option S9x09xxA/B, S9x090A/B Spectrum Analysis applications.

**Note:** For Option S93070xB Modulation Distortion, traditional S-parameter calibration using Cal All... must be performed before performing a Source Modulation Calibration. Cal All.. is not required for Option S9x09xxA/B, S9x090A/B Spectrum Analysis.

In this topic:

- Initiating the Calibration
- Modulation Cal - Setup Dialog Description
- Source Modulation Correction ON | OFF
- Example Calibration Procedure

### Initiating the Calibration

#### Accessing Source Modulation Cal - Option S93070xB Modulation Distortion Application

**Using Hardkey/SoftTab/Softkey**

1. Press **Cal > Main > Other Cals**.
2. Select **Source Modulation Cal...**.

**Using a mouse**

1. Click **Response**.
2. Select **Cal**.
3. Select **Other Cals**.
4. Select **Source Modulation Cal...**.
Accessing Source Modulation Cal - Option S9x09xxA/B, S9x090A/B Spectrum Analysis Application

Using **Hardkey/SoftTab/Softkey**

1. Press **Freq > Main > SA Setup...**.
2. Select the **Source** tab.
3. Click in the **IQMod** column for your external source then select **Edit**.
4. Click on the **Calibrate Modulation...** button.

**Note:** The calibration information is saved in the source modulation .mdx file and is enabled/disabled by the Source Modulation Correction ON/OFF selection.

**Cal Type** - Select the type of calibration to perform:

- **Power** calibrates the total output power of the carrier integrated over the signal span.
- **Equalization** performs a linear pre-distortion calibration which equalizes the magnitude and phase of the modulated signal. The Cal Span defaults to the occupied BW of the Carrier.
Signal.

**LO Feedthru** minimizes the LO feedthru tone. (LO feedthru tone may be changed after calibration. LO Feedthru monitor allows you to check the LO feedthru at every measurement)

**Distortion** minimizes the vector error of the modulation signal over the Cal Span. The Cal Span defaults to the occupied BW of the Carrier Signal.

> **Note:** An Equalization calibration is included when a Distortion calibration is performed.

**NPR Notch** nulls the NPR notch. The Cal Span defaults to the frequency range of the notch. If there are multiple notches having different spans, then Cal Span displays **Various**.

**ACP Upper/ACP Lower** nulls the ACP upper/lower sideband of the signal. The Cal Span is set to the frequency width of the ACP sideband being calibrated. The default Cal Span is equal to the carrier span.

> **Note:** The Cal Span may be set so high that the source cannot correct the signal. In this case, the Cal Span will be reduced automatically. The maximum span is 80% of the source sampling frequency.

**Cal Port** - Selects the calibration plane used for the calibration.

> **Note:** The Cal Port selections for Distortion, NPR, and ACP are coupled. Changing one selection will change them all.

**Cal Span** - Sets the span of the calibration.

**Guard Band** - (ACP Cal only) Sets the frequency delta from the edge of the carrier to the beginning of the Cal Span. The default is set to zero. This value is positive for both ACP Lower and ACP Upper.

> **Note:** The Guard Band may be set so high that the source cannot correct the signal. In this case, the Cal Span will be reduced automatically. In addition, it is possible to set the Guard Band so high that the Cal Span is set to zero, in which case the Guard Band will also be limited automatically.

**Max Iterations** - The calibration routine uses successive approximation. This value sets the maximum number of iterations used by the calibration routine.

**Desired Tolerance** - Sets the desired tolerance for each calibration.

**Save Cal in File** - Saves the modulation calibration to a file.
Mod Cal Details... button - Opens the Mod Cal Details dialog.

Enable Faster Cal with Reduced Accuracy - Check to perform a faster calibration.

Use Previous Cal as Initial Value - Check to start the successive approximation search algorithm for the calibration with previously measured data as the initial value to increase the
search speed. If the .mdx file contains a previous calibration, then that calibration will be used as the initial value. If there is no previous calibration in the .mdx file, but the calibration is being performed over multiple power levels, then the calibration at a previous power level will be used as the initial value. The default is checked.

**RF Power for calibration**:

- **Fixed** - Calibration is performed at a fixed power level.
  - **Cal Power** - Sets the fixed power level.
- **Swept** - Calibration is performed at multiple power levels.
  - **Start** Sets the beginning value of the power sweep.
  - **Stop** Sets the end value of the power sweep.
  - **Nmbr of Points** Sets the number of power points to calibrate during a sweep.

**RF Carrier for calibration**:

- **Fixed** - Calibration is performed at a fixed frequency.
  - **Cal Frequency** - Sets the fixed frequency.
- **Swept** - Calibration is performed at multiple frequencies.
  - **Start** Sets the beginning frequency value.
  - **Stop** Sets the end frequency value.
  - **Nmbr of Points** - Sets the number of frequency points to calibrate during a sweep.

**Note:** Several powers and several frequencies can be set up and the VNA will calibrate all combinations.

**VNA Status** - Displays a list of parameters and their settings.

**File Properties... button**

**Waveform Tab**

Displays the properties of the currently active modulation file, which was selected in the Modulation settings dialog. This information cannot be edited.
Calibration Tab

The Calibration tab displays the calibration files and their properties stored in the .mdx file. Each calibration displayed in the list is for one power level. Calibrations may have been performed on multiple power levels during a single calibration. In this case, multiple calibrations will be saved in the .mdx file. Any of these calibrations can be deleted by selecting the calibration name then clicking on the Delete Cal button.

Next> button- When pressed, the wizard checks to see if the selected filename contains an existing cal file. If it does, then the following warning message is displayed:

Overwrite Cal Data - Existing cal data will be overwritten.
**Append to Cal Data** - Keeps all existing cal data and appends only the data for the currently selected **Cal Type**. For example, if **LO Feedthru** is selected as the only **Cal Type** in the **Modulation Cal - Setup** dialog and the calibration is performed, only the LO Feedthru cal data will be updated and the cal data for all previous cals will remain unchanged. If you repeat the LO Feedthru calibration, and append to the file, it will replace the LO Feedthru cal while leaving the other calibrations (distortion, equalization, NPR, and ACPR) unchanged.

**Choose Different File** - Allows selection of a different cal file.

Accesses the **Modulation Cal - Measure** dialog:

![Modulation Cal - Measure](image)

**Calibrate** button - Starts the calibration.

**Display** - Select to display one of the following:

- **Desired Modulation** is the ideal simulated signal from the modulation file data.
- **Uncorrected Modulation** is the real-time measured signal without correction.
- **Corrected Modulation** is the real-time measured signal with correction.
- **Correction Terms** displays a static trace showing the correction terms applied.
to the modulation data.

**Calibration Iterations** displays multiple static traces used during the calibration.

If one or more of the default tolerance values could not be achieved, a warning message is displayed. You have the option to change the tolerance values and re-calibrate or save the current calibration.

![Warning message](image)

---

**Source Modulation Correction ON | OFF**

- Click on **Cal > Main > Src Mod Correct** to toggle between ON and OFF.

- ONLY correction for the Modulation Distortion channel is turned ON and OFF.

---

**Example Calibration Procedure**

The Source Modulation Cal procedure optimizes the signal across the pass band with the DUT connected. The following example describes how to perform a Source Modulation calibration for a typical setup using an N5182B MXG.

1. If you have the Option S93070xB Modulation Distortion Application installed, perform the following steps:
   a. From the **Modulate** tab, select the source with the **Source** pulldown menu.
   b. Load the **Modulation File** using the **Load File...** button.
   c. Click **Source Cal...** button.
2. If you have the Option S9x09xxA/B, S9x090A/B Spectrum Analysis Application installed, perform the following steps:
   a. Click on the **SA Setup...** button to open the **SA Setup** dialog then select the **Source** tab.
   
   b. Click in the **IQMod** column for your external source and select **Edit** to access the **Modulation settings** dialog.
   
   c. Load the **Modulation Filename**.
   
   d. Click on the **Calibrate Modulation...** button.

3. The **Modulation Cal - Setup** dialog is displayed.

4. Select **Power** check box to calibrate power across the entire pass band.
5. Select **Equalization** check box to perform a linear pre-distortion calibration.

6. Under **Cal Port** for **Power** and **Equalization**, select **DUT In Port 1** to calibrate the input side, or **DUT Out Port 2** to calibrate the output side (select a receiver for SA Analysis option). This will perform a calibration at the input or output of the DUT.

7. The default values for **Cal Span**, **Max Iterations** and **Desired Tolerance** will be used. Also, the default fixed power level will be used that is set using the **Mod Cal Details** dialog (-10 dBm).

8. Click **Next**. The following dialog is displayed.
9. Make the connections shown in the **Modulation Cal -Measure** dialog.

10. Click **Calibrate** to begin the calibration. When finished, the corrected modulated signal is displayed.

11. Click on the **Finish** button. If one or more of the default tolerance values could not be achieved, a warning message is displayed. You have the option to change the tolerance values and re-calibrate or save the current calibration.

12. The **Enable Source Correction** or **Enable Modulation Correction** check box is checked.

13. Click **OK**.
**Setup Examples**

The Option S9x070xB Modulation Distortion application measures the nonlinear behavior of an RF microwave amplifier and converters under a modulated signal.

A modulation file is created, uploaded to a signal generator, then fed into the VNA to test the behavior of the device by measuring band power, ACP, and EVM.

In this topic:

- Example Distortion Setup Procedure
- Physical Setup - Typical
- Physical Setup - High Power
- Physical Setup - External
- Physical Setup - Converters

**See Also**

- Configuring Distortion Measurements
- Creating Modulation Files
- Displaying Distortion Parameters
- Programming Examples:
  - Measurement Setup
  - Create Modulation File
  - Display Data Setup
  - Source Modulation Calibration

**Example Distortion Setup Procedure**

The following example describes how to set up a typical measurement using an N5182B MXG. In this example, a modulation file for a Compact modulation type is created. Compact signals cut a slice of the IQ data from an original waveform. Learn more.

1. Connect the equipment as shown after this procedure.
2. On the VNA front panel, press *Preset*.
3. Perform the following steps to set up an external MXG signal generator:
a. On the VNA front panel, press Setup > External Hardware > External Device...

b. Click on the New button.

c. Click in the Name field and type a name for the source. For example, myMXG.

d. For Device Type, select Source.

e. For the Driver, select MXG_Vector.

f. Select Active - Show in UI.

g. Ensure that Enable IO is checked.

h. In the I/O Configuration field, type the VISA address of the MXG.

i. Click on the OK button. The following is an example:

4. On the VNA front panel, press Meas > S-Param > Meas Class...

5. Select Modulation Distortion, then either:

   - OK delete the existing measurement, or

   - New Channel to create the measurement in a new channel.

6. The Modulation Distortion Setup dialog will be displayed.

7. Click on the Sweep tab and define the parameters as shown below:
8. Click on the **RF Path** tab and define the parameters as shown below:
Nominal Src Amp - If there is a loss between the MXG source output and Port 1 of the VNA, then use a negative number. This value is used for power calibration and to set the power level at the output of the DUT.

To access the Offsets and Limits dialog, click on the Offsets and Limits... button.

Nominal DUT Gain - This value is used for power calibration.

9. Click on the Modulate tab.
10. Select the MXG source from the **Source** pull down.

11. Click on the **Create...** button to access the **Create Modulation** dialog.
12. For the **Modulation Type**, ensure that **Compact** is selected from the pull down menu. This is the default selection.

13. Click on the "... " button to the right of the **Filename** field to load the original file from which to create a compact signal.

14. For **Signal Span**, use the default value.

15. For **Tone Spacing** and **Number of Tones**, use the default values.

16. For **DAC Scaling**, use the default value. For better S/N, increase the scaling value until a DAC overload occurs.

17. For **Frequency Tolerance**, use default value of 1 %.

18. Click on the **Calculate** button then verify that the signal is reasonable.

19. Click the **Save...** button and save the compact signal file. The filename is displayed below the display window.
20. In the Display pull down menu, select **Spectrum-Ideal**. Signals similar to the following should be displayed:

21. In the Display pull down menu, select **Time**. Signals similar to the following should be displayed:
22. In the Display pull down menu, select CCDF. Signals similar to the following should be displayed:

**CCDF**

Increasing the number of tones results in the following:

- Finer tone spacing.
- Longer period for the compact test signal.
- More accurate CCDF as shown below.
23. In the Create Modulation dialog, click **OK**.

24. Click on the **Measure** tab and define the parameters as shown below:

25. Selecting **ACP+EVM** measures band power, ACP, and EVM for the specified frequency settings in the **Measure** tab.

26. Click **OK**.

27. To make a measurement check without calibration, perform the following steps:
a. Make a Thru connection between the Pin and Pout reference planes.

b. Press Format > Format 1 > Log Mag then select dBm/Hz.

c. Press Scale > Main > Scale then set the scale to 10 dBm Per Division.

d. Press Reference Level then set it to -70 dBm.

e. Press Reference Position then set it to 10 Div.

f. Press Display > Display Setup > Show Table then select Distortion.

28. Note the following:

a. The Modulation Distortion channel makes multiple background sweeps to complete the measurement. During the measurement, the carrier frequency of the compact signal does not change, but the VNA local frequency changes to cover the SA span (300 MHz in this example). The result is stitched together and stored in the Modulation Distortion channel.

b. The Power Spectral Density (PSD) dBm/Hz at the reference plane is displayed.
c. The power level is not calibrated at this point. The result shown is based on the factory calibration of the receiver.
29. Refer to S-Parameter Calibration for a procedure on calibrating the VNA receivers.
30. After performing an S-parameter calibration, refer to Source Modulation Calibration for a procedure on calibrating source power and flatness.
31. After calibration, perform the following steps:
   a. Press Trace > Trace 1-7 > New Traces...
   b. In the New Trace dialog, select Pin1 (Power In), POut2 (Power Out) and MDist2 (Modulation Distortion Out).
   c. Change the format to dBm/Hz.
   d. Scale as needed.
   e. If the Distortion Table is not displayed, Press Display > Display Setup > Show Table then select Distortion.
32. The following is an example showing input signal and output signal measuring ACPR and EVM.

![Graph showing input and output signals with ACPR and EVM measurements.]

33. Add/remove parametric values by right-clicking in the distortion table at the bottom of the measurement area to access the pop up menu then selecting Edit Columns...

![Pop-up menu with options to edit columns, rows, or hide the table.]

34. Make desired selections from the Distortion Table Setup dialog.
Physical Setup - Typical

The following diagram shows a typical hardware setup using an N5182B MXG with the signal connected to the VNA rear-panel Port 1 J10 input connector.

**Note:** Though this example uses an N5182B MXG, an M8190A with E8267D PSG, or an M9383A MCS can be used as the external source.
**Key Parameters**

Frequency: 4.5 GHz

BW Signal: 100 MHz

Pin Max: -10 dBm at reference plane

SA Span: 300 MHz

**Physical Setup - High Power**

The following diagram shows a typical high power hardware setup using an M8190A with E8267D PSG with the signal connected to the VNA rear-panel Port 1 J10 input connector.
Key Parameters

Frequency: 28 GHz

BW Signal: 400 MHz

Pin Max: 10 dBm at reference plane

SA Span: 1.2 GHz

Physical Setup - External

The following diagram shows a typical external hardware setup using an M8190A with E8267D PSG with the signal connected externally.
Key Parameters

Frequency: 28 GHz

BW Signal: 400 MHz

Pin Max: 0 dBm at reference plane

SA Span: 1.2 GHz

Physical Setup - Converter

The following diagram shows a typical hardware setup using an N5182B MXG with the signal connected to the VNA rear panel Port 1 J10 input connector. Another method of connecting an N5182B MXG is with the signal connected to the VNA rear panel Port 1 J9 and connecting J10 to J11. This allows switching between the MXG and Source 1 without a mechanical switch.
**Key Parameters**

The Key Parameters are dependent upon the mixer setup and the measurement setup.

- Carrier Frequency
- Carrier Power
- SA Center/Span
- Noise BW
- LO Frequency
- LO Power
- Measurement Type
- Measurement Offset Frequency
- Measurement IBW

See https://www.keisight.com/find/m981xa-install/ for more detailed setup information.
Modulation Distortion Measurement

The procedures in this section describe how to set up and measure Modulation Distortion. The topics include:

- Hardware Setup for Amplifiers
- Hardware Setup for Converters
- Create a Modulation Distortion Channel
- Set Up a Sweep
- Set Up the RF Path
- Set Up a Converter
- Set Up the External Source
- Set Up Source3 Modulation Source - Option S93072B

- Set Up the Modulation File
  - Modulation File Overview
  - Set Up a Compact Modulation Type
  - Set Up a Flat Tones Modulation Type
  - Set Up a NPR Notch Modulation Type

- Set Up Modulation Distortion Measurements
  - Set Up an ACP Measurement
  - Set Up an EVM Measurement
  - Set Up a Band Power Measurement
  - Set Up a NPR Measurement
  - Set Up a Multi-Band Measurement
  - Set Up a Power Sweep Measurement
Hardware Setup for Amplifiers

A Vector Signal Generator is used to generate a repetitive signal with a given CCDF (Complementary-Cumulative-Distribution-Function) and PSD (Power Spectral Density).

Physical Setup - Typical

The following diagram shows a typical hardware setup using an N5182B MXG with the signal connected to the VNA rear-panel Port 1 J10 input connector.

Note: Though the following setup example uses an N5182B MXG, an M8190A with E8267D PSG, or an M9383A MCS can also be used as the external source.

Physical Setup - High Power

The following diagram shows a typical high power hardware setup using an M8190A with E8267D PSG with the signal connected to the VNA rear-panel Port 1 J10 input connector.
Hardware Setup for Converters

The following diagram shows a typical hardware setup using an N5182B MXG with the signal connected to the VNA rear panel Port 1 J10 input connector. Another method of connecting an N5182B MXG is with the signal connected to the VNA rear panel Port 1 J9 and connecting J10 to J11. This allows switching between the MXG and Source 1 without a mechanical switch.
Create a Modulation Distortion Channel

In this topic:

- Modulation Distortion Channel
- Modulation Distortion Converter Channel

Modulation Distortion Channel

1. On the VNA front panel, press **Meas > S-Param > Meas Class...**
2. Select **Modulation Distortion**, then either:
   - **OK** delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.
3. In the **Confirm Measurement Class Change** dialog, click **OK** to proceed or **Cancel** to exit.
4. The **Modulation Distortion Setup** dialog will be displayed.
5. The default Distortion Measurement Parameter is PIn1. This means that the displayed trace is the input power incident on the DUT. This setting is set using the DUT Input selection in the RF Path tab.

Modulation Distortion Converter Channel

1. On the VNA front panel, press Meas > S-Param > Meas Class....

2. Select Modulation Distortion Converters, then either:
   
   - OK delete the existing measurement, or
   
   - New Channel to create the measurement in a new channel.

3. In the Confirm Measurement Class Change dialog, click OK to proceed or Cancel to exit.

4. The Modulation Distortion Mixer Setup dialog is displayed.

5. The default Distortion Measurement Parameter is PIn1. This means that the displayed trace is the input power incident on the DUT. This setting is set using the DUT Input selection in the RF Path tab.
Set Up a Sweep

1. If the **Modulation Distortion Setup** dialog is not displayed, press **Freq > SA Frequency > MOD Setup**.

2. The **Sweep, RF Path, Modulate**, or **Measure** tab functions can now be selected.

3. Select the **Sweep** tab.

4. Select **Fixed** from the **Sweep Type** pull down menu. The **Fixed** sweep measures a modulated signal with a fixed carrier LO frequency and power level. A **Power** sweep can be selected which sweeps the total power of the modulated signal at multiple power levels defined by a start/stop power level. [Learn more.](#)

5. Set the **Carrier Frequency** by either using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad. This is dependent upon the frequency of the DUT.

6. Set the **SA Center** by either using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad. This sets the Spectrum Analyzer display center.

7. Set the **SA Span** by either using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad. This sets the Spectrum Analyzer (SA) display span. The following graphic shows the SA display set to the same center frequency as the Carrier and covering a span that includes the integrated bandwidth settings of ACPLo + Carrier + ACPUp.
8. Set the **Noise BW** by either using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad. **Noise BW** is equal to the Resolution BW divided by the Vector Average factor.

9. Click on the **Sweep Details...** button to set the following:

   - **Force RF Power OFF at the End of Sweep - System Preference** - If enabled, RF power is turned off at the end of a sweep.

   - **Delay Before Start of Sweep** - Same as **Sweep Delay** in a standard channel.

   - **Delay Before Distortion Measurement** - Adds delay after the linear S-parameter sweep and before the distortion measurement to allow the RF source to settle.

   - **Enable S-Param Sweep** - Enables S-parameters to be measured at the low linear power level. If disabled (default), curve fitting (smoothing) is used instead of the S-parameter sweep. Curve fitting can be adjusted by varying the window aperture using the **Distortion Measurement Correlation Aperture** setting in the **Measurement Details** dialog.

   - **S-Param Source** - Selects source and signal used to measure S-parameters.

   - **External Vector - Chirp** - Selects a chirp signal from the external signal generator for measuring S-parameters.

   - **S-Param Power** - Sets the power level used at the DUT input to measure the DUT gain when operating in its linear region. The linear gain is used to calculate distortion.

   - **S-Param Freq Step** - Frequency step size setting used when measuring S-parameters.

   - **S-Param IFBW** - Bandwidth setting when measuring S-parameters.
Re-use Previous S-Parameter Measurements If Available - If disabled, S-parameters will be measured at the low linear power level. If enabled and compatible measurements exist from a previous sweep, those measurements will be used and a Linear S-parameter sweep will not be performed. If enabled and compatible measurements are not available, then a Linear S-parameter sweep will be performed.

10. Select the Set Power At pull down to set the power at either the DUT In or DUT Out. Adjust the power using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad.

11. To set up a power supply, click on the DC Sources... button which accesses the standard dialog for controlling power supplies. Learn more.

12. Click on the Apply button to apply the setting changes made in this dialog.
Set Up the RF Path

1. If the Modulation Distortion Setup dialog is not displayed, press Freq> SA Frequency > MOD Setup....

2. The Sweep, RF Path, Modulate, or Measure tab functions can now be selected.

3. Select the RF Path tab.

4. To change the attenuation in the source path, check Include under the VNA Source Attenuator then use either up/down arrows or double-click in the data entry field and enter the attenuation using the displayed keypad. The source power level to the input of the attenuator will be adjusted based on the attenuation value to ensure the power level specified in the Sweep tab is provided at either the DUT In or DUT Out.

5. The Nominal Source Amp sets the nominal gain (positive number) from an amplifier, or loss (negative number) due to an attenuation, cable loss, etc. This function provides a method of compensating port power for added attenuation or amplification in the source path. This value is used by the Set Power At function, receiver leveling, and calibration. This value is equal to the Power Offset setting found on the Offsets and Limits dialog. For purposes of this example, assume there is no external source amplifier or attenuation. Since there is no need to account for gain or loss, the value should be set to 0 dB.

6. Select the DUT Input using the drop down menu then select the VNA port number connected to the DUT input.

7. Set the nominal DUT gain by clicking in the Nominal DUT Gain data entry field then using either the up/down
arrows or double-clicking in the data entry field and entering the gain using the displayed keypad. This value is used by Set Power At DUT Out, receiver leveling, and calibration. This function is also displayed on the calibration dialog.

8. Select the **DUT Output** using the drop down menu then select the VNA port number connected to the DUT output.

9. To change the receiver attenuation, under **Receiver Attenuator** use either the up/down arrows or double-click in the data entry field and enter the attenuation using the displayed keypad.

10. Click on the **Path Configuration...** button and ensure that the correct RF path is selected according to the Hardware Setup then click **OK**. For this example, the Combiner path for Port 1 is selected since the modulated signal is connected to the rear-panel J10 connector.

11. Click on the **Apply** button to apply the setting changes made in this dialog.
Set Up a Converter

For purposes of these examples, only 1 converter stage will be configured.

In this topic:

- Procedure for Converter without Embedded LO
- Procedure for Converter with Embedded LO

Procedure for Converter without Embedded LO

1. If the Modulation Distortion Setup dialog is not displayed, press **Freq** > **SA Frequency** > **MOD Setup**.

2. The **Sweep**, **RF Path**, **Mixer**, **Modulate**, or **Measure** tab functions can now be selected.

3. Select the **Mixer** tab.

4. In the **Converter Stages** pull down menu, select 1.
5. Click on the **Source Name** pull down menu then select **Port 3**. This is the LO source.

6. Adjust the **Power** using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad.

7. Click on the **Leveling** pull down menu then select **Open Loop** or **Internal**. The **Open Loop** selection does not use ALC or receiver leveling. The **Internal** selection uses ALC leveling.

8. Adjust the receiver **Attenuator** using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad.

9. Adjust the **LO** frequency using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad.

10. Set the multiplier and denominator. The combination of numerator / denominator forms a fractional value that is multiplied by the input and LO frequency ranges. [Learn more.](#)

11. Adjust the **Input** using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad. This is the same as **Carrier Freq In** in the **Sweep** tab.

12. Click on the Output equation pull down menu then select the mixer configuration calculated from the output mixer equation selection.

13. To save the mixer setup, click on the **Save...** button then enter the file name and directory path.

### Procedure for Converter with Embedded LO

The following procedure describes how to set up measurements of mixers that have a fixed LO inside the DUT.

1. If the **Modulation Distortion Setup** dialog is not displayed, press **Freq > SA Frequency > MOD Setup**....

2. The **Sweep**, **RF Path**, **Mixer**, **Modulate**, or **Measure** tab functions can now be selected.

3. Select the **Mixer** tab.
4. In the **Converter Stages** pull down menu, select 1.

5. Adjust the **LO** frequency using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad.

6. Set the multiplier and denominator. The combination of numerator / denominator forms a fractional value that is multiplied by the input and LO frequency ranges. Learn more.

7. Adjust the **Input** using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad. This is the same as **Carrier Freq In** in the **Sweep** tab.

8. Click on the **Output equation** pull down menu then select the mixer configuration calculated from the output mixer equation selection.

9. Click on the **Setup...** button. The **Embedded LO** dialog is displayed:
10. Check the **Enable Embedded LO** check box.

11. Click on the **Tuning Method** pull down menu. These settings determine the amount of time spent versus the degree of accuracy to which the LO Frequency is measured. Accuracy is compromised when noise starts to appear on the measurement trace. Select from the following:

   - **Broadband and Precise** Does the entire tuning process for each background sweep.

   - **Precise Only** Does NOT perform broadband tuning on each sweep. Use this setting when the embedded LO is stable. The signal (after broadband) must be within $\frac{1}{2}$ the tuning IFBW. If the signal will always be within $\frac{1}{2}$ the IFBW, broadband tuning is not needed.

   - **Disable Tuning** Only the previously measured **LO Frequency Delta** is applied to the reference mixer LO and VNA receivers.

12. Set the **Tune every** interval at which tuning is performed before a measurement sweep using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad. For
example, 'Tune every 3 sweeps' means that every third measurement sweep is preceded by tuning sweeps.
If the embedded LO drifts, or if regularly changing DUTs, set it to 'Tune every 1 sweep'.

13. Set the **Broadband Search** frequency span over which to measure the embedded LO frequency using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad.

14. Set the **Noise BW** used for Broadband and Precise tuning sweeps using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad. This sets the resolution in the Broadband sweeps.

15. Set the **Max Iterations** of Precise sweeps to make using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad. When this number is reached, the final measurement is used.

16. Set the **Tolerance** using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad. When two consecutive Precise measurements are made within this value, the final measurement is used. If this is not achieved within the Max Iterations value, then the last measurement is used.

17. Click on the **Find Now** button to find and measure the actual LO frequency using the current dialog settings. This data is displayed in the **Status** box.

18. The **LO Frequency Delta** displays the absolute difference between the measured embedded LO frequency and the LO setting that is entered in the Mixer Tab dialog.

19. Click on the **OK** button.

20. In the **Mixer** tab, click on the **Apply** button to apply the settings.

21. To save the mixer setup, click on the **Save...** button then enter the file name and directory path.
Set Up the External Source

Perform the following steps to set up an external MXG signal generator:

**Note:** If you are setting up an M8190A Arbitrary Waveform Generator with E8267D PSG Vector Signal Generator, refer to [Drivers for External Compound Sources](#) for information about setting up compound sources.

1. On the VNA front panel, press **Setup > External Hardware > External Device....**
2. Click on the **New** button.
3. Click in the **Name** field and type a name for the source. For example, **myMXG**.
4. For **Device Type**, select **Source**.
5. For the **Driver**, select **MXG_Vector**. See a list of supported external source drivers.
6. Select **Active - Show in UI**.
7. Ensure that **Enable IO** is checked.
8. In the **I/O Configuration** field, type the VISA address of the MXG.
9. Click on the **OK** button. The following is an example:

![External Device Configuration: myMXG](image)
Set Up Source3 Modulation Source - Option S93072B

Option S93072B Arbitrary Waveform Generation on XSB Port (requires Option S93072B, Option XSB, Option S93070xB or Sx090A/B, and either Option 422 or Option 423) allows you to generate arbitrary waveforms including multitone signals.

The following PNA-X block diagram shows Option XSB with Option 423.

Source3 Setup Limitations

Sample Rate

Allowed sample rate values are integer divisors of 19.2 GHz from 19.2 GHz to 1 MHz.

Signal Span

Allowed signal span is 1 MHz to 6 GHz.

Tone Spacing/Waveform Period

Waveform Period can be from 1.666 nsec to 6.822 microseconds in multiples of 1.666 nsec. The waveform memory limitation is $2^{17}$ or 128k and the waveform length must be a multiple of 32. That is, the tone spacing of the underlying waveform must be 600 MHz/N where N is 1 to 4096.
Carrier Frequency

Signal Span/2 <= Carrier Frequency <= (6 GHz - Signal Span/2)

The carrier must land on the tone spacing grid:

- For odd number of tones, Carrier Frequency = \( N \times \text{Tone Spacing} \)
- For even number of tones, Carrier Frequency = \( N \times (\text{Tone Spacing})/2 \), where \( N \) is odd.

Select Source3 in Create Modulation Dialog

1. If the Modulation Distortion Setup dialog is not displayed, press Freq > SA Frequency > MOD Setup... .
2. The Sweep, RF Path, Modulate, or Measure tab functions can now be selected.
3. Select the Modulate tab.
4. In the Modulate tab, click on the Create... button to access the Create Modulation dialog.
5. In the **Modulation Type** pull down menu, select the desired modulation type.

6. In the **Source Name** pull down menu, select **Source3**.

7. For the remainder of the setup, refer to the procedure corresponding to the **Modulation Type** selection in Step 5.

   - Set Up a Compact Modulation Type
   - Set Up a Flat Tones Modulation Type
   - Set Up a NPR Notch Modulation Type
Set Up the Modulation File

The procedures in this section describe how to set up the Modulation Files. The topics include:

- Modulation File Overview
- Set Up a Compact Modulation Type
- Set Up a Flat Tones Modulation Type
- Set Up a NPR Notch Modulation Type
Modulation File Overview

Modulation files are created then loaded into the source. The modulation files created using the Create Modulation dialog include Flat Tones, NPR Notch, and a shortened version of a modulation file called Compact.

When viewed in the frequency domain, this signal is a mult-tone "grid" of frequencies. The data is measured on this multi-tone grid.

Compact Modulation Type

Compact test signals cut a slice of the IQ data from an original parent signal created using Keysight Signal Studio or created using a .csv file. The slice of IQ data is chosen that best fits in terms of statistical distribution (CCDF), then conditions the waveform so that it has the same spectrum signature as the original signal.
Keysight Signal Studio Waveform (.wfm)

The Keysight Signal Studio Software can be used to create the original parent signal in a .wfm file format for a compact modulation type measurement.

In order to use a compact file (.mdx) which is created from .wfm, the corresponding license is required on the Signal Generator to play the .mdx.

When the waveform pack license is used, the original waveform file must be assigned / locked to the signal generator in order to use the .mdx created from the original waveform file.

Note: Encrypted .wfm files created using the N5182B MXG RF Vector Signal Generator are not supported.

IQ File (.csv)

The original parent file can also be created using a .csv file format that has a timestamp, I, and Q that uses a comma to separate the values.

No license is required on the signal generator to play the .mdx created from the .csv.

Flat Tones Modulation Type

Flat tones are used to measure the frequency response of a device with a single input signal. This allows the gain and electrical delay through the device to be measured.

A modulated communications signal such as 5G NR or OFDM, when plotted versus frequency, is not a set of flat tones. It is relatively flat, but some tones may be very small (near zero). So those signals are not as useful for measuring gain and electrical delay, although it can be done by ignoring noisy data and smoothing.

It is possible to measure distortion (EVM and ACP) using flat tones. The creation of flat tones is
provided because it is very simple and is useful.

- Flat tones are often used with satellites (which are complicated converters):
  - Users are less concerned about communications formats and EVM.
  - Users are interested in gain and electrical delay through their device.
- Flat tones may also be used with any amplifier/converter:
  - Measure gain and electrical delay and EVM/ACP.
  - Make measurements using Modulation Distortion or Modulation Distortion Converters application.

### NPR Notch Modulation Type

NPR Notch modulation type is a Noise Power Ratio measurement. The notch is created where no signals are present. The location and span of the notch is set by the user. With the stimulus applied to the DUT, the ratio of the noise power of the carrier to the noise power in the notch is measured.
Set Up a Compact Modulation Type

1. If the **Modulation Distortion Setup** dialog is not displayed, press **Freq > SA Frequency > MOD Setup...**.

2. The **Sweep**, **RF Path**, **Modulate**, or **Measure** tab functions can now be selected.

3. Select the **Modulate** tab.

4. In the **Modulate** tab, click on the **Create...** button to access the **Create Modulation** dialog.
5. In the **Modulation Type** pull down menu, select **Compact**.

6. In the **Source Name** pull down menu, select the source. If it is not in the list, select **Add Source...** then refer to Set Up the External Source to set up a source.

7. In the **Filename** field, click on the "... " button to the right of this field and load the original (parent) signal file. The compact test signal will be based on this original signal and is generated automatically. The input waveforms are created using Signal Studio or IQ files in *.csv format. The resultant compact test signal version of the waveform matches the CCDF and PSD of the "parent-IQ waveform".

8. The settings under **Compact Signal** are filled in based on the original signal.

9. Click on the **Calculate** button then under **Calculated Result**, verify that the signal looks reasonable.

10. To make adjustments to the settings, perform the following procedure (optional):

   a. Set the **Number of Files** to 2 using the up arrow. This function is useful to create several signals, compare them, then save the best signal.

   b. To adjust the frequency span of the modulated carrier, click in the **Signal Span** data entry field then use the up/down arrows or double-click in the data entry field then enter the frequency using the displayed keypad.

   c. To adjust the distance between each tone, click in the **Tone Spacing** data entry field then use the up/down arrows or double-click in the data entry field then enter the frequency using the displayed keypad.
d. To adjust the number of tones, click in the **Number of Tones** data entry field then use the up/down arrows to make a change. Increasing the number of tones increases the CCDF accuracy.

e. To adjust the carrier offset value relative to the carrier LO frequency, click in the **Carrier Offset** data entry field then use the up/down arrows or double-click in the data entry field then enter the frequency using the displayed keypad.

f. To define a multicarrier signal, select 2, 3, 4, 5, 6, 7, 8, or 9 from the **Nmbr of Subcarriers** pull-down menu then set the corresponding **SubN Span** and **SubN Offset** (where "N" is the subcarrier number).

g. To adjust the the scaling factor used for the waveform (full scale = 100%), click in the **DAC Scaling** data entry field then use the up/down arrows or double-click in the data entry field then enter the value using the displayed keypad. This ensures that the DAC filter does not output a signal that is larger than the DAC's maximum output level, which can cause distortion in the system. Setting the scaling factor to 100% will usually cause excessive distortion.

h. To adjust where to start the compact signal within the original signal, check the **Priority** check box for **Signal Start Time**, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the start time using the displayed keypad.

i. With the **Enable Optimizer** check box checked, the calculated modulated signal will be optimized according to the constraints defined in **Optimize Signal** group box. For more information, refer to the description for Enable Optimizer and the Optimizer Setup dialog.

j. To adjust the allowed tolerance for tone spacing when calculating the modulation signal, select **Frequency Tolerance** from the **Optimize Signal** pull down menu, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the tolerance using the displayed keypad.

k. To adjust minimum period of the waveform greater than or equal to the specified value (seconds), select **Min Waveform Period** from the **Optimize Signal** pull down menu, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the period using the displayed keypad.

l. To adjust minimum number of tones greater than or equal to the specified value, select **Min Number of Tones** from the **Optimize Signal** pull down menu, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the number of tones using the displayed keypad. This will ignore the **Number of Tones** selection.

m. To adjust maximum tone spacing less than or equal to the specified value (Hz), select **Max Tone Spacing** from the **Optimize Signal** pull down menu, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the maximum tone spacing using the displayed keypad. This will ignore the **Tone Spacing** selection.

n. After making any adjustments, click on the **Calculate** button then, under **Calculated Result**, verify that the signal looks reasonable.

o. Use the **File** pull down menu to switch between the first and second file for comparison.

p. Choose the best signal.
11. Click the **Save**... button and save the compact test signal file. The filename is displayed below the display window.

![Image of Create Modulation window](image1)

12. In the **Display** pull down menu, select **Spectrum-Ideal**. Signals similar to the following should be displayed:

![Image of Spectrum-Ideal](image2)

13. In the **Display** pull down menu, select **Time**. Signals similar to the following should be displayed:
14. In the **Display** pull down menu, select **CCDF**. The following plot represents the complementary cumulative distribution function (CCDF) curve of the original (parent) signal, compact test signal, as well as the Gaussian distribution (pink trace):

15. Increasing the number of tones results in the following:

- Finer tone spacing.
- Longer period for the compact test signal.
- More accurate CCDF as shown below.
16. Click on the **OK** button. The modulation file is uploaded to the external source.

17. The **Autofill Measurements** message is displayed asking whether to autofill measurement frequencies or not. These frequencies are in the **Measure** tab. Learn more.

18. Click **Yes**.
Set Up a Flat Tones Modulation Type

1. If the **Modulation Distortion Setup** dialog is not displayed, press **Freq > SA Frequency > MOD Setup...**.

2. The **Sweep**, **RF Path**, **Modulate**, or **Measure** tab functions can now be selected.

3. Select the **Modulate** tab.

4. In the **Modulate** tab, click on the **Create...** button to access the **Create Modulation** dialog.
4. In the Modulation Type pull down menu, select Flat Tones.

5. In the Source Name pull down menu, select the source. If it is not in the list, select Add Source... then refer to Set Up the External Source to set up a source.

6. Click on the Calculate button then under Calculated Result, verify that the signal looks reasonable.

7. To make adjustments to the settings, perform the following procedure (optional):

   a. To adjust the frequency span of the modulated carrier, click in the Signal Span data entry field then
use the up/down arrows or double-click in the data entry field then enter the frequency using the displayed keypad.

b. To adjust the distance between each tone, click in the **Tone Spacing** data entry field then use the up/down arrows or double-click in the data entry field then enter the frequency using the displayed keypad.

c. To adjust the number of odd or even tones, select either **Nmbr Tones, Odd** or **Nmbr Tones, Even**, click in the data entry field, then use the up/down arrows to make a change. This setting forces the optimizer to choose an odd or even number of tones. The optimizer will also choose an offset such that the carrier lands on one of the tones.

d. To adjust the carrier offset value relative to the carrier LO frequency, click in the **Carrier Offset** data entry field then use the up/down arrows or double-click in the data entry field then enter the frequency using the displayed keypad.

e. To select a phase type, click on the **Phase Type** pull down menu then select **Random**, **Fixed**, or **Parabolic**. The **Random Phase Seed** sets the phase seed when **Random** phase is the **Phase Type**.

f. To adjust the the scaling factor used for the waveform (full scale = 100%), click in the **DAC Scaling** data entry field then use the up/down arrows or double-click in the data entry field then enter the value using the displayed keypad. This ensures that the DAC filter does not output a signal that is larger than the DAC's maximum output level, which can cause distortion in the system. Setting the scaling factor to 100% will usually cause excessive distortion.

g. With the **Enable Optimizer** check box checked, the calculated modulated signal will be optimized according to the constraints defined in **Optimize Signal** group box. For more information, refer to the description for Enable Optimizer and the Optimizer Setup dialog.

h. To adjust the allowed tolerance for tone spacing when calculating the modulation signal, select **Frequency Tolerance** from the **Optimize Signal** pull down menu, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the tolerance using the displayed keypad.

i. To adjust minimum period of the waveform greater than or equal to the specified value (seconds), select **Min Waveform Period** from the **Optimize Signal** pull down menu, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the period using the displayed keypad.

j. To adjust minimum number of tones greater than or equal to the specified value, select **Min Number of Tones** from the **Optimize Signal** pull down menu, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the number of tones using the displayed keypad. This will ignore the **Number of Tones** selection.

k. To adjust maximum tone spacing less than or equal to the specified value (Hz), select **Max Tone Spacing** from the **Optimize Signal** pull down menu, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the maximum tone spacing using the displayed keypad. This will ignore the **Tone Spacing** selection.

l. After making any adjustments, click on the **Calculate** button then, under **Calculated Result**, verify that the signal looks reasonable.
9. Click the **Save...** button and save the flat tone file. The filename is displayed below the display window.

![Calculated Result](image1)

10. In the **Display** pull down menu, select **Spectrum-Ideal**. Signals similar to the following should be displayed:

![Spectrum-Ideal](image2)

11. In the **Display** pull down menu, select **Time**. Signals similar to the following should be displayed:
12. In the Display pull down menu, select **CCDF**. The following plot represents the complementary cumulative distribution function (CCDF) curve of the flat tones test signal, as well as the Gaussian distribution (pink trace):

![Plot of flat tones and Gaussian distribution](image)

13. Increasing the number of tones results in the following:

- Finer tone spacing.
- Longer period for the test signal.
- More accurate CCDF as shown below.
14. Click on the **OK** button. The modulation file is uploaded to the external source.

15. The **Autofill Measurements** message is displayed asking whether to autofill measurement frequencies or not. These frequencies are in the **Measure** tab. Learn more.

16. Click **Yes**.
Set Up a NPR Notch Modulation Type

1. If the Modulation Distortion Setup dialog is not displayed, press Freq > SA Frequency > MOD Setup...

2. The Sweep, RF Path, Modulate, or Measure tab functions can now be selected.

3. Select the Modulate tab.

4. In the Modulate tab, click on the Create... button to access the Create Modulation dialog.
5. In the **Modulation Type** pull down menu, select **NPR Notch**.

6. In the **Source Name** pull down menu, select the source. If it is not in the list, select **Add Source...** then refer to Set Up the External Source to set up a source.

7. Click on the **Calculate** button then under **Calculated Result**, verify that the signal looks reasonable.

8. To make adjustments to the settings, perform the following procedure (optional):

   a. To adjust the frequency span of the modulated carrier, click in the **Signal Span** data entry field then
use the up/down arrows or double-click in the data entry field then enter the frequency using the displayed keypad.

b. To adjust the distance between each tone, click in the **Tone Spacing** data entry field then use the up/down arrows or double-click in the data entry field then enter the frequency using the displayed keypad.

c. To adjust the number of odd or even tones, select either **Nmbr Tones, Odd** or **Nmbr Tones, Even**, click in the data entry field, then use the up/down arrows to make a change. This setting forces the optimizer to choose an odd or even number of tones. The optimizer will also choose an offset such that the carrier lands on one of the tones.

d. To adjust the carrier offset value relative to the carrier LO frequency, click in the **Carrier Offset** data entry field then use the up/down arrows or double-click in the data entry field then enter the frequency using the displayed keypad.

e. To select a phase type, click on the **Phase Type** pull down menu then select **Random**, **Fixed**, or **Parabolic**. The **Random Phase Seed** sets the phase seed when **Random** phase is the **Phase Type**.

f. To set the number of notches, select the **Nmbr of Notches** pull down menu and select up to 3 notches.

g. To set the notch location, select the **Notch Location** pull down menu and select **Custom**, **Symmetric**, or **Avoid Carrier**.

- **Custom** selection allows the user to define the notch span using **Notch1 Span** and notch offset using **Notch1 Offset**.

- **Symmetric** selection locates the notch in the center of the **Signal Span** and allows the notch span to be changed using **Notch1 Span**.

- **Avoid Carrier** selection locates the notch near the center of the **Signal Span** but will be shifted to avoid the LO carrier feedthrough and allows the notch span to be changed using **Notch1 Span**.

h. To adjust the the scaling factor used for the waveform (full scale = 100%), click in the **DAC Scaling** data entry field then use the up/down arrows or double-click in the data entry field then enter the value using the displayed keypad. This ensures that the DAC filter does not output a signal that is larger than the DAC's maximum output level, which can cause distortion in the system. Setting the scaling factor to 100% will usually cause excessive distortion.

i. With the **Enable Optimizer** check box checked, the calculated modulated signal will be optimized according to the constraints defined in **Optimize Signal** group box. For more information, refer to the description for Enable Optimizer and the Optimizer Setup dialog.

j. To adjust the allowed tolerance for tone spacing when calculating the modulation signal, select **Frequency Tolerance** from the **Optimize Signal** pull down menu, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the tolerance using the displayed keypad.
k. To adjust minimum period of the waveform greater than or equal to the specified value (seconds), select **Min Waveform Period** from the **Optimize Signal** pull down menu, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the period using the displayed keypad.

l. To adjust minimum number of tones greater than or equal to the specified value, select **Min Number of Tones** from the **Optimize Signal** pull down menu, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the number of tones using the displayed keypad. This will ignore the **Number of Tones** selection.

m. To adjust maximum tone spacing less than or equal to the specified value (Hz), select **Max Tone Spacing** from the **Optimize Signal** pull down menu, click in its data entry field, then use the up/down arrows or double-click in the data entry field then enter the maximum tone spacing using the displayed keypad. This will ignore the **Tone Spacing** selection.

n. After making any adjustments, click on the **Calculate** button then, under **Calculated Result**, verify that the signal looks reasonable.

9. Click the **Save...** button and save the NPR Notch file. The filename is displayed below the display window.

![Calculated Result](image)

10. In the **Display** pull down menu, select **Spectrum-Ideal**. Signals similar to the following should be displayed:
11. In the **Display** pull down menu, select **Time**. Signals similar to the following should be displayed:

![Time signal](image1)

12. In the **Display** pull down menu, select **CCDF**. The following plot represents the complementary cumulative distribution function (CCDF) curve of the NPR Notch test signal, as well as the Gaussian distribution (pink trace):

![CCDF plot](image2)

13. Increasing the number of tones results in the following:
- Finer tone spacing.
- Longer period for the test signal.
- More accurate CCDF as shown below.

Number of Tones = 1,001

Number of Tones = 10,001

14. Click on the **OK** button. The modulation file is uploaded to the external source.

15. The **Autofill Measurements** message is displayed asking whether to autofill measurement frequencies or not. These frequencies are in the **Measure** tab. Learn more.

16. Click **Yes**.
Set Up Modulation Distortion Measurements

The procedures in this section describe how to set up Modulation Distortion measurements. The topics include:

- Set Up an ACP Measurement
- Set Up an EVM Measurement
- Set Up a Band Power Measurement
- Set Up a NPR Measurement
- Set Up a Multi-Band Measurement
- Set Up a Power Sweep Measurement
Set Up an ACP Measurement

ACP (Adjacent Channel Power) measures the channel power of the lower and upper adjacent channel band. DUT input and output ACP can be measured as well as the DUT contribution.

In this topic:

- ACP Measurement Criteria
- Upper and Lower Side Bands of the Input and Output Signal
- Upper and Lower Side Bands Contribution of the DUT

ACP Measurement Criteria

The procedures in the list below must be performed prior to making ACP measurements. For purposes of this example, the DUT used is an amplifier and the Modulation Type is Flat Tones.

- Required set up procedures:
  - Hardware Setup for Amplifiers
  - Create a Modulation Distortion Channel
  - Set Up a Sweep
  - Set Up the RF Path
  - Set Up the External Source
  - Set Up a Flat Tones Modulation Type
Calibration procedures:
- Phase Reference Wizard
- S-Parameter Calibration
- Receiver IF Cal
- Source Modulation Calibration

Upper and Lower Side Bands of the Input and Output Signal

1. If the Modulation Distortion Setup dialog is not displayed, press **Freq > SA Frequency > MOD Setup...**.

2. The **Sweep**, **RF Path**, **Modulate**, or **Measure** tab functions can now be selected.

3. Select the **Measure** tab.

4. In the **Measurement Type** pull down menu, select **ACP**.
5. Click on the **Autofill** button to automatically fill in the measurement settings for all bands from the currently active modulation file loaded in the source. Therefore, changing the settings in the following steps is optional.

6. To offset the Carrier integration bandwidth relative to the Carrier LO used to generate the modulation signal, use the up/down arrows in the **Carrier Offset Freq** field or double-click in the field then enter the value using the displayed keypad.

7. To set the Carrier integration bandwidth for the distortion measurement, use the up/down arrows in the **Carrier Integ BW** field or double-click in the field then enter the value using the displayed keypad. The IBW is used to determine total signal power within a specified frequency span.

8. To offset the lower ACP integration bandwidth relative to the LO used to generate the modulated signal, use the up/down arrows in the **ACPLo Offset Freq** field or double-click in the field then enter the value using the displayed keypad.

9. To set the integration bandwidth of the lower ACP measurement, use the up/down arrows in the **ACPLo Integ BW** field or double-click in the field then enter the value using the displayed keypad. The IBW is used to determine total signal power of the lower ACP within a specified frequency span.

10. To offset the upper ACP integration bandwidth relative to the LO used to generate the modulated signal, use the up/down arrows in the **ACPUp Offset Freq** field or double-click in the field then enter the value using the displayed keypad.

11. To set the integration bandwidth of the upper ACP measurement, use the up/down arrows in the **ACPUp Integ BW** field or double-click in the field then enter the value using the displayed keypad. The IBW is used to determine total signal power of the upper ACP within a specified frequency span.
12. To adjust other measurement settings, click on the Measurement Details... button:

![Measurement Details dialog](image)

a. To set the frequency span window used for modeling the DUT’s gain and distortion, use the up/down arrows in the Distortion Measurement Correlation Aperture field or double-click in the field then enter the value using the displayed keypad. Checking Auto will automatically set Distortion Measurement Correlation Aperture to window size.

b. To select the IF filter anti-aliasing path, click on the ADC Anti-alias Filter pull down menu then select Auto, Wide, or Narrow. Auto will automatically set the ADC Filter setting based on the ADC Sampling Frequency. If the currently selected modulation waveform was created with Nyquist Rejection = OFF, then the VNA will measure the signal using the Narrow anti-alias filter in the receiver. Wide selects the ADC 38 MHz IF filter path. Narrow selects the ADC 11 MHz IF filter path. A warning message will appear if the Narrow IF filter path is selected and the Resolution Bandwidth is > 1 MHz.

c. To set the measurement filter to either None (default) or RRC (root-raised-cosine filter), click on the Modulation Filter pull down menu then make the selection. With RRC selected, Alpha sets the Alpha factor of the filter and Symbol Rate sets the Symbol Rate of the filter. If Auto is selected, the Symbol Rate will be approximated from the bandwidth of the signal.

d. Click on the OK button.

13. Click on the Apply button to apply the setting changes made in this dialog.

14. Click on the OK button to close the dialog.

15. Ensure that the VNA Trigger is set to Continuous. Press Trigger > Main > Continuous.

16. The Distortion Table is displayed below the measurement area. Each column represents a measurement parameter. The measurement parameters shown in the Distortion Table below are the default parameters.
selected for an ACP measurement.

<table>
<thead>
<tr>
<th>Type</th>
<th>Carrier In1 dBm</th>
<th>Carrier Out2 dBm</th>
<th>Carrier Gain21 dB</th>
<th>Carrier IBW</th>
<th>ACP LowIn1 dBc</th>
<th>ACP LowOut2 dBc</th>
<th>ACP UpIn1 dBc</th>
<th>ACP UpOut2 dBc</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACP</td>
<td>-20.85 dBm</td>
<td>-7.85 dBm</td>
<td>13.00 dB</td>
<td>100.1 MHz</td>
<td>-44.83 dBc</td>
<td>-53.65 dBc</td>
<td>-45.17 dBc</td>
<td>-53.66 dBc</td>
</tr>
</tbody>
</table>

17. The measurement parameters are selected using the Distortion Table Setup dialog. To add or change parameters, press *Meas > Main > Distortion Table...* or right-click on the Distortion Table displayed below the measurement area then select *Edit Columns...* in the pop-up menu. The *Distortion Table Setup* dialog is displayed:

![Distortion Table Setup dialog](image)

18. Select the **ACP** tab.
19. Check the desired measurement parameters then click on the **OK** button.

20. To view traces displaying the ratio of upper and lower side bands of the input and output signal, perform the following:

   a. Press **Meas** > **Main** > **Other**... then click on the "..." button. The **Measure** dialog is displayed:
b. Select the **Distortion** tab to display distortion traces:
c. To display a trace measuring the ratio of upper and lower side bands of the input signal, check ACPIn1 then click on the Apply button.

d. In the Distortion tab, check New trace.

e. To display a trace measuring the ratio of upper and lower side bands of the output signal, check ACPOut2 then click on the OK button.

21. The following is an example of a typical trace and Distortion Table showing the ACP lower and upper input/output:

![Typical trace and Distortion Table](image)

**Upper and Lower Side Bands Contribution of the DUT**

After setting up and performing an ACP measurement as described in the above procedure, perform the following procedure to measure the distortion contribution by the DUT.

1. Press Meas > Main > Distortion Table... or right-click on the Distortion Table displayed below the measurement area then select Edit Columns... in the pop-up menu.

2. In the Distortion Table Setup dialog, select the ACP tab.
3. Check the **ACP LoDist21 dBc** (lower side added by DUT) and **ACP UpDist21 dBc** (upper side added by DUT) parameters then click on the **OK** button. The added parameters are displayed as a new column in the Distortion Table displayed below the measurement area.

4. To view a trace displaying the distortion contribution by the DUT, perform the following:

   a. Press **Meas > Main > Other...** then click on the " ... " button. The **Measure** dialog is displayed:
b. Select the **Distortion** tab to display distortion traces:
c. In the **Distortion** tab, check **New trace**. If any parameters are currently checked, uncheck them.

d. Check **ACPDist21** then click on the **Apply** button.

e. If necessary, press **Scale > Main > Scale Coupling ...**, select **Window** as the **Coupling Method**, then click **OK**. This re-scales all traces to the same scale.

f. Click on the **Autoscale All** softkey.

5. The following is an example of a typical trace and Distortion Table showing the lower and upper contributions by the DUT:
Set Up an EVM Measurement

To compute EVM, the input and output response of the DUT in the frequency domain is measured. Spectrum correlation is then performed to compute the EVM. The procedures in this topic describe how to measure the equalized EVM of the DUT (non-linear contribution) and the un-equalized EVM of the DUT (Includes non-linear and linear distortion due to frequency dispersion).

In this topic:

- EVM Measurement Criteria
- Equalized EVM of the DUT (non-linear contribution)
- Un-equalized EVM of the DUT (Includes non-linear and linear distortion due to frequency dispersion)

EVM Measurement Criteria

The procedures in the list below must be performed prior to making EVM measurements. For purposes of this example, the DUT used is an amplifier and the Modulation Type is Flat Tones.

- Required set up procedures:
  - Hardware Setup for Amplifiers
  - Create a Modulation Distortion Channel
  - Set Up a Sweep
  - Set Up the RF Path
  - Set Up the External Source
  - Set Up a Flat Tones Modulation Type

- Calibration procedures:
  - Phase Reference Wizard
  - S-Parameter Calibration
  - Receiver IF Cal
  - Source Modulation Calibration

Equalized EVM of the DUT (non-linear contribution)

1. If the Modulation Distortion Setup dialog is not displayed, press Freq > SA Frequency > MOD Setup...

2. The Sweep, RF Path, Modulate, or Measure tab functions can now be selected.

3. Select the Measure tab.
4. In the **Measurement Type** pull down menu, select **EVM**.
5. Click on the Autofill button to automatically fill in the measurement settings for all bands from the currently active modulation file loaded in the source. Therefore, changing the settings in the following steps is optional.

6. To offset the Carrier integration bandwidth relative to the Carrier LO used to generate the modulation signal, use the up/down arrows in the Carrier Offset Freq field or double-click in the field then enter the value using the displayed keypad.

7. To set the Carrier integration bandwidth for the distortion measurement, use the up/down arrows in the Carrier Integ BW field or double-click in the field then enter the value using the displayed keypad. The IBW is used to determine total signal power within a specified frequency span.

8. To adjust other measurement settings, click on the Measurement Details... button:
a. To set the frequency span window used for modeling the DUT’s gain and distortion, use the up/down arrows in the Distortion Measurement Correlation Aperture field or double-click in the field then enter the value using the displayed keypad. Checking Auto will automatically set Distortion Measurement Correlation Aperture to window size.

b. To select the IF filter anti-aliasing path, click on the ADC Anti-alias Filter pull down menu then select Auto, Wide, or Narrow. Auto will automatically set the ADC Filter setting based on the ADC Sampling Frequency. If the currently selected modulation waveform was created with Nyquist Rejection = OFF, then the VNA will measure the signal using the Narrow anti-alias filter in the receiver. Wide selects the ADC 38 MHz IF filter path. Narrow selects the ADC 11 MHz IF filter path. A warning message will appear if the Narrow IF filter path is selected and the Resolution Bandwidth is > 1 MHz.

c. To set the measurement filter to either None (default) or RRC (root-raised-cosine filter), click on the Modulation Filter pull down menu then make the selection. With RRC selected, Alpha sets the Alpha factor of the filter and Symbol Rate sets the Symbol Rate of the filter. If Auto is selected, the Symbol Rate from the file is used. If no Symbol Rate is indicated in the file, then the Symbol Rate will be approximated from the bandwidth of the signal.

d. To set the scaling factor applied to the EVM measurements. Enter a value between 0.1 and 1.0. The default is 1.0 using the up/down arrows in the EVM Normalization field.

e. To set the DUT noise figure, use the up/down arrows in the Nominal DUT NF field or double-click in the field then enter the value using the displayed keypad. This value is used by the EVM measurement. This setting adds noise to the measurement for more realistic results. The default is 0 dB. Setting the noise figure to a low value (-200 dB) will make EVM measurements without the effects of noise.

f. Click on the OK button.

9. Click on the Apply button to apply the setting changes made in this dialog.
10. Click on the **OK** button to close the dialog.

11. Ensure that the VNA Trigger is set to Continuous. Press **Trigger > Main > Continuous**.

12. The Distortion Table is displayed below the measurement area. Each column represents a measurement parameter. The measurement parameters shown in the Distortion Table below are the default parameters selected for an EVM measurement.

<table>
<thead>
<tr>
<th>Type</th>
<th>Carrier In1 dBm</th>
<th>Carrier Out2 dBm</th>
<th>Carrier Gain21 dB</th>
<th>Carrier IBW</th>
<th>EVM DistEq21 %</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVM</td>
<td>-20.07 dBm</td>
<td>-7.84 dBm</td>
<td>-13.00 dB</td>
<td>100.1 MHz</td>
<td>0.69 %</td>
</tr>
</tbody>
</table>

13. The measurement parameters are selected using the Distortion Table Setup dialog. To add or change parameters, press **Meas > Main > Distortion Table...** or right-click on the Distortion Table displayed below the measurement area then select **Edit Columns...** in the pop-up menu. The Distortion Table Setup dialog is displayed:

14. Select the **EVM** tab.
15. Check the desired measurement parameters then click on the OK button.

16. To view a trace displaying the equalized EVM of the DUT (non-linear contribution), perform the following:
   
a. Press **Meas > Main > Other...** then click on the "..." button. The **Measure** dialog is displayed:
b. Select the **Distortion** tab to display distortion traces:
c. Check **EVMDistEq21** then click on the **OK** button.

17. The following is an example of a typical trace and Distortion Table showing equalized distortion of the DUT:

![Graph of EVM vs Frequency](image)

**Un-equalized EVM of the DUT (Includes non-linear and linear distortion due to frequency dispersion)**

After setting up and performing an EVM measurement as described in the above procedure, perform the following procedure for a measurement that includes non-linear and linear distortion due to frequency dispersion.

1. Press **Meas > Main > Distortion Table**... or right-click on the Distortion Table displayed below the measurement area then select **Edit Columns**... in the pop-up menu.

2. In the Distortion Table Setup dialog, select the **EVM** tab.
3. Check the EVM DistUn21 % parameter then click on the OK button. The added parameter is displayed as a new column in the Distortion Table displayed below the measurement area.

4. To view a trace displaying the un-equalized EVM of the DUT, perform the following:

   a. Press Meas > Main > Other... then click on the "..." button. The Measure dialog is displayed:
b. Select the **Distortion** tab to display distortion traces:
c. In the **Distortion** tab, check **New trace**. If any parameters are currently checked, uncheck them.

d. Check **EVMDistUn21** then click on the **OK** button.

5. The following is an example of a typical trace and Distortion Table showing un-equalized distortion of the DUT:
Set Up a Band Power Measurement

Band Power measures the input carrier band power, output carrier band power, and band power gain of the DUT (magnitude and phase).

In this topic:

- Band Power Measurement Criteria
- Input/Output Band Power and Band Power Gain (magnitude and phase)

Band Power Measurement Criteria

The procedures in the list below must be performed prior to making Band Power measurements. For purposes of this example, the DUT used is an amplifier and the Modulation Type is Flat Tones.

- Required set up procedures:
  - Hardware Setup for Amplifiers
  - Create a Modulation Distortion Channel
  - Set Up a Sweep
  - Set Up the RF Path
  - Set Up the External Source
  - Set Up a Flat Tones Modulation Type

- Calibration procedures:
  - Phase Reference Wizard
  - S-Parameter Calibration
  - Receiver IF Cal
  - Source Modulation Calibration

Input/Output Band Power and Band Power Gain (magnitude and phase)

1. If the Modulation Distortion Setup dialog is not displayed, press Freq > SA Frequency > MOD Setup...

2. The Sweep, RF Path, Modulate, or Measure tab functions can now be selected.

3. Select the Measure tab.
4. In the **Measurement Type** pull down menu, ensure that **Band Power** is selected (default).

5. Click on the **Autofill** button to automatically fill in the measurement settings for all bands from the currently active modulation file loaded in the source. Therefore, changing the settings in the following steps is optional.

6. To offset the Carrier integration bandwidth relative to the Carrier LO used to generate the modulation signal, use the up/down arrows in the **Carrier Offset Freq** field or double-click in the field then enter the value using the displayed keypad.

7. To set the Carrier integration bandwidth for the distortion measurement, use the up/down arrows in the **Carrier Integ BW** field or double-click in the field then enter the value using the displayed keypad. The IBW is used to determine total signal power within a specified frequency span.

8. To adjust other measurement settings, click on the **Measurement Details...** button:
To set the frequency span window used for modeling the DUT’s gain and distortion, use the up/down arrows in the Distortion Measurement Correlation Aperture field or double-click in the field then enter the value using the displayed keypad. Checking Auto will automatically set Distortion Measurement Correlation Aperture to window size.

To select the IF filter anti-aliasing path, click on the ADC Anti-alias Filter pull down menu then select Auto, Wide, or Narrow. Auto will automatically set the ADC Filter setting based on the ADC Sampling Frequency. If the currently selected modulation waveform was created with Nyquist Rejection = OFF, then the VNA will measure the signal using the Narrow anti-alias filter in the receiver. Wide selects the ADC 38 MHz IF filter path. Narrow selects the ADC 11 MHz IF filter path. A warning message will appear if the Narrow IF filter path is selected and the Resolution Bandwidth is > 1 MHz.

To set the measurement filter to either None (default) or RRC (root-raised-cosine filter), click on the Modulation Filter pull down menu then make the selection. With RRC selected, Alpha sets the Alpha factor of the filter and Symbol Rate sets the Symbol Rate of the filter. If Auto is selected, the Symbol Rate from the file is used. If no Symbol Rate is indicated in the file, then the Symbol Rate will be approximated from the bandwidth of the signal.

Click on the OK button.

9. Click on the Apply button to apply the setting changes made in this dialog.

10. Click on the OK button to close the dialog.

11. Ensure that the VNA Trigger is set to Continuous. Press Trigger > Main > Continuous.

12. The Distortion Table is displayed below the measurement area. Each column represents a measurement parameter. The measurement parameters shown in the Distortion Table below are the default parameters selected for a Band Power measurement.
13. The measurement parameters are selected using the Distortion Table Setup dialog. To add or change parameters, press **Meas > Main > Distortion Table...** or right-click on the Distortion Table displayed below the measurement area then select **Edit Columns...** in the pop-up menu. The **Distortion Table Setup** dialog is displayed:

![Distortion Table Setup dialog](image)

14. Check the desired measurement parameters then click on the **OK** button.

15. To view a trace displaying the Input/Output Band Power and Band Power Gain, perform the following:

   a. Press **Meas > Main > Other...** then click on the "..." button. The **Measure** dialog is displayed:
b. Select the **Distortion** tab to display distortion traces:
c. Check **CarrIn1** to measure input Band Power then click on the **Apply** button.

d. In the **Distortion** tab, check **New trace**. If any parameters are currently checked, uncheck them.

e. Check **CarrOut2** to measure the output Band Power then click on the **Apply** button.

f. In the **Distortion** tab, check **New trace**. If any parameters are currently checked, uncheck them.

g. Check **CarrGain21** to measure the Band Power gain then click on the **OK** button.

h. If necessary, press **Scale** > **Main** > **Scale Coupling** ..., select **Window** as the **Coupling Method**, then click **OK**. This re-scales all traces to the same scale.

i. Click on the **Autoscale All** softkey.

16. The following is an example of a typical trace and Distortion Table showing input and output Band Power and Band Power Gain:
Set Up a NPR Measurement

NPR Notch is a Noise Power Ratio measurement. The notch is created where no signals are present. With the stimulus applied to the DUT, the ratio of the noise power of the carrier to the noise power in the notch is measured.

In this topic:

- NPR Measurement Criteria
- Input/Output NPR and NPR Contribution of the DUT

NPR Measurement Criteria

The procedures in the list below must be performed prior to making NPR measurements. For purposes of this example, the DUT used is an amplifier and the Modulation Type is NPR Notch.

- Required set up procedures:
  - Hardware Setup for Amplifiers
  - Create a Modulation Distortion Channel
  - Set Up a Sweep
  - Set Up the RF Path
  - Set Up the External Source
  - Set Up a NPR Notch Modulation Type

- Calibration procedures:
  - Phase Reference Wizard
  - S-Parameter Calibration
  - Receiver IF Cal
  - Source Modulation Calibration

Input/Output NPR and NPR Contribution of the DUT

1. If the Modulation Distortion Setup dialog is not displayed, press Freq > SA Frequency > MOD Setup...

2. The Sweep, RF Path, Modulate, or Measure tab functions can now be selected.

3. Select the Measure tab.
4. In the **Measurement Type** pull down menu, select **NPR**.
5. Click on the **Autofill** button to automatically fill in the measurement settings for all bands from the currently active modulation file loaded in the source. Therefore, changing the settings in the following steps is optional.

6. To offset the Carrier integration bandwidth relative to the Carrier LO used to generate the modulation signal, use the up/down arrows in the **Carrier Offset Freq** field or double-click in the field then enter the value using the displayed keypad.

7. To set the Carrier integration bandwidth for the distortion measurement, use the up/down arrows in the **Carrier Integ BW** field or double-click in the field then enter the value using the displayed keypad. The IBW is used to determine total signal power within a specified frequency span.

8. To offset the Notch center relative to the carrier LO used to generate the modulation signal, use the up/down arrows in the **Notch Offset Freq** field or double-click in the field then enter the value using the displayed keypad.

9. To set the integration bandwidth of the NPR notch measurement, use the up/down arrows in the **Notch Integ BW** field or double-click in the field then enter the value using the displayed keypad.

10. To adjust other measurement settings, click on the **Measurement Details...** button:

![Measurement Details](image)

    a. To set the frequency span window used for modeling the DUT’s gain and distortion, use the up/down arrows in the **Distortion Measurement Correlation Aperture** field or double-click in the field then enter the value using the displayed keypad. Checking **Auto** will automatically set **Distortion Measurement Correlation Aperture** to window size.

    b. To select the IF filter anti-aliasing path, click on the **ADC Anti-alias Filter** pull down menu then select **Auto**, **Wide**, or **Narrow**. **Auto** will automatically set the ADC Filter setting based on the ADC Sampling Frequency. If the currently selected modulation waveform was created with Nyquist Rejection = OFF, then the VNA will measure the signal using the Narrow anti-alias filter in the receiver. **Wide** selects the ADC 38 MHz IF filter path. **Narrow** selects the ADC 11 MHz IF filter path. A warning
message will appear if the Narrow IF filter path is selected and the Resolution Bandwidth is > 1 MHz.

c. To set the measurement filter to either None (default) or RRC (root-raised-cosine filter), click on the Modulation Filter pull down menu then make the selection. With RRC selected, Alpha sets the Alpha factor of the filter and Symbol Rate sets the Symbol Rate of the filter. If Auto is selected, the Symbol Rate from the file is used. If no Symbol Rate is indicated in the file, then the Symbol Rate will be approximated from the bandwidth of the signal.

d. Click on the OK button.

11. Click on the Apply button to apply the setting changes made in this dialog.

12. Click on the OK button to close the dialog.

13. Ensure that the VNA Trigger is set to Continuous. Press Trigger > Main > Continuous .

14. The Distortion Table is displayed below the measurement area. Each column represents a measurement parameter. The measurement parameters shown in the Distortion Table below are the default parameters selected for an NPR measurement. In this case, only the output NPR is selected so the input NPR will be added in the following steps.

<table>
<thead>
<tr>
<th>Type</th>
<th>Carrier In1 dBm</th>
<th>Carrier Out2 dBm</th>
<th>Carrier Gain21 dB</th>
<th>Carrier IBW</th>
<th>NPR Out2 dBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPR</td>
<td>-19.15 dBm</td>
<td>-7.84 dBm</td>
<td>-13.00 dB</td>
<td>90.00 MHz</td>
<td>-42.07 dBc</td>
</tr>
</tbody>
</table>

15. The measurement parameters are selected using the Distortion Table Setup dialog. To add or change parameters, press Meas > Main > Distortion Table... or right-click on the Distortion Table displayed below the measurement area then select Edit Columns... in the pop-up menu. The Distortion Table Setup dialog is displayed:
16. Select the **NPR** tab.
17. Check the **NPR In1 dBc** (input NPR) and **NPR Dist21 dBc** (NPR contribution by DUT) measurement parameters then click on the **OK** button.

18. To view traces displaying the Input/Output NPR and NPR contribution by DUT, perform the following:

   a. Press **Meas > Main > Other...** then click on the "..." button. The **Measure** dialog is displayed:
b. Select the **Distortion** tab to display distortion traces:
c. Check NPRIn1 to measure input NPR then click on the Apply button.

d. In the Distortion tab, check New trace. If any parameters are currently checked, uncheck them.

e. Check NPROut2 to measure the output NPR then click on the Apply button.

f. In the Distortion tab, check New trace. If any parameters are currently checked, uncheck them.

g. Check NPRDist21 to measure the NPR contribution by the DUT then click on the OK button.

h. If necessary, press Scale > Main > Scale Coupling ..., select Window as the Coupling Method, then click OK. This re-scales all traces to the same scale.

i. Click on the Autoscale All softkey.

19. The following is an example of a typical trace and Distortion Table showing input and output NPR and NPR contribution of the DUT:
Set Up a Multi-Band Measurement

The Multi-Band measurement allows multiple measurement types with corresponding measurement data to be displayed in a single table.

In this topic:

- Multi-Band Measurement Criteria
- Multi-Band Measurement Setup

Multi-Band Measurement Criteria

The procedures in the list below must be performed prior to making Multi-Band measurements. For purposes of this example, the DUT used is an amplifier and the Modulation Type is Flat Tones.

- Required set up procedures:
  - Hardware Setup for Amplifiers
  - Create a Modulation Distortion Channel
  - Set Up a Sweep
  - Set Up the RF Path
  - Set Up the External Source
  - Set Up a Flat Tones Modulation Type

- Calibration procedures:
  - Phase Reference Wizard
  - S-Parameter Calibration
  - Receiver IF Cal
  - Source Modulation Calibration

Multi-Band Measurement Setup

1. If the Modulation Distortion Setup dialog is not displayed, press Freq > SA Frequency > MOD Setup...

2. The Sweep, RF Path, Modulate, or Measure tab functions can now be selected.

3. Select the Measure tab.
4. In the **Measurement Type** pull down menu, select **Multi-Band**.
5. Click on the **Edit Table** pull down menu then select **Append Band**.

6. In the **MeasType** column for the new entry, select **ACP**.

7. Click on the **Edit Table** pull down menu then select **Append Band**.

8. In the **MeasType** column for the new entry, select **EVM**.

9. Click on the **Edit Table** pull down menu then select **Autofill Current Table from Mod File**. Measurement settings from the currently active modulation file loaded in the source are used for all measurement types in the table.

10. To change the band name, right-click in the **Band Name** field to display a keyboard for editing the name.

11. Type in the band name then click on **Enter**.

12. Click on the **Apply** button to apply the changes. The Multi-Band table is displayed below the measurement area with the default measurement parameters.

13. Click on the **OK** button to close the dialog.

14. Additional measurement parameters can be selected for any measurement type using the Distortion Table Setup dialog. Press **Meas > Main > Distortion Table** or right-click on the Distortion Table displayed below the measurement area then select **Edit Columns**... in the pop-up menu. The **Distortion Table Setup** dialog is displayed:
15. Select the measurement type tab then add the desired measurement parameters.

16. Click on the **OK** button to close the dialog.

17. Ensure that the VNA Trigger is set to Continuous. Press **Trigger > Main > Continuous**.

18. The following shows a Multi-Band measurement:

<table>
<thead>
<tr>
<th>Band</th>
<th>Type</th>
<th>Carrier In1 dBm</th>
<th>Carrier Out2 dBm</th>
<th>Carrier Gain21 dB</th>
<th>Carrier IBW</th>
<th>EVM DistEq21 %</th>
<th>ACP Loin1 dBC</th>
<th>ACP Loin2 dBC</th>
<th>ACP UpIn1 dBC</th>
<th>ACP UpOut2 dBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Band 1</td>
<td>Band Power</td>
<td>-20.03 dBm</td>
<td>-7.09 dBm</td>
<td>12.94 dB</td>
<td>100.1 MHz</td>
<td>-44.70 dBC</td>
<td>-53.62 dBC</td>
<td>-44.87 dBC</td>
<td>-53.72 dBC</td>
<td></td>
</tr>
<tr>
<td>Band 2</td>
<td>ACP</td>
<td>-20.03 dBm</td>
<td>-7.09 dBm</td>
<td>12.94 dB</td>
<td>100.1 MHz</td>
<td>-44.70 dBC</td>
<td>-53.62 dBC</td>
<td>-44.87 dBC</td>
<td>-53.72 dBC</td>
<td></td>
</tr>
<tr>
<td>Band 3</td>
<td>EVM</td>
<td>-20.03 dBm</td>
<td>-7.09 dBm</td>
<td>12.94 dB</td>
<td>100.1 MHz</td>
<td>-44.70 dBC</td>
<td>-53.62 dBC</td>
<td>-44.87 dBC</td>
<td>-53.72 dBC</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** NPR was not measured because it requires a NPR Notch modulation type instead of the Flat Tones modulation type used here to measure ACP, EVM, and Band Power.
Set Up a Power Sweep Measurement

A Power Sweep measurement can be set up to sweep power within a specified power range and specified number of powers. A trace can be set up to display input power on the X axis with the EVM measurement on the Y axis. This displays a "bathtub" type curve due to noise at low power levels and distortion at high power levels.

1. If the Modulation Distortion Setup dialog is not displayed, press Freq > SA Frequency > MOD Setup...

2. The Sweep, RF Path, Modulate, or Measure tab functions can now be selected.

3. Select the Sweep tab.

4. Click on the Sweep Type pull down menu then select Power.

5. Set the Carrier Frequency by either using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad. This is dependent upon the frequency of the DUT.

6. Set the SA Center by either using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad. This sets the Spectrum Analyzer display center.

7. Set the SA Span by either using the up/down arrows or by double-clicking in the data entry field and entering
the frequency using the displayed keypad. This sets the Spectrum Analyzer (SA) display span. The following graphic shows the SA display set to the same center frequency as the Carrier and covering a span that includes the integrated bandwidth settings of ACPLo + Carrier + ACPUp.

8. Set the Noise BW by either using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad. Noise BW is equal to the Resolution BW divided by the Vector Average factor.

9. Click on the Sweep Details... button to set the following:

- **Force RF Power OFF at the End of Sweep - System Preference** - If enabled, RF power is turned off at the end of a sweep.

- **Delay Before Start of Sweep** - Same as Sweep Delay in a standard channel.

- **Delay Before Distortion Measurement** - Adds delay after the linear S-parameter sweep and before the distortion measurement to allow the RF source to settle.

- **Enable S-Param Sweep** - Enables S-parameters to be measured at the low linear power level. If disabled (default), curve fitting (smoothing) is used instead of the S-parameter sweep. Curve fitting can be adjusted by varying the window aperture using the Distortion Measurement Correlation Aperture setting in the Measurement Details dialog.

- **S-Param Source** - Selects source and signal used to measure S-parameters.

- **External Vector - Chirp** - Selects a chirp signal from the external signal generator for measuring S-parameters.

- **S-Param Power** - Sets the power level used at the DUT input to measure the DUT gain when operating in its linear region. The linear gain is used to calculate distortion.
- **S-Param Freq Step** - Frequency step size setting used when measuring S-parameters.
- **S-Param IFBW** - Bandwidth setting when measuring S-parameters.
- **Re-use Previous S-Parameter Measurements If Available** - If disabled, S-parameters will be measured at the low linear power level. If enabled and compatible measurements exist from a previous sweep, those measurements will be used and a Linear S-parameter sweep will not be performed. If enabled and compatible measurements are not available, then a Linear S-parameter sweep will be performed.

19. Click on the **Power Sweep...** button to set up the **Power Sweep** dialog as follows:

![Power Sweep Dialog](image)

   a. Set **Start Power At** to -20 dBm.
   b. Set **Stop Power At** to 0 dBm.
   c. Set **Number of Powers** to 21.
   d. Set **Noise BW** to 520.833 Hz.
   e. Click **OK**.

20. Select the **Set Power At** pull down to set the power at the **DUT In**.

21. To set up a power supply, click on the **DC Sources...** button which accesses the standard dialog for controlling power supplies. Learn more.
22. Click on the **Apply** button to apply the setting changes made in this dialog.

23. Set up an EVM measurement as described in Set Up an EVM Measurement.

24. To view a trace displaying the equalized EVM of the DUT (non-linear contribution), perform the following:

   a. Press **Meas > Main > Other...** then click on the "..." button. The **Measure** dialog is displayed:

   ![Measure dialog](image1.png)

   b. Select the **Distortion** tab to display distortion traces:

   ![Distortion tab](image2.png)
c. Check **EVMDistEq21** then click on the **OK** button. This trace measures the EVM equalized distortion added by DUT (non-linear contribution).

25. Select **Sweep** > **Main** > **X-axis Type**... and ensure that the **X-axis Type** is set to **Power In**.

26. Click on the **OK** button.

27. The following is an example of a typical "Bathtub" trace and Distortion Table showing power sweep values:
<table>
<thead>
<tr>
<th>Power (dBm)</th>
<th>Type</th>
<th>Carrier</th>
<th>Overlap</th>
<th>Carrier</th>
<th>Overlap</th>
<th>EVM (2% PES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-15.00 dBm</td>
<td>CW</td>
<td>26.35</td>
<td>15.15</td>
<td>35.15</td>
<td>15.00</td>
<td>1.0%</td>
</tr>
<tr>
<td>-15.00 dBm</td>
<td>CW</td>
<td>26.45</td>
<td>15.45</td>
<td>35.45</td>
<td>15.00</td>
<td>1.0%</td>
</tr>
<tr>
<td>-15.00 dBm</td>
<td>CW</td>
<td>26.55</td>
<td>15.55</td>
<td>35.55</td>
<td>15.00</td>
<td>1.0%</td>
</tr>
<tr>
<td>-15.00 dBm</td>
<td>CW</td>
<td>26.65</td>
<td>15.65</td>
<td>35.65</td>
<td>15.00</td>
<td>1.0%</td>
</tr>
<tr>
<td>-15.00 dBm</td>
<td>CW</td>
<td>26.75</td>
<td>15.75</td>
<td>35.75</td>
<td>15.00</td>
<td>1.0%</td>
</tr>
<tr>
<td>-15.00 dBm</td>
<td>CW</td>
<td>26.85</td>
<td>15.85</td>
<td>35.85</td>
<td>15.00</td>
<td>1.0%</td>
</tr>
<tr>
<td>-15.00 dBm</td>
<td>CW</td>
<td>26.95</td>
<td>15.95</td>
<td>35.95</td>
<td>15.00</td>
<td>1.0%</td>
</tr>
<tr>
<td>-15.00 dBm</td>
<td>CW</td>
<td>27.05</td>
<td>16.05</td>
<td>36.05</td>
<td>15.00</td>
<td>1.0%</td>
</tr>
<tr>
<td>-15.00 dBm</td>
<td>CW</td>
<td>27.15</td>
<td>16.15</td>
<td>36.15</td>
<td>15.00</td>
<td>1.0%</td>
</tr>
<tr>
<td>-15.00 dBm</td>
<td>CW</td>
<td>27.25</td>
<td>16.25</td>
<td>36.25</td>
<td>15.00</td>
<td>1.0%</td>
</tr>
<tr>
<td>-15.00 dBm</td>
<td>CW</td>
<td>27.35</td>
<td>16.35</td>
<td>36.35</td>
<td>15.00</td>
<td>1.0%</td>
</tr>
<tr>
<td>-15.00 dBm</td>
<td>CW</td>
<td>27.45</td>
<td>16.45</td>
<td>36.45</td>
<td>15.00</td>
<td>1.0%</td>
</tr>
</tbody>
</table>
The Noise Figure Application makes fast, easy, and accurate noise figure measurements. This function is available with Opt 028, S9x029A/B, or S93027B.

The information presented in this topic pertains to Noise Figure measurements on BOTH Amplifiers and Converters unless stated otherwise.

- Noise Figure Hardware and Software Options Explained (028, S9x029A/B, S93027B, H29 - obsolete)
- Features, Requirements, and Limitations
- Noise Concepts
- How the Noise Figure Application Works
- Scalar Noise Figure Measurements
- PNA-X Option H29 - Block Diagram
- PNA-X Option 029 - Block Diagram
- Noise Figure Option S93027B
- The Noise Tuner Switch
- Noise Parameters that are Offered
- Using Noise Figure App
  - Connect Tuner and Noise Source
  - Create a Noise Figure Measurement
  - Make Noise Figure Settings
  - Perform Calibration (separate topic)
  - Save Noise Data
- Noise Figure Measurement Tips
- Using Noise Figure Traces in Equation Editor
- Noise Model and the Noise Correlation Matrix
Noise Figure Hardware and Software Options Explained

See Also: PNA-X Noise Figure Options (VNA Configuration Guide) - Internet connection required

- **029** - (PNA-X ONLY) Includes low-noise receivers and noise tuner bypass switch to enable noise figure measurements to 50 GHz. Also includes Opt 028 capability.

- **028** - Uses VNA receivers to measure noise figure. A noise source is NOT used during calibration. Any two ports can be used. Use with DUTs that have sufficiently high gain and noise figure. Additional filtering may be necessary. Learn more.

- **S93027B** - (PNA-X ONLY) Allows PNA-X models with Opt 029 to use mechanical noise tuners. Learn more.

- **S9x029A/B** - Noise figure measurement application. A Noise tuner and noise figure measurement of frequency translating devices are not supported.

- **H29 (obsolete)** - Noise Figure on N5244A (43.5 GHz PNA-X model) and N5245A (50 GHz PNA-X model). Includes both:
  
  - Opt 029 hardware from the N5242A (noise measurements to 26.5 GHz)
  

  **Note:** Option H29 does not include an internal noise tuner.

50 GHz Noise Figure Receivers

Beginning in October 2012, Option 029 (low-noise receivers) is available in the N5244B (43 GHz), N5245B (50 GHz), and N5247B (67 GHz) models.
• The low-noise receiver and the noise receiver path switch is added between the port 2 CPLR THRU jumper connector and the port 2 bias tee. See Option 029 block diagram below. When it is configured with a multiport test set (i.e. U3024AH10), the low-noise receiver cannot be used with test ports on the test set with standard multiport test set jumper connections. See modified test set connections to use 50 GHz low-noise receiver with multiport test set.

• These models (with option 029) include a built-in (internal) Noise Tuner which can be selected at the beginning of the Noise Figure Calibration. The Noise Tuner switch is managed differently than in 26.5 GHz models. Learn more.

• When used with the N5247A, Noise Figure measurements between 50 GHz and 67 GHz are possible using the standard receivers (Opt 028) and a 67 GHz Power Sensor. See Opt 028 Measurement tips.

• See limitations of the 50 GHz Noise Source.

• See Using Opt. 029 (low-noise receiver) with External Test Set

Noise Figure Application Features

• Cold noise method includes correction for imperfect system source match for highly accurate noise figure measurements.

• With Opt 029/S93029A/B you can measure devices with noise figure values ranging from about 0 to 50 dB and devices with GAIN ranging from about -40 to +60 dB. Learn more.

• With Opt 028/S93029A/B you can also measure noise figure using standard VNA receivers. Learn more.

• Measure noise figure of frequency translating devices. Learn more.

• During calibration, ENR values are interpolated for frequencies between the supplied data points.

Noise Figure Application Requirements

• Noise Tuner - Required for vector noise figure measurements. Not required for scalar noise figure.

  • ONLY the N4690 Series ECal modules are supported. The N4691B m-f is recommended.

  • Opt 029 provides an additional cable and adapter to connect the ECal module to the front-panel connectors. Learn more.

  • A built-in Noise Tuner is provided with the 50 GHz noise receiver models. Learn more.

• Power Meter - Required when calibrating NFX (Noise Figure on Converters).

• Recommended: An accurate thermometer. Learn more.
**Noise Source**

A Noise Source is NOT required to calibrate the Opt 029 Noise Receivers. Instead, the Noise Receivers can be calibrated using a calibrated VNA source. [Learn more.](#)

When using a Noise Source, the following requirements apply:

- The 346C Noise Source (recommended) produces ENR values to 26.5 GHz.
- The 346B Noise Source can be used up to 18 GHz.
- The 346A Noise Source can also be used up to 18 GHz, but requires more averaging for calibration.
- The 346C K01 (50 GHz) Noise Source typically has about 6 dB of ENR at 50 GHz which may NOT yield an adequate calibration, depending on how many noise averages are used. An alternative approach calibrates the noise receivers using a power sensor-based method. Select [Use Power Meter](#) for the noise figure calibration. [Learn more.](#)
- The U1831C Noise Source can be used up to 26.5 GHz.
- The U1832(A/B/C/D) and U1833(A/B/C/D) Noise Sources can be used.
  - A: 26.5GHz, B: 50GHz, C: 50GHz D: 55GHz
  - U1832X: Low ENR models
  - U1833X: High ENR models
- An adapter may be necessary to connect the Noise Source to the VNA port 2 reference plane during [calibration](#). Cal Kit (or second ECal module) with same connector type and gender as DUT connectors.

**Limitations with the Noise Figure Application**

The following features are NOT supported in a noise figure channel:

- [FCA or Frequency Offset](#)
- [Analog sweep](#). All frequency sweeps are STEPPED.
- Independent IFBW, Power Levels, or Sweep Time in a segment table is NOT supported.
- [External Test Set Control](Opt S93551A/B)
- Receiver calibration.
- Enhanced Response Cal
- ECAL User Characterization.
- Some Fixturing Features
- Auto Port Extensions
- Auto Formatted Citifile data.
- External DC Devices

- Pulsed noise figure measurements are supported with the following limitations:
  - Minimum 300 microsecond pulse width using 24 MHz noise bandwidth
  - Narrower noise bandwidths cause larger minimum pulse widths
  - A drop-out may occur at start of sweep and at 3 GHz. This is corrected by a 1 ms pulse width at 24 MHz Noise BW.

- Option 205 or Option 425

### Noise Concepts

The following conceptual information is a short summary taken from the Keysight Noise Figure App Note 57-1.

All electronic circuits have some degree of random noise. The most common form is thermal noise, which increases as the temperature of the circuit increases.

The signal-to-noise (S/N) ratio of components in a communications system is a very important parameter. To improve the S/N ratio, it is usually easier and more cost-effective to reduce noise than to increase signal power. In order to reduce noise, an accurate method to measure noise is required.

### Noise Figure

Noise Figure is the degradation in the signal-to-noise ratio as a signal passes through a device. For example, in the following images:
(a) At the INPUT of the DUT:
The noise floor is -100 dBm, the signal is at -60 dBm, 40 dB above the noise floor.

(b) At the OUTPUT of the same DUT:
The gain has boosted the signal AND the noise floor by 20 dB.
The DUT then added 10 dB of its own noise.
The output signal is now only 30 dB above the noise floor.
Since the degradation in signal-to-noise ratio is 10 dB, the DUT has a 10 dB noise figure.

For consistency, noise measurements are calculated as if using a 1 Hz bandwidth, although measurements are almost always made at higher bandwidths.

The following formula shows the lowest possible noise power in dBm at 290°K (room temperature). The only way to measure noise lower than this is to make the measurement at a lower temperature.

- \[ P = 10 \log(4.0 \times 10^{-21} \, \text{watts} / 0.001 \, \text{watt}) \]
- \[ P = -174 \, \text{dBm} / \text{Hz} \]

**How the Noise Figure Application Works**

The goal of the noise figure application is to accurately measure the noise that is generated by the DUT. This may be done using special low-noise receivers or using the standard VNA receivers depending on whether the VNA has Options 029 or 02. Learn more.
The standard receivers are always calibrated using a power meter and a measurement of the receivers effective noise bandwidth. The low-noise receivers can be calibrated using either a characterized noise source or using the same process as a standard VNA source. Learn more about the noise calibration process.

Some noise measurement error is caused by a poor source match presented to the DUT input. Therefore, during every measurement, the noise figure application uses an ECal module to present at least four different impedances at the input of the DUT. This "Noise Tuner" is connected to the VNA port 1 front-panel loops that are in the VNA internal source path (see block diagram below). From the measurements at various impedance states, the VNA calculates the noise out of the DUT as though the VNA were exactly 50 ohms. No assumptions are made regarding the input impedance of the DUT.

Here is how a vector noise figure measurement is made using Option 029. The sweep numbers are annotated on the VNA display as they occur.

1. With the noise tuner in the THRU state, S-parameter measurements are made to accurately characterize the gain of the DUT. This requires sweeps in both forward and reverse directions. (sweep #1 and #2).

2. The noise measurements are performed next. VNA source power is turned OFF and the noise tuner is switched to the first impedance state.

3. At each frequency, the noise receiver samples a large number of readings in order to attain one valid measurement. If Noise Averaging is selected, the specified number of measurements are made and averaged together to obtain one noise measurement. This continues for all frequencies (sweep #3).

4. The next noise tuner impedance state is switched IN and the noise measurements in step 3 are repeated. This occurs until measurements are made at all impedance states. At least four impedance states must be used. (sweeps #4, #5, #6+)

5. Calibration error terms are applied and calculations made to simulate the measurement with a perfect 50 ohm input impedance. The sweep result is plotted on the VNA display.

6. The VNA begins sweeping again with step 1.

**Scalar Noise Figure Measurements**

As described above, the noise tuner is switched to at least four different impedance states before a sweep is plotted. These sweeps are NOT made in a scalar noise figure measurement, resulting in much faster measurements. Of course, a scalar noise figure measurement is NOT as accurate as a vector noise figure measurement because scalar noise figure measurements assumes that all impedances are 50 ohms. Measurement accuracy can be improved by adding an attenuator as close to the DUT input as possible. This improves the effective system source match. The effect of the attenuator loss is removed during the calibration process.

With scalar noise figure, it is not necessary to connect the noise tuner. If a noise tuner remains connected, it is switched to the THRU state for scalar noise figure measurements. This results in a small
amount of loss which slightly degrades measurement accuracy. To increase measurement accuracy, manually switch the noise tuner switch to the INTERNAL position. Learn how.

Select Scalar Noise at the first page of a Noise Figure calibration.

50 GHz Noise Figure Components are shaded yellow

- At test port 1 front-panel loops, a noise tuner bypass switch connects the noise tuner (ECal module) in series with Source1 providing several different input impedances.

- At test port 2, a switch and coupler to route RF from the DUT output to two noise receivers. The appropriate receiver is automatically switched as required for the frequency being measured.
26.5 GHz Noise Figure Components are shaded yellow

- At test port 1 front-panel loops, a **noise tuner bypass switch** connects the noise tuner (ECal module) in series with Source 1 providing several different input impedances. Learn more about managing the Noise Tuner switch.

- At test port 2, a **switch** and **coupler** to route RF from the DUT output to **two noise receivers**. The appropriate receiver is automatically switched as required for the frequency being measured.
50 GHz Noise Figure Components are shaded yellow

- At test port 1, a **noise tuner bypass switch** connects the built-in noise tuner in series with Source1 providing several different input impedances. Learn more about managing the Noise Tuner switch.

- At test port 2, a **switch** routes RF from the DUT output to **two noise receivers**. The appropriate receiver is automatically switched as required for the frequency being measured. For using the noise receiver with an external test set, see 50 GHz Noise Figure with Opt. 029 and External Test Set.

**See Also:** 50 GHz Noise Figure - Built-in Tuner switch below.

---

**Noise Figure Option S93027B**

Option S93027B allows a PNA-X with a low-noise receiver to use specialized mechanical tuners on the input port. These tuners are designed to have a large number of impedance states that are broadly distributed on a Smith chart. There are two situations in which a mechanical tuner is advantageous:

1. When measuring noise figure at low frequencies. While Ecals are good general-purpose tuners, they tend to have a suboptimal spread of impedance states at low frequencies.
2. When measuring NF of a device that is poorly matched. Ecal modules are well suited to measure the noise figure of devices with a match near 50 ohms, but do not perform as well with devices that have match far from 50 ohms.

Mechanical tuners also improve the quality of the Noise Parameter measurements NFmin, GammaOpt, and Rho.

The following tuners are supported by Option S93027B. All are manufactured by Maury Microwave.
<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MT981AL14</td>
<td>LXI TUNER, 0.227-4.0 GHZ, 7MM</td>
</tr>
<tr>
<td>MT981BL10</td>
<td>LXI TUNER, 0.4-4.0 GHZ, 7MM</td>
</tr>
<tr>
<td>MT981BL15</td>
<td>LXI TUNER, 0.4-2.5 GHZ, 7MM</td>
</tr>
<tr>
<td>MT981BL18</td>
<td>LXI TUNER, 0.4-8.0 GHZ, 7MM</td>
</tr>
<tr>
<td>MT981EL10</td>
<td>LXI TUNER, 0.8-8.0 GHZ, 7MM</td>
</tr>
<tr>
<td>MT981HL13</td>
<td>LXI HGT, 0.8-8.0 GHz, 7MM</td>
</tr>
<tr>
<td>MT981HL14</td>
<td>LXI HGT, 1.8-8.0 GHz, 7MM</td>
</tr>
<tr>
<td>MT981HL15</td>
<td>LXI HGT, 0.65-6.0 GHz, 7MM</td>
</tr>
<tr>
<td>MT981VL10</td>
<td>LXI TUNER, 0.65-6.0 GHZ, 7MM</td>
</tr>
<tr>
<td>MT981WL40</td>
<td>LXI TUNER, 0.6-6.0 GHZ, 7MM</td>
</tr>
<tr>
<td>MT982AL02</td>
<td>LXI TUNER, 1.8-18.0 GHZ, 7MM</td>
</tr>
<tr>
<td>MT982BL01</td>
<td>LXI TUNER, 0.8-18.0GHZ, 7MM</td>
</tr>
<tr>
<td>MT982EL30</td>
<td>LXI TUNER, 0.8-8.0 GHZ, 7MM</td>
</tr>
<tr>
<td>MT982GL01</td>
<td>LXI TUNER, 0.65-18 GHZ, 7MM</td>
</tr>
<tr>
<td>MT982GL30</td>
<td>LXI TUNER, 0.65-8.0 GHZ, 7MM</td>
</tr>
<tr>
<td>MT983BL01</td>
<td>LXI TUNER, 2.0-26.5 GHZ, 3.5MM</td>
</tr>
<tr>
<td>MT984AL01</td>
<td>LXI TUNER, 8-50 GHz, 2.4MM</td>
</tr>
<tr>
<td>XT981AL14</td>
<td>LXI TUNER, 0.227-4.0 GHZ, 7MM</td>
</tr>
<tr>
<td>XT981BL10</td>
<td>LXI TUNER, 0.4-4.0 GHZ, 7MM</td>
</tr>
<tr>
<td>XT981BL18</td>
<td>LXI TUNER, 0.4-8.0 GHZ, 7MM</td>
</tr>
<tr>
<td>XT981HL13</td>
<td>LXI HGT, 0.8-8.0 GHz, 7MM</td>
</tr>
<tr>
<td>XT981HL14</td>
<td>LXI HGT, 1.8-8.0 GHz, 7MM</td>
</tr>
<tr>
<td>XT981VL10</td>
<td>LXI TUNER, 0.65-6.0 GHz, 7MM</td>
</tr>
<tr>
<td>XT982AL02</td>
<td>LXI TUNER, 1.8-18.0 GHZ, 7MM</td>
</tr>
<tr>
<td>XT982GL01</td>
<td>LXI TUNER, 0.6-18 GHZ, 7MM</td>
</tr>
<tr>
<td>XT982GL30</td>
<td>LXI TUNER, 0.6-8.0 GHZ, 7MM</td>
</tr>
<tr>
<td>XT983BL01</td>
<td>LXI TUNER, 2.0-26.5 GHZ, 3.5MM</td>
</tr>
</tbody>
</table>

To use a tuner supported by Option S93027B, follow this procedure:

1. Download driver files for the tuner at the Web address shown below:

2. Run the driver installation program.

3. Power up the tuner and connect it to the PNA via a USB cable.

4. The tuner should now be available for use. To verify this, start the PNA application and create a Noise Figure or Noise Figure Converters channel. Start the Calibration Wizard, and expand the selections for the Noise Tuner combo box. The attached tuner should appear in the displayed list. In the example below, a Maury Microwave MT982BL01 tuner was used.
5. Once the tuner has been recognized, it should be inserted into the signal path between port 1 and the DUT input. To increase measurement accuracy, it is best to have the tuner as close to the port 2 calibration plane as possible. This ensures that the spread of tuner impedances is as large as possible.

**Calibrating with a Mechanical Tuner**

No special steps are required to use a mechanical tuner for noise figure calibration. After verifying that the tuner is recognized by the PNA, simply perform a noise figure calibration in the usual way. The PNA firmware will measure all impedance states of the tuner during calibration. When the calibration is complete, the number of states used in vector NF correction can be set in the Noise Figure Setup dialog. In the example below, 15 states have been chosen, but up to 21 states are available.

**Caveats when using a Mechanical Tuner**

Mechanical tuners offer distinct advantages, but there are caveats to keep in mind:
Mechanical tuners are slow. They take more time to move from one impedance state to another than Ecal modules do. This increases the amount of time required for corrected NF measurements.

Because of the increased measurement time, interference can be more of an issue. Any external signals that can interfere with a noise figure measurement, such as cell phone traffic, have more opportunity to intrude. This is important to keep in mind if you are making lengthy measurements in an unshielded environment.

The number of sweeps taken during a vector corrected measurement is usually equal to the number of impedance states, plus two (for the forward and reverse S-parameter measurements). If a mechanical tuner is used, the number of sweeps may be about twice as large. This is because (depending upon frequency range) the tuner may need to take two sweeps per impedance state, one for low band, the other for high band.

If a Noise sweep is aborted for any reason, such as a change in stimulus conditions, the dialog below may appear. This is because the PNA is waiting for the tuner to complete an operation. The dialog should disappear after the pending operation is complete.

The Noise Tuner Switch while making S-parameter measurements

Because of the built-in Noise Tuner in the Option 029 50 GHz noise figure models, the Noise Tuner switch is managed differently than the 26.5 GHz noise figure models.

26.5 GHz Models

The default setting for the port 1 noise tuner switch is "External" as shown in the above diagram. This setting provides incident power through the front panel loops and the Noise Tuner when connected. When connected, the Noise Tuner may NOT be in the THRU state, which is necessary for accurate S-parameter measurements.

The switch is changed in any of the following ways:

- Set the switch to INTERNAL for the S-parameter channel using the path configuration dialog.
- Set the switch to INTERNAL for the S-parameter channel using the following commands:
  - SCPI - SENS:PATH:CONF:ELEM:STAT "Port1NoiseTuner", "Internal"
- COM - PathConfiguration.Element("Port1NoiseTuner").Value = "Internal"

- Set the switch default to INTERNAL using a preference setting.

- Set the Noise Tuner (ECal module) to the THRU state using SCPI: CONT:ECAL:MOD:PATH:STATE.

**Important Note:** On the 26.5 GHz Opt 029 models, once you set this switch to "Internal", you must set it back to "External" to make noise figure measurements. The switch is NOT automatically set to "Internal" during a noise figure measurement.

**50 GHz Models**

The switch for the built-in tuner (Opt. 029) is set to "Internal" (Tuner) ONLY when making vector noise figure measurements. Otherwise, it is set to "Bypass" (the tuner). Therefore, you should NOT need to make switch settings. However, the switch can be changed in any of the following ways:

- Using the path configuration dialog.

- Using the following commands:
  
  - SCPI - SENS:PATH:CONF:ELEM:STAT "Port1NoiseTuner", "Bypass" (or "Internal")
  
  - COM - PathConfiguration.Element("Port1NoiseTuner").Value = "Bypass" (or "Internal")

- Manage the built-in Noise Tuner impedance states using the SCPI and COM commands and Element / Value settings.

**Using the Noise Figure Application**

Use the following general procedure to make noise figure measurements:

1. Connect Tuner and Noise Source.

2. Create a Noise Figure Measurement.

3. Make Noise Figure Settings.

4. For Opt 029 and H29, copy your Noise Source ENR file to the VNA "C:\Program Files(x86)\Keysight\Network Analyzer\Noise folder"

5. Perform Calibration

6. Connect the DUT. Learn more about DUT input and output ports.

7. Measure Noise Figure.
8. Optional Click File, then Save to save noise figure data. Learn more.

**Connect Noise Tuner and Noise Source**

- Connect the noise source to the 28V connector on the VNA rear panel. **NOT required for Opt 028.** The Noise Source is turned ON and OFF automatically as needed during a calibration. Connect the noise source to Port 2 reference plane when prompted during calibration.

- Connect the noise tuner (ECal module). See Noise Tuner requirements.

  a. On the VNA front panel, remove the Port 1 jumper cable SOURCE OUT / CPLR THRU. Opt 028 allows noise figure measurements using any two VNA ports.

  b. Connect the noise tuner to the front-panel jumpers for the source (DUT input) port.

See the VNA Configuration Guide for recommended ECal modules, cables, and adapters.

**Create a Noise Figure Measurement**

1. On the VNA front panel, press Meas > S-Param > Meas Class....

2. Select Noise Figure Cold Source, then either:

   - OK delete the existing measurement, or
   - New Channel to create the measurement in a new channel.

3. A noise figure measurement is displayed. The following shows how to select or change displayed parameters.

**Noise Parameters**
Several noise parameters, as well as standard parameters, can be measured in the same Noise channel.

### How to add Noise Parameters

1. Create a Noise Figure channel.
2. Then do the following:

**Using Hardkey/SoftTab/Softkey**

1. Press **Trace > Trace N > Trace N**.
2. Press **Trace > Trace Setup > Measure...**

**Using a mouse**

1. Click **Instrument**
2. Select **Trace**
3. Select **Add Trace**
4. Click **Instrument**
5. Select **Trace**
6. Select **Measure...**

### How to CHANGE Noise Parameters

1. Create a Noise Figure channel.
2. Select the parameter to change.
3. Then do the following:

1. Select a trace by pressing **Trace > Trace N > Trace N**.
2. Press **Trace > Trace Setup > Measure...**
3. Select a parameter.

### Programming Commands

#### Noise Measurements that are offered

The following three categories of noise measurements can be made with the VNA:

1. **Noise Figure** is the amount of noise that the DUT is adding in a 50 ohm test setup. This is explained in detail in Noise Concepts.

2. **Noise Power Parameters** show the amount of noise coming out of the DUT in a 50 ohm test setup. With gain
measurements of the DUT, these noise power parameters are used to calculate noise figure.

3. **Noise Parameters** are models of the noise that is generated in a DUT, similar to how S-parameters model how RF flows through a DUT.

- **Noise Figure (NF)** - Explained in Noise concepts.
- **Excess Noise Ratio** - Select when measuring the noise source. Compare with the ENR table to validate accuracy of the system. ENR is calculated as:

\[
\text{ENR (in dB) = } 10 \log_{10} \left( \frac{T_{\text{hot}} - T_{\text{cold}}}{T_0} \right), \text{ where } T_0 = 290K.
\]

Learn more about the ENR table and Noise Source. Learn more about Noise Source ENR measurements.

- **T-Effective** - The effective temperature, in Kelvin, of the measured noise level. For example:

\[
290^\circ K = -174 \text{ dBm/Hz}.
\]

Available Gain \( G_a \) is a function of \( S_{11}, S_{22}, \) and \( \Gamma_s \)

**Noise Power Parameters**

The Noise Power parameters below are offered in the following two formats:

- **Available Noise Power**  The calculated power that is based on an ideal impedance match at the output of the DUT. These parameters have always been offered in the VNA noise figure App.

- **Incident Noise Power**  An 'I' is appended to the end of the Available Noise Power parameter. The calculated power into a perfect 50 ohm noise receiver, regardless of the output impedance of the DUT.

- **SYSNP / SYSNPD** - System Noise Power Density: Total noise power available at the ADC, including the noise contributed by both the DUT and the internal noise receiver. This is generally expressed as an
absolute power measurement in dBm, but can also be expressed in Watts or Kelvin.

\[ \text{dBm} = 10 \log_{10}(k \times T \times B \times 1000) \]

where:
- \( k \) = Boltzmann's constant
- \( T \) = the measured noise temperature
- \( B \) = bandwidth
- \( 1000 \) = conversion from milliwatts

- **SYSRNP / SYSRNPI** - System Relative Noise Power: The noise temperature of the combined DUT and receiver relative to 290 Kelvin. This is generally reported as a ratio in dB. Therefore a perfectly quiet device would render a trace at 0 dB.

\[ \text{dB} = 10 \log_{10}(T/290) \]

- **DUTNPD / DUTNPDI** - DUT Noise Power Density: When correction is ON, this trace exhibits the available noise power, best described as the maximum power available from the DUT where the impedance of the noise port is equal to the output match of the DUT. To be more precise, this occurs when the noise port match is equal to the conjugate of the output match of the DUT. The noise power contributed by the receiver is removed.

When correction is OFF, the trace exhibits what is more accurately described as delivered power. Delivered power is the power actually seen by the ADC. Any mismatch between the receiver and the DUT is ignored. The noise power contributed by the receiver is removed.

This measurement is generally expressed in dBm, normalized to a 1 Hz bandwidth. For convenience, marker and trace readout shows **dBm**.

You could display the power in a different bandwidth using Equation Editor.

\[ \text{dBm/Hz} = 10 \log_{10}( (\text{DUT Temperature} - \text{Receiver Temperature}) \times B \times 1000) \]

where:
- \( B \) = bandwidth
- \( 1000 \) = conversion from milliwatts

- **DUTRNP / DUTRNPI** - DUT Relative Noise Power: This measurement is rendered as a ratio of the DUT temperature to 290 Kelvin. It is generally expressed in dB. The same comments apply with respect to available versus delivered power as described above for DUTNPD.

\[ \text{dB} = 10 \log_{10} (\text{DUT Temperature} - \text{Receiver Temperature}) \]
Noise Model, Noise Parameters, and the Noise Correlation Matrix

Noise Parameters are models of the noise that is generated in a DUT, similar to how S-parameters model how RF flows through a DUT.

Noise Model

The noise wave model of any linear 2-port network may be represented by the following image:

This shows a noiseless 2-port network with noise waves ($a_{n1}$ and $b_{n1}$) added to the input terminals. The $a_1$, $a_2$, $b_1$, and $b_2$ are standard S-parameter waves.

The noise correlation matrix relates to the noise waves as follows:

\[
C_\mathbf{c} = \begin{bmatrix}
|a_{n1}|^2 & a_{n1}b_{n1}^* \\
\overline{a_{n1}} & |b_{n1}|^2
\end{bmatrix} = \begin{bmatrix}
c_{11} & c_{12} \\
c_{21} & c_{22}
\end{bmatrix}
\]

Where:

- $|a_{n1}|^2$ and $|b_{n1}|^2$ are time-averaged noise power in 1 Hz bandwidth.
- $a_{n1}b_{n1}^*$ and $\overline{a_{n1}}a_{n1}$ are time-averaged cross correlation terms, correlation of $a_{n1}$ to $b_{n1}$.
- Overbars represent time-averaging
- Star superscripts represent complex conjugation

Noise Parameters

- **GammaOpt** (Optimum Complex Reflection Coefficient) - The optimal impedance for the noise figure measurement. Select the data format to display GammaOpt in terms of Log Mag, Lin Mag, Phase, Unwrapped Phase, Real, Imaginary, Polar, or on a Smith Chart.
- **NF_{min}** - The minimum noise figure that occurs at GammaOpt. NF_{min} is a scalar quantity that can be displayed as Log Mag, Lin Mag, or Real.
- **R_n** (Noise Resistance) - Specifies the rate of change of the level of noise when varying the source impedance. R_n is a scalar quantity in units of ohms that should be displayed in Lin Mag or Real format.
Note: Rn, as a measurement parameter, is not normalized. When the value of Rn is written to an S2P file, it is normalized (Rn / Zo). When an S2P file is recalled, the normalized Rn is multiplied by the system impedance to obtain Rn.

- **NCorr_11, NCorr_21, NCorr_12, NCorr_22** - The NCorr_11 and NCorr_22 terms are effective noise temperature, normalized to 290 K. Both terms are time-averaged, noise-wave powers referred to the input of the DUT, where NCorr_11 is the forward wave (noise going through the device towards the output), and NCorr_22 is the reverse noise wave (noise coming out of port 1 of the DUT, going back towards the source).

  To convert to available noise power, multiply the terms by 290*k*B where:

  - k = Boltzmanns constant
  - B = system bandwidth

---

**Standard Parameters that are offered** (Amplifiers-only)

- **S-parameters**: S11, S21, S22, S12
- **Unratioed parameters** using the following notation: (Receiver, source port). These parameters REPLACE the active measurement. To do this (from front-panel ONLY), press **Meas**, then **More**, then **Receivers**.

  - (R1,1), (R2,2), (A,1), (A,2), (B,1), (B,2)

---

**Save Noise Data**

To save noise data, click **File**, then **Save Data As**. Then select from the following **Save As Types**:

- (**.prn**), (**.cti**), (**.csv**), (**.mdf**) - Noise Figure data can be saved ONLY with these choices. PRN saves only the active trace. CITI formatted, CSV Formatted, and MDF can save all displayed traces. Learn more about these formats.

- (**.s2p**) - Saves S-parameter data only after performing a Noise calibration. This data is saved regardless of which noise measurement is active or displayed. Learn more about **.s2p** data.

**Trace and Noise parameter (**.s2p**)** - Saves S-parameter data, then the Noise Parameters. This data is saved regardless of which noise measurement is active or displayed. When the vector calibration is not enabled or if the noise parameters are not realizable, the noise parameters have no calculated value. In this instance, the following values are displayed instead:

  - **GammaOpt** = 0
  - **NFmin** = 50 ohm noise figure
  - **Rn** = Zo / 4 * (F - 1). This equation is how Rn is currently calculated for ill-conditioned data. F is the
noise factor where \( F \) is related to the noise correlation value \( c_{11} \) and the normalized noise temperature \( T_n \) by \( F = 1 + c_{11} = 1 + T_n \) so that \( R_n = \left( \frac{Z_0}{4} \right) \times c_{11} \)

- **NoiseCorr (*.nco)** - Saves Noise Correlation data regardless of which noise measurement is active or displayed. The *nco* file is a noise correlation matrix expressed in T-parameter form \( (C_{t11}, C_{t21}, C_{t12}, C_{t22}) \). These parameters are exactly the same as the Noise parameters \( NCorr_{11}, NCorr_{21}, NCorr_{12}, NCorr_{22} \) that can be displayed as traces.

- When the vector calibration is not enabled, this data is set to -200 dBm.

**How to start the Noise Figure Setup dialog**

**Using Hardkey/SoftTab/Softkey**

1. **Freq > Main > NF Setup....**

**Using a mouse**

1. Click **Stimulus**
2. Select **NF Setup...**

**Noise Figure Setup** dialog box help

**Note:** In this topic, the term **Jitter** is used to describe the trace-to-trace fluctuations in a measurement. In other topics, this is called 'trace noise'.

**Bandwidth/Average**

The following settings work together to achieve the optimum balance of measurement accuracy versus speed:

- **Noise Bandwidth**  Increase the bandwidth to reduce the amount of trace noise on the noise power or noise figure measurement (jitter). However, a wider setting reduces the frequency resolution of the measurement. The noise bandwidth setting should always be smaller than the bandwidth of the DUT. The noise bandwidth setting is used only while measuring noise powers, and is independent from the **IF bandwidth setting** used to measure S-parameters. Noise figure is
The calibration and measurement should be performed using the SAME noise bandwidth. When the noise bandwidth is changed after calibration, noise figure measurements can change by 0.5 dB or more, depending on the DUT frequency range, gain, and noise figure.

**Note:** The Noise Bandwidth may be adjusted automatically at low frequencies according to the following table. At each data point frequency, if the specified Noise BW is higher than that shown in the table, the Noise BW is set to the max value in the table.

<table>
<thead>
<tr>
<th>RF Bands</th>
<th>10 to 25 MHz</th>
<th>25 to 60 MHz</th>
<th>60 to 150 MHz</th>
<th>Above 150 MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>.8 MHz</td>
<td>.8 MHz</td>
<td>.8 MHz</td>
<td>.8 MHz</td>
<td>.8 MHz</td>
</tr>
<tr>
<td>2 MHz</td>
<td>2 MHz</td>
<td>2 MHz</td>
<td>2 MHz</td>
<td>2 MHz</td>
</tr>
<tr>
<td>4 MHz</td>
<td>4 MHz</td>
<td>4 MHz</td>
<td>4 MHz</td>
<td>8 MHz</td>
</tr>
<tr>
<td>8 MHz</td>
<td>8 MHz</td>
<td>8 MHz</td>
<td>24 MHz</td>
<td>24 MHz</td>
</tr>
<tr>
<td>24 MHz</td>
<td>24 MHz</td>
<td>24 MHz</td>
<td>24 MHz</td>
<td>36 MHz</td>
</tr>
</tbody>
</table>

**Note:** Use Power Meter calibration method is NOT available when the Noise Bandwidth is 8 MHz or 24 MHz.

**Average Number** Increase the number of averages to reduce jitter. This also reduces measurement speed. For maximum accuracy, use the following recommendations for the noise calibration. When using the noise receivers, 10 noise averages is recommended. When using the standard receivers, at least 100 averages are recommended.

During a measurement, the gain of the DUT helps overcome the noise of the VNA receivers, so the number of noise averages can be reduced to improve measurement speed with minimal or no degradation to measurement accuracy.

**Use Narrowband Compensation**

The mathematics of noise figure assumes that the gain of the DUT is constant over the bandwidth of the noise receiver. The following image illustrates a case in which the gain (S21) of the DUT falls off sharply outside the passband region. When the VNA measures noise figure at the frequency indicated by the solid vertical line using a 4 MHz noise bandwidth, standard noise figure calculations assume the gain to equal its midpoint value (dashed horizontal line) over the
entire 4 MHz bandwidth. This assumption yields a composite gain-bandwidth value that is lower than the actual value, which in turn results in a noise figure value that is too high. This is the reason for the bump in the displayed NF value at this frequency and surrounding frequencies.

In the following image, Narrowband Compensation combines DUT measurements with characteristics of the noise receiver, which accommodates changes in DUT gain over the receiver bandwidth. The result is a better gain-bandwidth value of the system. Notice how the peaks and valleys of the NF measurement disappear when narrowband compensation is applied.

Notes on using Narrowband Compensation:
Can be used with either option 028 (Noise figure measurements using standard receivers) or 029 (Fully Corrected Noise Figure).

- With option 029 (NF receiver) Narrowband Compensation is available only for the 800 kHz, 2 MHz, and 4 MHz noise bandwidths.
- With option 028 (Standard VNA receivers) you **MUST** re-configure the front panel loops. Learn how.

Can be used with both Scalar and Vector NF calibrations, on either NF or NFX channels.

- The ON / Off state has no effect on calibration. In other words, it does not matter if Narrowband Compensation is On or Off while a noise calibration is being performed.
- Is applied only when corrected DUT measurements are made. If correction is turned off, it has no effect.
- Can be ON or Off while the NF channel is in Hold mode, and it will modify the NF trace appropriately. There is no need to re-sweep.

**Noise Receiver**

**NA (Network Analyzer) Receiver** (Opt 028) - Use a standard VNA receiver to measure noise figure.

- Connect the DUT to any VNA ports. For vector noise figure measurements, connect the noise tuner to the source port.
- The gain plus noise figure of the DUT minus cable loss must be at least 40 dB (**G+ NF - Loss > 40 dB**). This ensures that there is sufficient DUT noise power for the VNA to measure. Learn more.
- Additional filtering may be required. Learn more.

**Noise Receiver** (Opt 029) - Use internal low-noise receivers to measure noise figure.

- Opt 029 measures devices with noise figure values ranging from about 0 to 50 dB and devices with GAIN ranging from about -40 to +60 dB. Learn more.
- Amplifiers with higher gain can be measured by adding an attenuator to the output of DUT and using **fixture de-embedding** to remove the attenuator loss. An alternative for measuring high-gain devices is to use the standard receivers (Opt 028) as they have a higher compression level.

**Receiver Gain**

This setting is NOT available when Noise Receiver is set to **NA (Network Analyzer) Receiver** (Opt 028).

With knowledge of your DUT gain, set the appropriate amount of receiver gain in order to optimize
the power level at the noise receiver.

The following values reflect the SUM of the DUT gain (dB) **PLUS** NF (dB). For example: DUT gain = 20 dB; NF = 10 dB; SUM = 30 dB.

- Select **High** if the SUM is relatively low (<30 dB).
- Select **Medium** if the SUM is about average (20 dB to 45 dB).
- Select **Low** if the SUM is relatively high (>35 dB).

There is considerable overlap in these settings. Because all three gain settings are calibrated with each Noise Calibration, this setting can be changed after calibration to achieve the least amount of jitter without overpowering the noise receiver.

One of following messages appears when too much power is detected at the noise receiver:

- **Compression in noise receiver: excess signal** - The noise receiver is likely compressing. NF results are possibly not accurate. Select a lower gain setting.
- **Compression in noise receiver: gain has been limited** - The gain has been limited to avoid damage to the receiver. NF results are NOT accurate. Select a lower gain setting.
- **ADC over-range in noise receiver: excess signal** - Often caused by a CW signal, an oscillation, or LO feedthru during an NF measurement. Find and correct the cause, or try a lower gain setting.

Only ONE gain setting can be used for the entire frequency range of your noise measurement. Therefore, it may be necessary to use two noise channels with different frequency ranges and gain settings to achieve the very highest noise figure accuracy.

**Ambient Temperature**

**Note:** This setting is only used for calibrated noise figure measurements, but has no effect in an uncalibrated noise figure channel. The default value is used for uncalibrated measurements.

Enter the equivalent port 1 temperature at the time of the measurement, in Kelvin (K). One can use a thermometer to measure the temperature of the input cable.

In the case of full vector correction, it is the temperature of the Ecal Tuner (31 °C or 304.15K) minus the loss effect of the cable from the tuner to the DUT; both internal and external Ecals used as tuners have the same internal heater to heat to 31 °C.

For scalar it is the temperature of the internal load (such as the internal step attenuator) minus the loss of the cable, typically around 297K.
The cable loss compensation is computed from \(\text{Tambient} = T_{\text{pna\_source}} \cdot |S21|^2 + (1-|S21|^2) \cdot T_{\text{cable}}\) where \(S21\) is the loss of the port 1 cable, \(T_{\text{cable}}\) is the temperature of the cable, and \(T_{\text{pna\_source}}\) is the temperature of the either the Ecal used as a tuner, for full vector, or the temperature of the internal load for scalar calibration. \(T_{\text{ecal}}\) is typically 304.15K; the internal load is typically around 297K (if the attenuator internally is set to 10 dB or more) or 303K if the internal source attenuator is set to 0. The attenuator is physically located near the input of the air flow and so it is very close to the external ambient temperature, but with 0 dB setting, the temperature of the internal source becomes the effective input temperature and it is a little warmer at about 6 degrees rise above ambient.

This ambient temperature number has an inverse relationship to the noise figure. When using the effective noise temperature (\(Te\)) format, a 3 degree increase in the ambient temperature will make the calibration measurement result drop 3 degrees, which will then have an effect on subsequent noise figure measurements. One can directly measure the port 1 equivalent temperature by connecting port 1 to port 2 with a low loss through, and measuring the mean value directly. Because the noise value is quite low, averaging or using trace statistics should be used to find this value.

**Impedance States**

**Noise Tuner** Displays the ECal module to be used as a noise tuner. Select the Noise Tuner during calibration on the Select Cal Method dialog.

**Max Acquired Impedance States** Select the number of impedance states in which to make noise measurements. At least FOUR impedance states are required. Learn more.

**Frequency Tab - Noise Figure** dialog box help

These settings can also be made from the normal VNA setting locations. Click links below to learn...
how.

**Sweep Type**

Choose a sweep type. [Learn more.]

**Segment Sweep Notes:**

- The segment table shown on the dialog is *'READ-ONLY'*. 
- Learn how to [Create and edit the Segment Sweep table.]
- **Independent IFBW** and **Power** are NOT available.
- **X-axis point spacing** is available beginning with A.09.10.

**Sweep Settings**

Click each to learn more about these settings.

- **Number of points**
- **IF Bandwidth**. This setting is important for improving noise measurement accuracy. [Learn more.]
- **Start / Stop, Center / Span** frequencies.

---

**Power Tab - Noise Figure** dialog box help

![Power Tab - Noise Figure](image)

**Note:** S-parameter power settings are critical for accurate noise figure measurements. [See Noise Figure Measurement Tips.]

Configures RF power settings for the S-parameter measurements that occur before noise measurements. Input power to the DUT is turned OFF during noise measurements.

These settings can also be made from the normal **Power setting** locations.
**Power ON (All channels)**  Check to turn RF Power ON for all channels.

**DUT Input Port**

- Opt 028 - Select a VNA port to be connected to the DUT input.
- Opt 029 Scalar Noise Figure - Select a VNA port other than port 2.
- Opt 029 Vector Noise Figure - The DUT input CAN be connected to any VNA port other than port 2. However, without a noise tuner bypass switch, measurements on other channels that use the same source port will always go through the noise tuner. The noise tuner must be connected to the source loop of the selected port.

**Note:** Input power levels are critical for accurate noise figure measurements. [Learn more.](#)

**Power Level**  The input power to the DUT during S-parameter measurements.

**Source Attenuator Auto**  Check to automatically select the correct attenuation to achieve the specified input power. Clear, then select attenuator setting that is used achieve the specified Power Level. [Learn more about Source Attenuation.](#)

All VNA channels in continuous sweep must have the same attenuation value. [Learn more.](#)

**Receiver Attenuator**  Specifies the receiver attenuator setting for input port.

**Source Leveling**  Specifies the leveling mode. Choose Internal.  Open Loop should only be used when doing Wideband Pulse measurements (not available with Noise figure measurements).

**DUT Output Port**

- Opt 028 - Select a VNA port to be connected to the DUT output.
- Opt 029 - Connect the DUT output to VNA port 2.

**Output Power**  Sets power level in to the output port for reverse sweeps. Port power is automatically uncoupled. Reverse sweeps are always applied to the DUT when Full 2-port correction is applied. [Enhanced Response Cal](#) is NOT available for noise figure measurements.

**Source Attenuator**  Specifies the source attenuator setting for reverse power.

**Receiver Attenuator**  Specifies the receiver attenuator setting for the output port. [Learn more about Receiver Attenuation.](#)

**Source Leveling**  Specifies the leveling mode. Choose Internal.
26.5 GHz Models

The orange line between CPLR THRU and SRC OUT represents the Noise Tuner.

50 GHz Models (Opt. 029)

Port 1 Noise Tuner Switch (Opt 029)

26.5 GHz

The orange line between CPLR THRU and SRC OUT represents the Noise Tuner.
**External** selects the external Noise Tuner for making noise figure measurements.

**Internal** bypasses the external Noise Tuner.

See **Important Notes** about managing this switch.

### 50 GHz Models

- Tuner - Represents the built-in Noise tuner.
- Bypass - Bypasses the built-in tuner

**Port 2 Noise Receiver Switch (Opt 029 All models)** allows you to make Noise Receiver measurements.

To prevent premature wear on the above two Noise switches, the VNA does not allow these switches to be thrown when sweeping a Noise channel and non-Noise channel. To make noise figure measurements and non-noise figure measurements in different channels and continuously trigger both, set these switches to the same state as the Noise channel:

- With the **non-noise figure channel** active, go to Noise Path Configurator.

- Set Noise Tuner switch to **External**. This routes source power to the front-panel loops, and to the Noise Tuner when connected. Use **CONT:ECAL:MOD:PATH:STATE** to set the internal state of the Noise Tuner to THRU, which creates a small amount of additional loss in the source path.

- Set Noise Receiver Switch to **Noise Receiver**.

### Noise Figure Measurement Tips

**Note:** In this topic, the term **Jitter** is used to describe the trace-to-trace fluctuations in a measurement. In other topics, this is called 'trace noise'.

**Option 029**

See **Opt 028 (NF with Standard Receiver)**

- Measures devices with noise figure values ranging from about 0 to 50 dB and devices with GAIN ranging from about -40 to +60 dB.

- Highest noise figure accuracy is attained when the sum of device noise figure + GAIN is between 0 dB to +70 dB.

- For highest noise figure accuracy and stability, there should be the least amount of electrical loss possible.
between the DUT output and VNA Port 2.

**Power level at the DUT Output - Opt 029**

S-parameters are used to measure the gain of the DUT before each series of noise measurements. Jitter in the S-parameter measurements corresponds directly to jitter in the noise measurements.

For best noise figure accuracy, the power level out of the DUT should be between 15 dB and 20 dB below the compression point of the DUT during the S-parameter portion of the noise figure measurement.

To reduce jitter, the power level at the B receiver (port 2) should be above approximately -20 dBm. Much below this level, S-parameter measurements have more jitter. Power must be below +10 dBm as the B receiver starts to compress at this point, although there is no warning or annotation that shows this condition is occurring in S-parameter measurements.

The best way to monitor power at the B receiver is to display a B,1 measurement. With your DUT in place and powered ON, change the input power to the device and note the power at the B receiver.

- For low-gain DUTs, use 5 dB of source attenuation to improve the uncorrected match of port 1.
- For high-gain DUTs, source and receiver attenuation may be required. Use the lowest possible attenuation values.

Attaining the optimum power level during calibration can also be challenging since calibration is performed without the DUT in place. Because of this, it is often necessary to set source power higher during the calibration than during the measurement. This will cause the CΔ annotation on the status bar.

Measurement results are accurate as long as the step attenuators and other configuration switches are in the same position and all receivers remain in their linear range (below +10 dBm).

It is best to find the optimum power and attenuation settings for both the calibration and subsequent noise measurements before performing a calibration.

**IF Bandwidth**

Jitter is further reduced by narrowing the IF bandwidth. If the calibration needs to be performed at a low source power, or with receiver attenuation due to high DUT gain, the IF bandwidth should be reduced during the calibration to reduce jitter. The IF bandwidth can then be increased to improve measurement speed. The CΔ annotation can be ignored when changing IFBW after calibration.

**Noise Settings**

See Noise Figure dialog box help for a complete description of these important settings.

**Temperature**

6177
Noise Figure measurements are extremely sensitive to temperature. As such, there are two settings that require an accurate temperature measurement: At the DUT input, and at the Noise Source connector.

### Interference

When measuring the noise figure of an unshielded device, like an amplifier on a printed-circuit board, it is very common to pick up interference from external signals such as cellular phones, wireless LAN, or mobile radios. This interference shows up as non-repeatable spikes in the measurement, as shown below.

![Interference Graph]

Usually, the interference adversely affects the noise figure measurement only at the frequency where it occurs. However, if the interference is large enough and present all of the time, it can cause the noise receivers to compress, which results in inaccurate measurements at many frequencies. In this case, the noise figure measurements should be done in a shielded environment like a screen room.

### Option 028

**Noise Figure of VNA receiver** - Option 028 gives you the flexibility to measure noise figure using a standard VNA receiver. For best measurement accuracy, the DUT excess noise power, which is gain plus noise figure minus cable loss in dB (G + NF - Loss), should meet or exceed the noise figure of the receiver. This is generally not a problem with very high-gain devices such as converters with approximately 60 dB of gain.

If your DUT is NOT a very high-gain device, you can re-configure the VNA front panel loops to increase receiver sensitivity.

**Re-configuring the front panel loops** - This configuration reverses the main arm and coupled arm of the test-port coupler (see following images). This increases the signal to the receiver port by about 15 dB, while lowering the available port power by the same amount. This is a good tradeoff for noise figure measurements.
Configure the **receiver port** front-panel loops to a vertical orientation as shown here.

The following table shows the excess noise that is **required** at various frequencies. These values assume the front-panel loops have been re-configured as shown above:

<table>
<thead>
<tr>
<th>Frequency range</th>
<th>Required Excess Noise Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up to 20 GHz:</td>
<td>30 dB</td>
</tr>
<tr>
<td>Up to 50 GHz:</td>
<td>40 dB</td>
</tr>
<tr>
<td>Up to 67 GHz:</td>
<td>45 dB</td>
</tr>
</tbody>
</table>

For devices that do NOT meet this requirement, a low-noise amplifier (LNA) must be added to the receiver loop (see following image). This boosts the noise power at the receiver by the gain of the LNA. The disadvantage is the possibility of measurement drift and receiver compression. Any change in the gain of the LNA will have an impact on measurements that use the receiver with the LNA, so frequent calibration may be required. Care should also be taken when setting the channel power (used during the S-parameter portion of the calibration, and the gain portion of the DUT measurement) to ensure that the added gain of the LNA does not cause receiver damage or compression. A filter is also required on the output of the LNA. Learn more.
Filtering Requirement (Option 028)

Opt 029 includes noise receivers with filtering to keep mixing-product noise out of the low-noise receivers. These filters are not available when measuring noise with the standard VNA receiver. Therefore, for best measurement accuracy, a filter should be used at the output of the DUT (or LNA preamp if used).

- A bandpass filter at the frequencies of interest can always be used.
- A lowpass filter can be used when the VNA is doing fundamental mixing (up to 26.5 GHz). The lowpass filter must pass the fundamental frequency of the measurement but suppress the third harmonic. A measurement at 1 GHz would need a lowpass filter with a cutoff below 3 GHz, while a 5 GHz measurement would need a filter with a cutoff below 15 GHz.
- A single highpass filter can often be used when the VNA is doing 3rd-harmonic mixing (from 26.5 to 67 GHz). Use a highpass filter with cutoff about 18 GHz for frequencies up to 50 GHz. For operation to 67 GHz, the filter cutoff would need to be above 23 GHz.

Using Noise Figure Traces in Equation Editor

In a Noise Power trace, the underlying unit is noise temperature.

\[ 10 \log_{10} \left( \text{temperature} \times 1000 \text{mw/w} \times 1.38 \times 10^{-23} \right) \]

\((1.38 \times 10^{-23} \text{ is Boltzmann's constant})\)

Any time you use Equation Editor on a Noise Power trace, the LogMag formatting will apply the above equation. Therefore, first select REAL format and then generate the equation.
The following screen is an example showing three traces: DUTNPD (DUT Noise Power Density), NF (Noise Figure), and S11 with the equation set to "***=10". Note that formatting for noise figure measurements is different than noise power measurements or temperature measurements.

**Radio-Frequency Electromagnetic Field Immunity**

When a 3Vm-1 radio-frequency electromagnetic field is applied to an PNA with Opt 029 according to IEC 61000-4-3:1995, degradation of performance may be observed. When the frequency of the incident field matches the frequency of a measured noise figure or gain, the values displayed will deviate from those expected. This phenomenon will only affect that specific frequency, and the analyzer will continue to perform to the specification at all other frequency sample points.

The VNA with Opt 029 may be unable to calibrate a chosen frequency sample point if the frequency matches that of an incident electromagnetic field.
This topic discusses Noise Figure on Converters:

- Requirements and Limitations
- How to Configure your Hardware
- Create a NFX Measurement
- NFX Parameters
- Valid Mixer Configuration / Sweep Type Combinations
  - Frequency tab
  - Power tab
  - Noise Figure Setup tab
  - Mixer Frequency tab (separate topic)
  - Mixer Power tab (separate topic)
  - Mixer Setup tab (separate topic)

The following general information (contained in Noise Figure Application for Amplifiers) is also relevant for NFX measurements:

- Noise Figure Hardware and Software Options Explained (029, 028, S93029A/B, H29)
- Features and Requirements
- Noise Concepts
- How the Noise Figure Application Works
- Scalar Noise Figure Measurements
- Perform Calibration
- Noise Figure Measurement Tips
- Noise Model and the Noise Correlation Matrix
Other Application Topics

Requirements and Limitations

Noise Figure on Converters requires Noise Figure (Option 0pt 028 or 029 and S93029A/B).

Learn more about Noise Figure requirements.

Limitations

- Upconverters OR downconverters ONLY (not both).
- Image rejection is NOT provided in NFX.

The following VNA features are NOT available with NFX:

- Analog Sweep (Stepped sweep mode only)
- Independent IFBW, Power Levels, or Sweep Time in a segment table is NOT supported.
- Log frequency sweeps
- ECaL User Characterization
- Time Domain
- Balanced measurements
- Save Formatted Citifile data.
- Some Fixturing Features
- External Test Set Control (Option S93551A/B)
- External DC Devices
- Pulse measurements
- See Frequency Limitations

- Option 205 or Option 425

- Segmented sweeps (an alternative is to set up a separate channel for each frequency band)

Embedded LO measurements ARE supported in NFX. Learn more.
How to Configure your Hardware

The PNA-X is extremely versatile, and can be configured in many ways to make NFX measurements. While not all conceivable configurations are documented here, a few of the most common examples are provided to show the basic concepts.

DUT Configuration

Learn more about connecting the DUT input and output to the VNA.

The DUT LO can be connected to an external source OR VNA internal second source (if available).

Select an LO Source on the Mixer Setup tab.

Note: Noise that is present on the LO source will be directly transferred to the DUT output. This noise can NOT be calibrated out of the noise measurement. Therefore, choose a low-noise source for the LO, such as an Keysight ESG or PSG, which is better than the VNA internal source.

External LO Source though Port 3 or Port 4 (4-port PNA-X only)

- Connect the DUT LO to PNA-X Port 3 or Port 4.
- Connect external source to rear-panel J7 (Rear-panel load on below diagram) for Port 3; J3 for Port 4.
  - For Port 3, NO switching is required.
  - For Port 4, switch Port 4 Bypass Switch to Rear Panel.

PNA-X Internal LO Source

- SRC 2 Out 1 or Out 2 on front-panel or 2-port / 2-source PNA-X
- Port 3 or Port 4 on 4-port PNA-X
**VNA Source 2 for LO on 4-port models**

---

**Path Configurator - Default configuration on PNA-X option 423**

- By default, Internal source2 (Src2) is supplied at ports 3 and 4.
- Bypass switches must be in "Thru Path"

---

**Using the Noise Figure Application**

Use the following general procedure to make measurement with the Noise Figure App:

1. Connect Tuner and Noise Source.
2. Create a Noise Figure Measurement.
3. Make Noise Figure Settings.
4. For Opt 029 and H29, copy your Noise Source ENR file to the VNA C:\Program Files(x86)\Keysight\Network Analyzer\Noise folder.
5. Perform Calibration
6. Connect the DUT. Learn more.
7. Measure Noise Figure.
8. **Optional** Click File, then Save to save Noise Figure data in the following **formats**: (available ONLY when NF correction is ON.)

- *.CTI  Citifile
- *.PRN
- *.nco  Noise Correlation Matrix data in S2P format. See Noise Model.

See Also: Measurement Tips

---

**Connect Noise Tuner and Noise Source**

1. Connect the **noise source** to the 28V connector on the PNA-X rear panel. **NOT required for Opt 028.** The Noise Source is turned ON and OFF automatically as needed during a calibration. Connect the noise tuner to Port 2 reference place when prompted during calibration.

2. Connect the **noise tuner** (ECal module). **NOT required** for 50 GHz models and **Scalar Noise Figure measurements.**

   a. On the VNA front panel, remove the **Port 1** jumper cable SOURCE OUT / CPLR THRU. Opt 028 allows Noise Figure measurements using any two VNA ports. Connect the noise tuner to the front-panel jumpers for the source (DUT input) port.

   b. Connect M-F tuner (N4691B-M0F) using the supplied cable (N5242-20137) and adapter (85052-60013).

   c. When using F-F ECal module (N4691B-00F), order a 3.5 mm M-M adapter (85052-60014).

---

**Create a Noise Figure Measurement**

1. On the VNA front panel, press **Meas > S-Param > Meas Class....**

2. Select **Noise Figure Converters**, then either:
OK delete the existing measurement, or

New Channel to create the measurement in a new channel.

3. A Noise Figure measurement is displayed. Do the following to add or change parameters to display.

### NFX Parameters

#### How to ADD NFX Parameters

1. Create an NFX channel.
2. Then do the following:

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Trace</strong> &gt; <strong>Trace N</strong> &gt; <strong>Trace N</strong>.</td>
<td>1. Click Instrument</td>
</tr>
<tr>
<td>2. Press <strong>Trace</strong> &gt; <strong>Trace Setup</strong> &gt; <strong>Measure...</strong>.</td>
<td>2. Select Trace</td>
</tr>
<tr>
<td>3. Select a trace by pressing <strong>Trace</strong> &gt; <strong>Trace N</strong> &gt; <strong>Trace N</strong>.</td>
<td>3. Select Add Trace</td>
</tr>
<tr>
<td>4. Press <strong>Trace</strong> &gt; <strong>Trace Setup</strong> &gt; <strong>Measure...</strong>.</td>
<td>4. Click Instrument</td>
</tr>
<tr>
<td>5. Select a parameter.</td>
<td>5. Select Trace</td>
</tr>
<tr>
<td>6. Select <strong>Measure...</strong>.</td>
<td>6. Select <strong>Measure...</strong></td>
</tr>
</tbody>
</table>

#### How to CHANGE NFX Parameters

1. Create an NFX channel.
2. Select the parameter to change.
3. Then do the following:

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Select a trace by pressing <strong>Trace</strong> &gt; <strong>Trace N</strong> &gt; <strong>Trace N</strong>.</td>
<td>1. Right-click on a trace.</td>
</tr>
<tr>
<td>2. Press <strong>Trace</strong> &gt; <strong>Trace Setup</strong> &gt; <strong>Measure...</strong>.</td>
<td>2. Select a parameter</td>
</tr>
<tr>
<td>3. Select a parameter.</td>
<td></td>
</tr>
</tbody>
</table>

The same Noise parameters that are available in the Noise Figure application are also available in the NFX application. Learn more.
In addition, the following Mixer and Raw Receiver parameters are available in an NFx channel:

### Mixer Parameters

- **SC21** - Conversion Loss
- **SC12** - Reverse Conversion Loss
- **S11** - Input match
- **S22** - Output match
- **IPWR** - Input power to mixer/converter. Same as R1 (Source1)
- **OPWR** - Output power to mixer/converter. Same as B (Source1)
- **RevIPWR** - Power applied to mixer/converter Output. Same as R2 (Source2)
- **RevOPWR** - Power measured at mixer/converter Input. Same as A (Source2)

### Raw Receiver Parameters

Specify a receiver to measure at LO1 frequencies with the notation:

- `<Receiver>LO1`
- For example: **ALO1** or **R1LO1**

Specify a receiver to measure using a source port, with the notation:

- `<Receiver>_<source port>`
- For example: **A_3** or **R1_1**

### How to start the Noise Figure Setup dialog

**Using Hardkey/SoftTab/Softkey**

1. **Freq** > **Main** > **NFX Setup...**

**Using a mouse**

1. Click **Stimulus**
2. Select **NFX Setup...**

### Valid Mixer Configuration / Sweep Type Combinations
The following are the **Valid Mixer Configurations:**

<table>
<thead>
<tr>
<th>Sweep Type</th>
<th>Input</th>
<th>LO</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>Swept</td>
<td>Swept</td>
<td>Fixed</td>
</tr>
<tr>
<td>Linear</td>
<td>Swept</td>
<td>Fixed</td>
<td>Swept</td>
</tr>
<tr>
<td>CW</td>
<td>Fixed</td>
<td>Fixed</td>
<td>Fixed</td>
</tr>
<tr>
<td>CW</td>
<td>Fixed</td>
<td>Swept</td>
<td>Swept</td>
</tr>
</tbody>
</table>

For determining a valid mixer configuration with 2 LOs, one Fixed LO and one Swept is equivalent to having a single-stage Swept LO.

If you create an invalid Sweep Type / Mixer Configuration, a red message appears like the following:

```
ERROR: Input range must be swept in Linear sweep mode.
```

If this occurs, change the **Sweep Type** on the **Frequency** tab.

See other rules for configuring a mixer.

The following tabs are available on the NFX Setup dialog:

- Frequency
- Power
- Noise Figure Setup
- Mixer Frequency (separate topic)
- Mixer Power (separate topic)
- Mixer Setup (separate topic)
## Frequency tab - NFX Setup dialog box help

<table>
<thead>
<tr>
<th>Sweep Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear - Use for Swept Input parameters</td>
</tr>
<tr>
<td>CW Frequency - Use for Fixed Input parameters</td>
</tr>
</tbody>
</table>

### X-Axis Display

Annotation: Select the frequency range to display on the X-axis.

### Sweep Settings

Click to learn more about these settings.

- **Number of points**
- **IF Bandwidth** For standard VNA receiver measurements. This setting is important for improving noise measurement accuracy. [Learn more.](#)
- **Start / Stop, Center / Span** frequencies.

[Learn about the Load/Save *.mxr files](#), and other buttons across the bottom of all NFX Setup tabs.
**Power Tab - NFX Setup** dialog box help

Learn about this dialog.

Learn about the Load/Save *.mxr files, and other buttons across the bottom of all NFX Setup tabs.

---

**Noise Path Configurator** dialog box help

Learn about this and the Path Configuration dialog for the 50 GHz models.
This dialog is identical to the Noise Figure Setup for amplifier EXCEPT for the following setting. Learn about the rest of this dialog.

**Enable Source Pulling for S-Parameters** (For mixers with low reverse isolation). When checked, during S22 (output match) measurements, the noise tuner is switched to present different impedance states to the DUT input. From these measurements, S22 is computed as though the input is seeing a 50 ohm match. This requires more sweeps. Check this box when the converter has low reverse isolation, as is the case when the NO output path is NOT padded with attenuation. Otherwise, clear this checkbox as S22 measurements will not be improved.

An accurate S22 measurement is essential when measuring S-parameters during an NFX calibration.

Learn about the Load/Save *.mxr files, and other buttons across the bottom of all NFX Setup tabs.

Learn about this dialog.
Mixer (LO) Power tab - NFX Setup dialog box help

Learn about this dialog.

Mixer Setup tab - NFX Setup dialog box help

Learn about this dialog.
Calibration for Noise Figure on Amplifiers and Converters (NFX)

This topic discusses calibration for both Noise Figure on Amplifiers and Noise Figure on Converters (NFX).

- Overview
- How to Perform a Noise Figure Cal
  - Select Calibration Method
  - Configure Noise Source
  - Select DUT Connectors and Cal Kits
  - Measure Standards Steps
  - NFX Calibration
  - Validate Noise Source Cal

See Also

Noise Figure and TRL Cal

See Noise Figure and NFX Applications

Noise Figure Calibration Overview

**Note:** Noise Figure results are NOT at all accurate without a Noise Figure calibration.

NFX Calibration

Noise Figure calibration is very similar for both amplifiers and converters (NFX).

- NFX does NOT offer ‘S-params ONLY’ calibration.
- NFX includes an optional LO Power Cal.
- NFX Cal may generate a Cal Kit Frequency Error message. Learn more.
Calibrating the Noise Receivers

**Note:** The term 'noise receiver' is used here to refer to the receiver that is used to measure noise. It can be a standard VNA receiver or the dedicated noise receivers that are provided with Opt. 029.

The noise figure calibration process is different depending on if a Noise Source or a VNA source (calibrated with a power meter) is used to calibrate the noise receiver.

**Using a Power Meter**

When 'Use Power Meter' is selected on the 'Select Cal Method' dialog, a power meter is used in place of a noise source to characterize the noise receiver. **IMPORTANT: The power cal step must be completed first when acquiring the calibration.**

The process happens in three steps:

1. A Source Power Cal is performed at the port connected to the DUT's input, with a power level that is specified on the first measurement step of the calibration wizard.

2. A THRU connection is made from the calibrated source port to the specified noise receiver port. The gain of the noise receiver is then measured, as well as the receiver's noise floor.

3. With the THRU connection in place, the swept-frequency response of the noise bandwidth filter is measured. Since the noise receiver uses double-sideband homodyne mixing, the user sees a symmetrical response representing the low- and high-side responses, with a notch in the middle that nulls out the DC response. From the measured filter shape, the equivalent noise bandwidth is calculated. This information combined with the data from step 2 gives the gain-bandwidth product and noise figure of the noise receiver.

The following are variations to this process depending on the type of noise receiver selected.

**Using the low-noise (Opt 029) receiver:**

a. Step 3 (measure the frequency response of the noise bandwidth filter) is always performed, over the frequency range specified in the noise figure channel.

b. The gain-bandwidth information is contained within the noise figure calset.

c. **Note:** 'Use Power Meter' is NOT allowed when the Noise bandwidth is set to 8 MHz or 24 MHz.

d. Noise averaging is not automatically turned on.

**Using a standard receiver:**

a. Step 3 (measure the frequency response of the noise bandwidth filter) is performed one time for each noise bandwidth and each measurement receiver. When this step occurs, it happens over the full frequency range of the VNA. Therefore, a broadband connection must be ensured -- there should be
NO filtering in the thru path. For example, a wave-guide-to-coax adapter, which has a high-pass frequency response, should NOT be included.

b. After the initial cal, the noise bandwidth data is saved to an xml file and reused for subsequent noise calibrations. As new noise bandwidths and receivers are selected, new data is appended to the xml file. If you suspect that the xml file might contain bad values for the noise filter bandwidths, then delete:
   For XP: C:\Program Files\Keysight\Network Analyzer\Noise\noiseBW.xml
   For Win7: C:\programdata\Keysight\network analyzer\Noise\noiseBW.xml
This will cause the noise-filter shape for the selected receiver to be remeasured the next time a noise cal is performed using a standard receiver.

c. Noise averaging is automatically turned on, with a default value of 100 noise averages. This value can be modified as necessary. Learn more about Noise Averaging.

Using a Noise Source (See Noise Source requirements). NOT used when measuring noise figure with Standard VNA receivers.

A Noise Source is a device that generates two very consistent levels of noise over its operating frequency range:

- Hot (On) - the Noise Source is biased in order to provide a high level of noise.
- Cold (Off) - the Noise Source is unbiased to provide ambient temperature noise level.

These levels are measured by the Noise Source manufacturer and provided in table and electronic format with each Noise Source by serial number. The electronic file is known as the ENR (Excess Noise Ratio) file.

1. The Noise Source is connected to the noise receiver through test port 2.

   **Note:** For highest accuracy, the noise source should be connected as close as possible (the least amount of electrical loss) to the VNA port 2 connector. This causes the largest difference between the Noise Source HOT (on) and COLD (off) settings.

2. The Noise Source is measured by the noise receivers at each measurement frequency. The differences between the known ENR noise levels and the measured noise levels are the noise error terms. These values are removed from subsequent noise measurements.

3. During the Noise Source measurements, noise averaging and noise bandwidth is automatically turned ON to the values that you specify. Learn more about Noise Averaging.

**Following the Noise Receiver Cal**

- **A 2-port S-parameter calibration** is performed on the noise figure channel. This is because S-parameters
are measured at each frequency step before a noise measurement. Also during the S-parameter cal, at least FOUR different impedance states are presented at port 2 in order to later characterize the noise generated by the noise receiver. This cal can be either a SOLT or TRL cal. See Noise Figure and TRL Cal.

- After calibration, correction is automatically turned ON. The VNA status bar shows VNC_2P (for Vector) or SNC_2P (for Scalar).

**How to Perform a Noise Figure Calibration**

- Make the noise figure channel the active channel.
- Connect the noise figure Tuner to the VNA (for Vector noise figure cal).

**Using Hardkey/SoftTab/Softkey**

1. Press CAL > Main > Smart Cal....

**Using a mouse**

1. Click Response
2. Select Cal
3. Select Smart Cal...

The following Cal Wizard pages are unique to noise figure calibration. The remaining pages that are presented are the same as those in the standard Cal Wizard SmartCal.

**Select Calibration Method** dialog box help

- Vector Noise - Comprehensive noise figure calibration
- S-Parameter Only - Does NOT calibrate the noise receivers. NOT offered with NFX.
- Scalar Noise - Calibration for scalar noise figure measurements. Learn more.

**Enable LO Power Cal** - NFX ONLY. Check to cause the Cal Wizard to guide you through a
Power Calibration on the LO source.

**Note:** NO correction is provided for an adapter that may be used to connect the power sensor to the LO source.

### Noise Tuner

- Not available when Scalar Noise is selected.
- Select from the ECal modules that are connected to the USB.
- For 50 GHz Noise Receivers, select "Internal" to use the built-in Noise Tuner. Learn more.

### Orientation

**AutoOrient Tuner**  Check to allow the noise tuner orientation to be auto-detected. When cleared, use the following two fields to provide manual orientation of the noise tuner.

**Tuner In (SOURCE OUT) / Tuner Out (CPLRTHRU):** Specify the ECal module labels that are connected to the VNA front panel jumper connectors. Learn how to connect the noise tuner.

**Detect Tuners**  Click to re-detect the Noise Tuners (ECal modules) that are connected to the USB. If the ECal module is not detected, check the USB connection, then click this button. The label below the button indicates the total number of ECal modules that are connected to the USB.

### Receiver Characterization - Learn more about this process.

- **Use Noise Source** - A noise source is used to characterize the low-noise receivers.
- **Use Power Meter** - A Power Meter/Sensor is used to calibrate a VNA source, which then is used to characterize either the low-noise receivers or a VNA receiver. This selection is made for you and can NOT be changed when **NA Receiver** is selected on the Noise Figure Setup dialog.

**Note:** **Use Power Meter** is NOT available when the Noise Bandwidth is **8 MHz** or **24 MHz**. In the Noise Setup dialog, lower the Noise Bandwidth to allow this selection.
Configure Noise Source (Opt 029) dialog box help

**ENR File**  Select the Noise Source ENR file. If not already there, copy your Noise Source ENR file to the VNA C:\Program Files(x86)\Keysight\Network Analyzer\Noise folder. Then click **Browse** to find the ENR file.

**Clear ENR List**  Scroll to the bottom of the ENR list, then click to remove the selected ENR file. Then browse or select to find a new file.

**Edit ENR**  Click to launch the **ENR Editor** dialog box which is used to change or create ENR files. This is NOT usually necessary.

**Temperature**  Specify the current temperature at the Noise Source connector. The Noise source is kept ON during noise figure measurements. This results in the Noise Source being a few degrees warmer than Ambient temperature, and a more accurate calibration. See **Noise Figure tips** to learn more about the significance of temperature.

See **Noise Source requirements**.

Configure Noise Source dialog box help

**DC Bias Source using**  Select 28 V Rear Panel for PNA or External Source for PXI

**Use USB Noise Source**  Select the required noise source from selection.

**Temperature**  Specify the current temperature at the Noise Source connector. The Noise source is kept ON during noise figure measurements. This results in the Noise Source being a few degrees warmer than Ambient temperature, and a more accurate calibration. See **Noise Figure tips** to learn more about the significance of temperature.

See **Noise Source requirements**.
warmer than Ambient temperature, and a more accurate calibration. See Noise Figure tips to learn more about the significance of temperature.

**ENR File** Use USB Noise Source Internal File or select the Noise Source ENR file. If not already there, copy your Noise Source ENR file to the VNA C:/Program Files/Keysight/Network Analyzer/Noise folder. Then click **Browse** to find the ENR file.

**Clear ENR List** Scroll to the bottom of the ENR list, then click to remove the selected ENR file. Then browse or select to find a new file.

**Edit ENR** Click to launch the ENR Editor dialog box which is used to change or create ENR files. This is NOT usually necessary.

See Noise Source requirements.

Click either Create or Edit to launch the same dialog box, shown below.

- **Edit** populates all fields with existing data which can then be edited and stored.
- **Create** has empty fields except for frequencies.
**ENR Numeric Data**

Use **Previous** and **Next** buttons to scroll to **Entry #** to edit. Type **ENR** value in **dB**, then press **Enter**.

**Done**  Click when finished editing all values. Then click **Store ENR File** to save the file.

**Identifying Data**

**Model #** of the Noise Source. This can NOT be changed.

**Serial #** of the Noise Source.

**Temperature and Humidity** in which the Noise Source was calibrated. This is for information only. The ENR data is always normalized to 290 Kelvin.

**KeyBd**  launches a mouse-driven keyboard.

**Store ENR File**  Click to launch a dialog to save the new or edited ENR file.
Port 1 and Port 2

DUT (Device Under Test) Connectors Specify the connector and gender of the DUT.

Cal Kits Select the Cal Kit to be used to calibrate each test port. The list for each DUT Port displays kits having the same connector type as the DUT. Using incorrect calibration standards can significantly degrade measurement accuracy. Learn more.

Power Sensor Used to calibrate the source port. Specify the connector and gender of the Power Sensor.

Noise Src Used to calibrate the noise receivers (Opt 029). Specify the connector and gender of the Noise Source. The Keysight 346C has an "APC 3.5 male" connector.

Note: For highest accuracy, the noise source should be connected as close as possible to the VNA port 2 connector. This causes the largest difference between the Noise Source HOT (on) and COLD (off) settings.

For both Cal devices (power sensor and noise source, specify the connector type and gender. When the Cal device connector is NOT the same type and gender as the DUT Port connector, then for optimum accuracy, extra cal steps are used to measure and correct for the adapter that is used to connect the Cal device to the reference plane.

Select Ignored (at the bottom of the DUT Connectors list) to NOT compensate for the adapter.

Select the Cal Kit that will be used for that process.

De-embed power sensor adapter / noise source adapter / thru adapter The VNA uses the connector type and gender of the DUT along with the connector type and gender of the cal device to determine if an adapter removal operation is taking place AND whether or not that removal operation requires an additional cal step.

However, the use of the connector type can, in special cases, hide the need for the extra cal step. Check the "De-embed..." box in these cases to inform the VNA that the extra step is needed.

Such a case is illustrated below where the noise source is connected close to test port 2 for higher accuracy. If unchecked, the VNA would assume in this case that the Noise Source is connected to
the Thru standard at the port 1 (DUT input) reference plane.

**Source Cal Settings**  Click to launch the Source Power Cal (for apps) dialog. This dialog is used to set Power Meter / Sensor settings for both the Port 1 Power Cal, and the optional LO Power Cal.

**Modify Cal**  Check, then click Next, to Modify Cal (Standards AND Thru Method).

**Note:** Enhanced Response Calibration is NOT supported with noise figure.

---

**Measure Standards Steps** dialog box help

**Power Level** at which to perform the Power Cal.

It is usually best to set power level to 0 dBm at the power sensor because the power sensor is calibrated at that level. Lower power levels will yield a slower and noisier calibration.

However, with 20 dB of source attenuation (default NF setting), the VNA may not be capable of achieving this power level at higher frequencies. To check the max leveled power, view an R1 (port 1 reference receiver) trace over the frequency range of interest, then increase the power until roll-off appears. Power levels at the test port may be approximately 2 dB lower than at the R1 receiver.

If an external component is used between the VNA test port and the calibration reference plane,
then adjust the power level so that the power at the sensor is about 0 dBm if possible.

The current source attenuation value is shown on the dialog.

**LO Power Cal (Optional)** When enabled, perform a Source Power Cal at the DUT LO input. An LO must already be selected. Learn how. The power level of the LO source calibration is set on the NFX (LO) Power Tab.

**Connect Noise Source to the Port 2 measurement (reference) plane**

When the "De-embed Adapter.." boxes are checked, additional cal steps are required.

**Subsequent Steps**

- **Connect Port 1 to Port 2** - Connect port 1 reference plane to the port 2 reference plane using the required Thru standard or adapter.
- **Connect ECal to Ports 1 and 2** - Connect the ECal module between the port 1 reference plane and the port 2 reference plane.

**NFX Cal Kit Frequency Error**

When performing an NFX calibration with at least two ECal modules with different connector types, you may see the following error message.

This occurs because the VNA performs two full 2-port cals: one at the input frequencies with both connector types, and one at the output frequencies with both connector types. One of the ECal modules may not be defined over both frequency ranges.

To overcome this, you can perform a User-Characterization for the limited ECal module over the required frequency range. The VNA will present a warning, but it will be allowed. Learn how.

**Validate Noise Source Cal**

To validate a Noise Source calibration, connect the Noise Source to Port 2 and measure ENR.

Compare the measured values to the values in the ENR table.
How to manually turn the Noise Source ON | OFF

1. Press **Power > Main > Noise Source.**
When performing a TRL (or LRL, LRM) Cal as the 2-port S-parameter calibration of a Scalar or Vector Noise Figure measurement, you may see an error message that states that there are not enough standards for the cal.

This appears because, during the TRL calibration, at least **FIVE** impedance states must be presented to the Noise Receiver port. A typical TRL Cal Kit does not have 5 standards with the same connector type and gender as the DUT output port, and with different impedances.

To correct this situation, you must define additional standards for your TRL Cal Kit using the Edit Cal Kit dialog.

**Notes**

- Extra impedance standards are NOT required when you select and use an ECal module to perform the De-embed noise source adapter. In this case the ECal module is used to present five different impedance states to the Noise Receiver port.
- You can view the impedance match by measuring the standard over the frequency range of interest while viewing the Smith Chart format. Ideally, all five standards should have a response at different areas of the Smith Chart.

**To Modify the Cal Kit**

If the TRL Cal Kit is also defined as an SOLT kit, those **Selected Standards** will also be searched for an appropriate Reflect Standard.

1. Click **Edit** on the Error dialog, or **Cancel**.
2. Press **CAL > Cal Sets & Cal Kits > Cal Kit...**
3. Select the Cal Kit to be edited, then click **Edit...**
4. On the Edit Kit dialog, click the **TRL** tab.
5. For each of the following **Calibration Kit Classes**, note the **ID** number in the **Selected Standards** field:
For example, in the following image two shorts are defined as TRL REFLECT standards: ID numbers 7 and 14. Because they are already being used, you can NOT use these IDs for the additional standards required by the Noise Cal. You must select other standards available in the kit or you can define new standards.

6. Select the SOLT tab, then select any of the following Calibration Kit Class definitions:

   a. SA (Open)

   b. SB (Short)
7. In the **Available Standards** field, find a standard that is NOT one of the TRL IDs noted above. In this image, two opens are already assigned as SOLT standards, but because they are NOT assigned as TRL reflection standards, they are eligible to be additional standards.

8. If necessary, click >> to add it to the **Selected Standards**.

**Note:** Be sure to choose standards with the **same** connector gender as the DUT output port.

If no others exist, the following are good options:

- Offset Short
- Offset Open

**If your TRL Cal Kit does NOT have FIVE standards**

You can create a new device by reusing the LINE standard. However, instead of connecting both ends of the line, leave the line unterminated.

Although the following example creates an Open standard, during the calibration you will connect any line standard to the Noise Receiver port. The standard or definition is not important. It is simply another impedance state.

The following is an example of how to create this device.

1. On the Standards tab, click **Add**.
The following dialog appears:

Select **OPEN**, then click **OK**.

The following dialog appears:

3. Change the description to **Unterminated Line**. This will provide a prompt during the calibration.

4. Change **Connector** to match your physical line standard.

5. Note the new **Standard ID** number.

6. In the same manner as step 6 (above), on the **SOLT** tab, select **SA**.
7. Assign the new standard ID to the Selected Standards.

**Important:** Leave the new standard at the bottom of the Available standards list to which it was added. That way it will have no effect on the SOLT cals using that kit.
Narrowband Pulsed Application

Note: Beginning with the A.14.80.xx firmware release, the Narrowband Pulsed Application is no longer available. The Integrated Pulse App (Opt S93026A/B) replaces this application.

The Narrowband Pulsed Application is a Visual Basic program that provides a user interface for making pulsed measurements.

In this topic:

- Required Options
- Connecting External Pulse Generators
- Using the Narrowband Pulsed Application
- How to Configure Pulse Generators and Receivers
- Calibration in Pulse Mode
- Pulse Profiling
- Signal Reduction versus Gate Width
- Pulsed Frequency Converter Measurements
- Writing your own Narrowband Pulsed Application

- Enhanced Pulse Measurement Capabilities
- Support for Internal Pulse Generators / Modulators (PNA-X only)

See Also

- Learn about the Wideband Pulsed Application.
- For more conceptual information see our Pulsed Measurement App Notes.
- See PNA-X Block Diagram of IF Path / Pulse Generators / Source Modulation
- Programming commands

Other IF Access Topics
Required Options and Equipment

The VNA H08 option provides the Narrowband Pulsed Application. The following options are also required. If your VNA does not have the required options, a message is displayed on the screen. For more information, see Pulsed-RF Measurements Configuration Guide.

Models

- PNA-X Opt 224 (2-port Dual Source) select the 2-port Dual Source Configuration in the Path Configuration dialog to provide Pulse modulation on both port 1 and port 2.
- Use the Pulse I/O connector to access the internal pulse generators.
- See the PNA-X IF Path Configuration block diagram which includes the Pulse Modulators and Generators.

Other VNA Models

- PNA-L models: H08 NOT available.
- Keysight 81104A or 81110A Pulse Generator with ONLY the 81105A or 81111A output modules. The 81112A module does NOT have selectable 50 ohm/1K ohm output impedance/load compensation to drive the 1K ohm VNA IF gates. For more information, see the 81100 Family of Pulse Pattern Generators Technical Specifications at: http://literature.cdn.Keysight.com/litweb/pdf/5980-1215E.pdf

Connecting External Pulse Generators to the Z5623A H81

Each 81110A Pulse Generator has two output modules. Each output can drive a VNA IF Receiver or Source Modulation (Z5623A H81). Connect the Pulse Generators as follows:

81110A front panel connectors

- Connect GPIB cables to the 81110A and VNA.
- Connect the VNA 10 MHz Ref Out to the 81110A 10 MHz IN.
- If using two 81110As for a total of 4 outputs, then connect the TRIGGER OUT of one to the EXT INPUT of the other 81110A.
Connect the 81110A OUTPUTs to the VNA rear panel IF inputs to be gated. The outputs are mapped in the Pulsed Generator Configuration dialog box.

Connect the Z5623A H81Pulse Test Set (optional) to the VNA front-panel port 1 loops as follows:

<table>
<thead>
<tr>
<th>VNA</th>
<th>H81</th>
</tr>
</thead>
<tbody>
<tr>
<td>Src Out</td>
<td>Source IN</td>
</tr>
<tr>
<td>CPLR</td>
<td>CPLR</td>
</tr>
<tr>
<td>THRU</td>
<td>THRU</td>
</tr>
<tr>
<td>RCVR R1 IN</td>
<td>RCVR R1 Out</td>
</tr>
</tbody>
</table>

See Also

- VNA Front-panel loops
- PNA-X IF Connectors
- 81110A Documentation
- Z5623A H81 Documentation

Using the Narrowband Pulsed Application

How to start the Narrowband Pulsed Application

Using **Hardkey/SoftTab/Softkey**

1. Press **Macro > Pulse**

Using a mouse

1. Click **Utility**
2. Select **Macro**
3. Select **Pulse**

See Also

- See programming commands to launch the Macro remotely.
- See how to write your own custom Narrowband Pulsed Application.

Keypad Data Entry

The VNA front-panel Numeric Entry and Navigation keys can be used for dialog box input. Also, a
keyboard can be used to enter values, including alpha characters for prefixes (for example, \texttt{u} for usec.) . After typing values, first press \texttt{Enter}, then press \texttt{Tab} to go to the next field.

The following is an image of the main dialog box:

![Agilent PNA Pulse Application dialog box help](image)

**Pulsed Application Main** dialog box help

\textbf{Note:} An error message may appear on the VNA stating that the response frequency has exceeded the maximum allowed frequency.

The Narrowband Pulsed Application may set the offset frequency (option S93080A) of the VNA to some value other than zero (the default value). If the stop frequency is set to the maximum of the VNA model, then the error message will appear.

To fix this, set the stop frequency to a value that is at least 2 kHz less than the maximum allowed. For example, if you have a 20 GHz VNA, and the stop frequency is set to 20 GHz, and the error message appears, then set the stop frequency to 19.999998 GHz.

See Block Diagram of IF Path / Pulse Generators / Source Modulation

**Configure**

You can configure more than one channel to make pulsed measurements, but the channels must use the same pulse generator settings.
Only the Keysight 81110A Pulse Generator is supported with the Narrowband Pulsed Application. Refer to the 81110A documentation for pulse repetition frequency and duty cycle capabilities.

**See Also**

- Configure Receivers
- Converter Measurements

**Edit / Undo**  Pulse Application settings revert to those when Apply was last pressed.

**Desired PRF and IFBW**  Enter the DESIRED values. When Calculate is pressed, one or both of these values may change.

- **Pulse Repetition Frequency (PRF)**: Frequency of the pulses from the Pulse Generator.
- **Pulse Repetition Interval**: $1/\text{PRF}$  Changes to either PRF or this setting changes both.
- **Receiver IF Bandwidth**: IF Bandwidth of the VNA. Choose a setting from 1 Hz to 10 KHz.
- **Fixed PRF**  When checked, (default setting) the Calculate algorithm will NOT adjust the PRF, but only change the IF Bandwidth. When cleared, both the PRF and IF Bandwidth values are adjusted as necessary.

**Note:** On VNA's with DSP version 4, the Calculate algorithm will NOT find nulling at several PRF frequencies. If this error is returned, add a small offset to the PRF (for example, 2.1 MHz instead of 2 MHz) or clear the Fixed PRF checkbox.

**Modulation/Gates**  The Source Modulation and four VNA receiver gates can each have their own, or share, Pulse Generator outputs. Shared outputs have identical Width and Delay values. To configure and enable outputs, click Configure, then Pulse Generators to launch the Pulsed Generator Configuration dialog box.

**Note:** Option 036 and 037 limits the Source Modulation width to 117 ns.

- **Width**  Pulse Width.
- **Delay**  The delay that occurs before the pulse.
**Duty Cycle** Calculated Duty Cycle of the source and each of the selected receivers. Updated when Calculate is pressed.

**Pulse Mode On** When this box is checked, the VNA is enabled for Pulsed measurements. The VNA Status Bar annotation indicates the following:

- **G** Internal IF gates enabled.
- **F** Filtering for Pulsed Measurements enabled.

**Apply** All selections are sent to the pulse generator and the active channel of the VNA.

**Calculate** All selections are calculated and valid PRF and IFBW values are entered in their fields. If these settings are not acceptable, try changing the values you previously entered and click Calculate again. When acceptable values are attained, click Apply to send these values to the pulse generator and VNA.

**Pulse Profile** Launches the Pulse Profile dialog box. Same as clicking View / Pulse Profile. If not available, check Pulse Mode ON, click Calculate, then Apply.

**Minimize** Click to minimize the dialog box to make changes in the VNA application. To see the dialog again, select Macro, Pulse, or turn the Status Bar ON.

**Save** All settings from the Narrowband Pulsed Application are saved in a *.ppf file. These settings are NOT saved with VNA instrument state.

**Recall** Restore settings from the specified *.ppf file that were previously saved.

**Close** Closes the dialog box without saving changes.
How to configure Pulse Generators / Modulators and Receivers

From the Pulse App main dialog box

Learn about...

- Configure Receiver Gain
- Converter Measurements
- **No Pulse Generators** When checked, the Narrowband Pulsed Application does NOT attempt to communicate with internal or external pulse generators. This setting is used for troubleshooting purposes.
- **No SW Gating** When checked, the improved SW gating sensitivity is turned OFF. This setting is used for troubleshooting purposes.

Pulsed Generator Configuration dialog box help

Notes:

- See Block Diagram of PNA-X IF Path / Pulse Generators / Source Modulation
- For PNA-X Opt 224 (2-port Dual Source) select the 2-port Dual Source Configuration in the Path
Configuration dialog to provide Pulse Modulation on both port 1 and port 2.

- This dialog may look different depending on the VNA model and number of receivers available.

Configures either the internal pulse generators (PNA-X models with relevant options), or Keysight 81110A Pulse Generator outputs. You can configure each 81110A Pulse Generator with either one or two 81111A output modules.

The Source Mod and four VNA receiver gates can each have their own, or shared, pulsed generators allowing identical Width and Delay values which are selected on the Main dialog.

To share an external generator output between one or more VNA inputs, use the same GPIB address and output module for each VNA input.

**Internal Pulse Gen Output** (available ONLY on the PNA-X opt S93025A/B)

Specify the Pulse Gen (1 through 4) to use to modulate each of the VNA receiver IF gates or Sources.

**External Pulse Generator settings**

- **GPIB Addr:** The GPIB address of the 81110A.
- **Output:** The output module of the 81110A.
- **Primary:** The 81110A that uses the 10 MHz reference signal from the VNA.
- **Enabled:** Turns the pulse output ON.

**External Gate/Modulator settings**

- **High:** Specify a 'TTL-High' voltage level
- **Low:** Specify a 'TTL-Low' voltage level
- **Ext Impedance:** Impedance of the modulator used to create the pulse.
- **Complement:** When this box is cleared, TTL HIGH is the pulse. When checked, TTL LOW is the pulse.

**Using Internal Modulators** When this box is checked, the voltage, impedance, and complement values are forced to settings that prevent damage to the internal modulator.

**Using Internal Pulse Generators** Makes the appropriate settings on this dialog available.

**Using Internal VNA gates** When this box is checked, the voltage, impedance, and complement values are forced to settings that prevent damage to the internal gates.
Receiver Gain Configuration dialog box help

See Block Diagram of PNA-X IF Path / Pulse Generators / Source Modulation

This dialog may look different depending on the VNA model and number of receivers available.

Sets the gain of each VNA receiver manually or automatically.

**Auto** - The VNA selects the best gain level to make pulsed measurements.

Use the following to manually set the gain for each receiver.

**Low** - about 0 dB of gain

**Medium** - about 17 dB of gain

**High** - about 24 dB of gain

The PNA-X has the following attenuation settings:

**Low** - 30 dB of attenuation

**Medium** - 15 dB of attenuation

**Hi** - 0 dB of attenuation

Calibration in Pulse Mode

To perform a calibration in pulse mode (option H08), first configure and apply the pulse parameters (PRF, Pulse Width, Delays, IF gating, and so forth) **before** calibrating the system. This will ensure the VNA is configured properly during the calibration and measurement.

When performing **Unknown Thru** or **TRL calibrations**, ALL receivers must be gated. Otherwise, the error
terms will not be correct after the calibration has completed. This can be accomplished by either having a separate pulse generator output for each of the IF gates, or by connecting pairs of the IF gates together with BNC-T's. For example, if the pulse generator does not have enough outputs, then connect the R1 and R2 IF gates to the same pulse generator output. Also, connect the A and B IF gates to either separate outputs (recommended) or one output (reduces flexibility). The error terms will then be valid after the calibration is complete.

**Pulse Profiling**

Pulse profiling provides a time domain view of the pulse envelope. Profiling is performed using a measurement technique that "walks" a narrow receiver "snapshot" across the width of the pulse. This is analogous to using a camera to take many small snapshots of a wide image, then piecing them together to form a single, panoramic view.

- Pulse Profiling can be performed using ratioed or unratioed measurements.
- Pulse Profiling is performed at a single CW frequency.

**How to perform Pulse Profiling**

From the Pulse App main dialog box,

Click the **Pulse Profile** button. or:

![Configure View Help Desired Pulse Profile](image)

If this setting is unavailable, check **Pulse Mode ON**, click **Calculate**, then **Apply**.
Learn about Pulse Profiling (scroll up)

See Block Diagram of PNA-X IF Path / Pulse Generators / Source Modulation

**Modulation / Gates**

These setting duplicate those found on the main Pulse App dialog box.

In Pulse Profile, the Gate Delay settings (highlighted in yellow) are significant only with certain Measurement Parameter and Couple Gates settings.

**Time Parameters**

**Start, Stop** These two combine to make the window of the assembled pulse profile. To view the entire pulse, the start and stop values must be at least as wide as the Source Modulation Width plus Delay value.

**Step** Each consecutive snapshot is incremented by this value until the stop value is reached. Therefore, the number of points for the pulse profile measurement can be calculated as: (Stop - Start) / Step. The higher the number of points, the longer it takes to make the measurement.

**Measurement Parameter**

**CW Freq.** Frequency of the VNA source.

**Source Port** The VNA port supplying the source power. Only required for single receiver (unratioed) measurements.
**Param(eter)** Only those receiver gates (and relevant measurements) that are configured in Pulsed Generator Configuration are available.

**Note:** When a single receiver (unratioed) is selected, **Gate Delay** Settings (highlighted in yellow on above dialog image) are ignored.

If the reference receiver gate is NOT configured, the average of the Source Modulation pulse is used as the reference. **For example:** With S21 Selected, but ONLY B receiver gate is configured, then...

- **B Gate is walked across the Source Modulation pulse.**
- **Source Modulation pulse average is used as reference (not gated).**

**Coupled Gates** Used when the appropriate receiver gates are configured for your **S-parameter** measurement ONLY. This setting is ignored when a single receiver (**Param**) is selected.

- **Uncoupled** (box cleared) The reference gate is FIXED at the delay setting as the test gate is walked across the Source Modulation pulse as dictated by the **Time Parameter** settings.

  **For example:**
  - **S21 Selected, B and R1 receiver gates configured, Gates Uncoupled**
    - **B Gate is walked across the Source Modulation pulse.**
    - **R1 gate is fixed at pulse width and delay setting.**

- **Coupled** (box checked) The reference gate is walked synchronously with the test gate as dictated by the **Time Parameter** settings. Only the **difference** between the test and reference gate delay values is significant; NOT the absolute values.
For example:
S21 Selected, B and R1 receiver gates configured, Gates Coupled

B gate delay = 3 microseconds,

R1 gate delay = 2 microseconds

Difference = 1 microsecond

B Gate is walked across the Source Modulation pulse.

R1 gate is fixed at pulse width and delay setting.

B gate leads R1 gate by 1 microsecond.

Data Format Log Magnitude, Linear Magnitude, or Phase (only available if S-parameter selected).

Buttons

Show Gates Allows you do change the receiver gating width and delay while looking at the results.

Apply Gate Settings Click after making changes to gate settings.

Continuous Sweep Check, then click Measure, to continuously measure pulse profiling.

Measure Click to start the pulse profile measurement. Becomes Stop when continuously sweeping.

Marker to Delay After making a measurement, you can drag the display maker to any point along the trace. Click this button and the marker time is entered into the Receiver Delay field on the main dialog box.

Save Data Saves time domain data to the VNA hard drive in any of the following formats:

- Touchstone (*.s1p)
- Comma delimited (*.prn)
- Citifile (*.cti)

Learn more about these data formats

Signal Reduction versus Gate Width
The following two figures show the performance of the internal IF gates as the width is narrowed.

The following is a zoomed image of the shaded area (above).

- The straight line shows the theoretical loss in dynamic range due to duty cycle effects when using narrowband detection.
The curved (red) line shows the actual measured performance of the gates.

The minimum gate width for <1dB deviation from theoretical is approximately 20ns.

See the specifications for the option H11 and option H08.

Pulsed Frequency Converter Measurements

The Narrowband Pulsed Application works with both FCA (option S93083A/B) and standard Frequency Offset (opt S93080A) measurements. On the Configure menu, check Converter Measurements. When checked, this setting prevents the Narrowband Pulsed Application from overwriting frequency offset values. This may limit the number of PRF and IFBW solutions that are returned when Calculate is pressed on the main Pulsed Application dialog box.

Note: Pulse Profiling can NOT be performed with frequency converter measurements.

Writing your own Narrowband Pulsed Application

You can use the Narrowband Pulsed Application or use an example program as a template for making your own Narrowband Pulsed Application.

The Narrowband Pulsed Application uses a custom .dll to perform the calculations that are necessary to make pulsed measurements. Use the COM Method below to send and return values to agilentPNApulsed.dll. Then use SCPI or COM commands to control the VNA.

<table>
<thead>
<tr>
<th>COM Example Program</th>
<th>PNA-X Create</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCPI Example Programs</td>
<td>Point-in-Pulse</td>
</tr>
<tr>
<td></td>
<td>Pulse Profile</td>
</tr>
<tr>
<td>COM Methods</td>
<td>ConfigEnhancedNB2</td>
</tr>
<tr>
<td></td>
<td>ConfigEnhancedNBIFAtten</td>
</tr>
<tr>
<td>SCPI commands</td>
<td>SCPI</td>
</tr>
<tr>
<td>COM commands</td>
<td>COM</td>
</tr>
</tbody>
</table>

6225
To create your own Narrowband Pulsed Application, or run the Narrowband Pulsed Application from a remote PC, you must do the following:

1. Copy the following files from the VNA C:\program files(x86)\Keysight\network analyzer\ to a directory on your PC.
   - agilentpnapulsed.dll
   - OffsetList.txt
   - prfbw.txt
   - prfbwmixer.txt

2. To register the ActiveX DLL in Microsoft Windows Operating System:
   - From a command prompt on your PC, navigate to the directory where you copied the DLL.
   - Type: `regsvr32 agilentpnapulsed.dll` and press Enter

For Operating Systems other than Windows, see their associated help files to learn how to register DLL files.
Wideband Pulsed Application

Note: The PNA-X with Opt S93025A/B and Opt S93026A/B has an integrated Pulse Measurement Setup that makes this application no longer necessary on the PNA-X.

The Wideband Pulsed Application configures the PNA-X internal pulse generators and modulators for measuring pulsed S-parameters using the wideband mode detection technique.

The Wideband Pulse Application is designed to be used with the PNA-X with Opt 021, 022, and S93025A/B.

Note: Wideband Pulse application is NOT supported on the E836x and PNA-L models.

See Also

- To learn more about wideband detection, see Application Note 1408-12.
- See a Visual Basic example: Create a Wideband Pulsed Measurement using the PNA-X
- Learn about the Narrowband Pulsed Application.

Download and Install the Wideband Pulsed Application

This application is installed and run as a macro on the PNA-X. Learn more about macros.

2. Click the download link.
3. Save the downloaded file to the VNA hard drive.
4. Double-click the downloaded file to install the Wideband Pulsed Application on the VNA.
5. The application is saved on the VNA at C:\Program Files(x86)\Keysight\Network Analyzer\Applications\WB Pulse\Wideband_pulse.exe.
6. Configure the macro. Learn how.

To learn more about Wideband pulsed application, click Help in the application.
Overview

**Note:** The Phase Noise Application applies ONLY to PNA/PNA-X instruments with serial prefix 6021 and above.

The Option S93031xB Phase Noise application measures the phase noise of a DUT's output signal within a specified offset frequency range.

In this topic:

- Features and Requirements
- Phase Noise Floor versus Frequency Plot
- Hardware Setups
- Phase Noise Display
- Spurious Measurement
  - Spurious Sensibility
  - Spurious Threshold
  - Spurious Table
- Integrated Noise Measurement
  - Integrated Noise Table
- Spot Noise Measurement
  - Spot Noise Table
- Phase Noise Graphical User Interface

**See Also**

- Starting and Exiting Phase Noise
- Configuring Phase Noise
- Displaying Phase Noise Parameters
- Phase Noise Marker Search Functions
- Calibration
- Phase Noise Measurement Examples
  - Setting Up a Phase Noise Measurement
  - Spurious Measurement
  - Integrated Noise Measurement
  - Spot Noise Measurement
  - AM Noise Measurement
Features and Requirements

Features

- Absolute Phase Noise measurements
- AM noise measurement
- Spur identification and removal
- Automatic carrier search
- Extensible to mm-Wave

Requirements

- Phase Noise Option S93031xB
- 2-Port or 4-Port PNA/PNA-X instruments with serial prefix 6201 and above (new Direct Digital Synthesizers)
- Windows 10 operating system

Phase Noise Floor versus Frequency Plot

The noise floor for 2.4 GHz, 4.8 GHz, 9.6 GHz, and 19.2 GHz frequencies rises at an offset frequency of 300 kHz as shown in the following plot:
The following are typical hardware setups for measuring the phase noise of a device. The Option S93031xB Phase Noise application is installed on the PNA/PNA-X.
Phase Noise Display

The Phase Noise application measures the power density of the noise sideband relative to the power of
the carrier over a start/stop offset frequency. The offset frequency corresponds to the distance from the carrier frequency. The log-transformed phase noise is displayed in dBC/Hz.

The bottom of the display area shows the carrier frequency, carrier power, and the Phase Noise start and stop offset frequencies. The x-axis is the offset frequency range displayed on a logarithmic scale to allow viewing the full offset range. Also displayed is the delta frequency indicator next to the Carrier Frequency. The delta frequency displays the frequency from the nominal carrier frequency. The actual carrier is not tuned exactly to the nominal carrier frequency. The delta indicator displays this frequency difference.

If the sweep time is greater than 2 seconds, a sweep indicator is displayed at the bottom of the display area to provide sweep time percentage complete:

Noise tables showing Spurious, Integrated Noise, and Spot Noise data can be displayed below the trace area.

**Spurious Measurement**

Spurs can be analyzed using Spurious Sensibility or by defining Threshold levels.
Spurious Sensibility

Spurious Sensibility = Sensibility Value x Standard Deviation (or sigma).
The Spurious Table can be displayed below the trace area of the display showing Trace and Spur number, Offset frequency, Power level, and Jitter.

<table>
<thead>
<tr>
<th>Trace</th>
<th>Spur</th>
<th>Offset</th>
<th>Power</th>
<th>Jitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>59.905 Hz</td>
<td>-81.31 dBc</td>
<td>19.35 fs</td>
</tr>
</tbody>
</table>

**Integrated Noise Measurement**

Integrated noise is the single sideband noise integrated over a measurement bandwidth from a start/stop offset frequency. Up to four offset frequency ranges can be defined.

A weighting filter can be defined and saved to a file for later use. The weighting filters compensate external effects that affect the phase noise in the specified frequency ranges. When the weighting filters are applied, the trace data are corrected by the filter characteristics before the calculation of integrated noise.
Integrated Noise Table

The Integrated Noise Table can be displayed below the trace area of the display showing trace number, start/stop offset, weighting filename (if used) integrated noise, phase modulation measured in degrees and radians, residual FM (Hz) or residual AM (%), and jitter.

<table>
<thead>
<tr>
<th>Trace</th>
<th>Start Offset</th>
<th>Stop Offset</th>
<th>Weighting</th>
<th>Integ Noise</th>
<th>PM</th>
<th>FM/AM</th>
<th>Jitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 kHz</td>
<td>10 MHz</td>
<td></td>
<td>-56.182 dBC</td>
<td>11.273 kHz</td>
<td>349.32 fs</td>
<td></td>
</tr>
</tbody>
</table>

Spot Noise Measurement

Spot Noise measurement are measurements made at specific frequencies. The measurements can be made at each decade, determined by the start/stop offset frequencies, up to six defined frequencies, or a combination of both decade and defined frequencies.
Spot Noise Table

The Spot Noise Table can be displayed below the trace area of the display showing the specific measurement frequencies and the phase noise in dBc/Hz.

<table>
<thead>
<tr>
<th>Trace</th>
<th>10 Hz</th>
<th>100 Hz</th>
<th>1 kHz</th>
<th>10 kHz</th>
<th>100 kHz</th>
<th>1 MHz</th>
<th>10 MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>-56.123 dBc/Hz</td>
<td>-107.14 dBc/Hz</td>
<td>-119.77 dBc/Hz</td>
<td>-119.01 dBc/Hz</td>
<td>-121.16 dBc/Hz</td>
<td>-136.44 dBc/Hz</td>
<td>-133.20 dBc/Hz</td>
</tr>
</tbody>
</table>

Phase Noise Graphical User Interface

The GUI consists of setup dialogs accessed by clicking on their corresponding tabs. In this way, configurations can be set up quickly. See Configuring Phase Noise for information about these dialogs.
Starting and Exiting Phase Noise

In this topic:

- Starting Phase Noise Application
- Exiting Phase Noise Application

Starting Phase Noise Application

How to start Phase Noise

Using Hardkey/SoftTab/Softkey

1. On the VNA, press **Meas > S-Param > Meas Class**....
2. Select **Phase Noise**, then click **OK**.
3. In the **Confirm Measurement Class Change** dialog, click **OK** to proceed or **Cancel** to exit.
4. The **Phase Noise** application is displayed (shown below).

Using a mouse

1. Click **Instrument**.
2. Select **Meas Class**....
3. Select **Phase Noise**, then click **OK**.
4. In the **Confirm Measurement Class Change** dialog, click **OK** to proceed or **Cancel** to exit.
5. The **Phase Noise** application is displayed (shown below).
Exiting Phase Noise Application

1. Select a different Measurement Class for the currently active channel.
Configuring Phase Noise

The Phase Noise dialog settings are contained within five tabs: **Sweep**, **RF Path**, **Spurious**, **Integrated Noise**, and **Spot Noise**.

In this topic:

- Create a Phase Noise Channel
- Opening the Phase Noise Dialog
- Sweep Tab in Dialog
- RF Path Tab in Dialog
- Source Tab in Dialog
- Spurious Tab in Dialog
- Integrated Noise Tab in Dialog
- Spot Noise Tab in Dialog

See Also

- Displaying Phase Noise Parameters
- Spectrum Analyzer Settings
- Phase Noise Marker Search Functions
- Phase Noise Measurement Examples
  - Setting Up a Phase Noise Measurement
  - Spurious Measurement
  - Integrated Noise Measurement
  - Spot Noise Measurement
  - AM Noise Measurement
- Phase Noise SCPI Programming Examples
  - Setting Up a Phase Noise Measurement
  - Setting Up a Source
  - Spurious Measurement
  - Integrated Noise Measurement
  - Spot Noise Measurement

Create a Phase Noise Channel
1. On the VNA front panel, press **Meas > S-Param > Meas Class...**.

2. Select **Phase Noise**, then either:
   - **OK** to delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.

3. In the **Confirm Measurement Class Change** dialog, click **OK** to proceed or **Cancel** to exit.

4. The **Phase Noise** dialog is displayed.

**Opening the Phase Noise Dialog**

**Accessing Phase Noise Settings**

- **Using Hardkey / SoftTab / Softkey**
  1. Press **Setup > Main > Phase Noise Setup...**.

- **Using a mouse**
  1. Click **Stimulus**.
  2. Select **Phase Noise Setup...**.

**Sweep Tab in dialog help**

**Noise Type** - Selects between standard phase noise measurements and residual noise.
measurements.

**Phase Noise** - (Default) - Measures phase noise of a DUT. Select for single channel residual noise measurement when the input and output frequencies of the DUT are not the same. For example, frequency converters.

**Residual Noise** - Measures the additive phase noise of a DUT. This measurement is set up in the RF Path tab using DUT Input and DUT Output. Select for 2-channel residual noise measurement when the input and output frequencies of the DUT are the same. For example, amplifiers.

**Carrier Frequency** - Sets the carrier frequency. The range of the phase noise measurement is determined by the start/stop offset frequency which is relative to the carrier frequency.

**Carrier Threshold** - Sets the threshold to use during a carrier search.

**Start Offset** - Sets the start frequency of the phase measurement. This is an offset relative to the carrier frequency to position the measurement about the noise sideband to measure power density in dBc/Hz. For example, if the Start Offset is set to 1 MHz with a carrier frequency of 1 GHz, then the dBc/Hz phase noise measurement starts at 1.001 GHz. The minimum frequency is 0.1 Hz.

**Stop Offset** - Sets the stop frequency of the phase measurement. This is an offset relative to the carrier frequency to position the measurement about the noise sideband to measure power density in dBc/Hz. For example, if the Stop Offset is set to 10 MHz with a carrier frequency of 1 GHz, then the dBc/Hz phase noise measurement stops at 1.010 GHz. The maximum frequency is 10 MHz.

**RBW Ratio** - Sets the resolution bandwidth ratio, which is the specified resolution bandwidth percentage of every half decade offset frequency.

Example:

Start Offset = 1 kHz
Stop Offset = 100 kHz
RBW Ratio = 10%

1 kHz - 3 kHz: RBW = 100 Hz (10% of 1 kHz)
3 kHz - 10 kHz: RBW = 300 Hz (10% of 3 kHz)
10 kHz - 30 kHz: RBW = 1 kHz (10% of 10 kHz)
30 kHz - 100 kHz: RBW = 3 kHz (10% of 30 kHz)
**FFT Avg Factor** - Sets the FFT average factor number. The average factor is multiplied by the default average count, which changes for each frequency range. The default average count of the lower frequency range is 1 and at the higher offset frequency range is a larger average count.

**Note:** The minimum **RBW Ratio** setting relative to the maximum **FFT Avg Factor** setting are limited based on the frequency offset. The firmware restricts these settings if they exceed the limits.

**Noise Mode** - Sets the measurement speed and noise floor.

- **Fast** - Fastest measurement speed with highest noise floor.
- **Normal** - Between fastest and slowest measurement speed and noise floor.
- **Best** - Slowest measurement speed with lowest noise floor.

The zone boundaries and decimation factors change with noise mode as shown in the following example:

![Graph showing different modes of measurement](image)

**Defaults** button - Restores default phase noise settings.

**Apply** button - Applies setting changes and leaves the dialog box open to make more setting changes.
Measurement Type: Residual Noise

**VNA Input** - Sets the receiver input to use for the phase noise measurement.

**DUT Input/DUT Output** - Displayed when **Noise Type** in the **Sweep** tab is set to **Residual Noise**. Sets up ratios for residual phase noise measurements. For example, with DUT Input set to a1 and DUT Output set to b2, the additive phase noise at b2 relative to a1 is measured.
**Rcvr Atten** - Sets the attenuator on the test receiver.

**R F Path Configuration**... - Learn more

**Defaults** button - Restores default phase noise settings.

**Apply** button - Applies setting changes and leaves the dialog box open to make more setting changes.

---

### Source Tab in dialog help

![Source Tab in dialog help](image)

**Power On (All Channels)** Check to enable source power for all channels. Only turns power ON if the port State setting is ON.

**Port Powers Coupled**

- **Coupled** (checked) The power levels are the same at each test port. Set power at any test port and all test ports change to the same power level.
- **Uncoupled** (cleared) The power levels are set independently for each test port. Uncouple power, for example, if you want to measure the gain and reverse-isolation of a high-gain
amplifier. The power required for the input port of the amplifier is much lower than the power required for the output port. A power sweep can also be performed with uncoupled power. Learn more about Setting Independent Port Power.

Source Cells

Name - Lists the test ports through which an internal source is available. If an external source has been configured, it will appear at the bottom of the list.

State

- **ON** Source power is ALWAYS ON. Turning ON port 1 will also turn ON port 2 and vice versa. The same is true for port 3 and port 4. Learn about internal second source restrictions.

- **OFF** Source power is never ON, regardless of the measurement requirements. Use this setting to prevent damage to a sensitive DUT test port.

Frequency - Click in the cell, then click **Edit**, to start the **Frequency Settings** dialog (below).

![Frequency Settings Dialog](image)

Power - Sets the power level at the output of the source. Click in the cell, then click **Edit**, to start the **Power Settings** dialog (below).
Pulse - Enable/disable pulse measurements. Learn more.

Power and Attenuator... - Learn more

External Devices... - Learn more

Defaults button - Restores default phase noise settings.

Apply button - Applies setting changes and leaves the dialog box open to make more setting changes.
**Show Spurious Table** - Enable or disable displaying the spurious table below the trace display.

<table>
<thead>
<tr>
<th>Trace</th>
<th>Spur</th>
<th>Offset</th>
<th>Power</th>
<th>Jitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>59.905 Hz</td>
<td>-81.31 dBc</td>
<td>19.35 fs</td>
</tr>
</tbody>
</table>

**Trace** - Displays trace number(s).

**Spur** - Displays the spur number(s) per trace.

**Offset** - Offset frequency.

**Power** - Power level of the spur.

**Jitter** - The phase deviation in the time domain.

**Table Sort Order** - Sets the spurious table sorting by power or by offset.

**Select** -

**Enable Spur Analysis** - Enables and disables spurious analysis search markers. See Spurious Search.

**Spur Sensibility** - Sets the spurious sensibility value. The default is 3 (3 x standard deviation (sigma)).

**Example of Spurious Judgement Using Spurious Sensibility**
**Min. Spur Level** - Spurious data larger than this minimum spurious level will be omitted from the spurious data.

**Omit Displayed Spur** - Omits all displayed spurs.
**Defaults** button - Restores default phase noise settings.

**Apply** button - Applies setting changes and leaves the dialog box open to make more setting changes.

---

**Integrated Noise Tab in dialog help**

Show Integrated Noise Table - Enable or disable displaying the integrated noise table below the trace display. Integrated noise is the single sideband noise integrated over a measurement bandwidth from a start/stop frequency.

<table>
<thead>
<tr>
<th>Trace</th>
<th>Start Offset</th>
<th>Stop Offset</th>
<th>Weighting</th>
<th>Integ Noise</th>
<th>FM</th>
<th>FM/AM</th>
<th>Jitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 kHz</td>
<td>10 kHz</td>
<td>Off</td>
<td>-62.10 dBc</td>
<td>5.75 m&quot;</td>
<td>11.273 kHz</td>
<td>349.32 fs</td>
</tr>
</tbody>
</table>

**Trace** - Displays trace number(s).

**Start Offset** - This is the same as the **Start Offset** defined in the **Sweep** tab.

**Stop Offset** - This is the same as the **Stop Offset** defined in the **Sweep** tab.

**Weighting Filter** - Displays the weighting filename or None for no weighting.

**Integ Noise** - Noise over a measurement bandwidth from the defined start/stop frequency.
**PM** - Phase Modulation measured in degrees and radians.

**FM/AM** - The FM/AM column displays a value depending on the measurement parameter. When the measurement parameter is Phase Noise (for example, PN_b2), the residual FM (Hz) is shown in the FM/AM column. When the measurement parameter is AM Noise (for example, AM_b2), the percentage of modulation of the carrier is shown in the FM/AM column.

**Jitter** - The phase deviation in the time domain.

**Weighting Filter Setup...** button - Accesses Weighting Filter Table Setup dialog. The weighting filters compensate external effects that affect the phase noise in the specified frequency ranges. When the weighting filters are applied, the trace data are corrected by the filter characteristics before the calculation of integrated noise. You must define at least 2 frequencies when using the Weighting Filter.

**Add** button - Adds a frequency below existing entries.

**Delete** button - Deletes the currently selected frequency in the table.

**Delete All** button - Deletes all frequencies in the table.

**Save Table** button - Saves the weighting filter table to a .csv file. The file should be saved in the following directory: C:\ProgramData\Keysight\Network Analyzer\WeightingFilter. All filters saved in this directory are shown on the drop down list in the Weighting Filter column.

**Load Table** button - Loads a previously saved weighting filter table.

**Select** - Selects an existing phase noise trace or allows you to add a new trace. Each
trace can have up to four defined ranges. See Displaying Phase Noise Parameters.

**Range** - Sets the integration range number.

**Type** - Sets the integration range type of the selected integration range defined as offset frequency from the carrier.

- **Off** - Disables specific range and will not be measured.
- **Full Span** - Enables the measurement bandwidth to be the full offset range defined in the Sweep tab.
- **Custom** - Enables custom-defined start and stop offset range.

**Start** - Start offset frequency for the integrated noise measurement.

**Stop** - Stop offset frequency for the integrated noise measurement.

**Weighting Filter** - Selects pre-defined weighting filter file. Select None when a weighting filter file will not be used.

**Defaults** button - Restores default phase noise settings.

**Apply** button - Applies setting changes and leaves the dialog box open to make more setting changes.

| Spot Noise Tab in dialog help | Programming Commands |
**Show Spot Noise Table** - Enable or disable displaying the Spot Noise Table below the trace display.

<table>
<thead>
<tr>
<th>Trace</th>
<th>10 Hz</th>
<th>100 Hz</th>
<th>1 kHz</th>
<th>10 kHz</th>
<th>100 kHz</th>
<th>1 MHz</th>
<th>10 MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>56.123 dBc/Hz</td>
<td>1.071 dBc/Hz</td>
<td>118.77 dBc/Hz</td>
<td>198.01 dBc/Hz</td>
<td>121.16 dBc/Hz</td>
<td>138.44 dBc/Hz</td>
<td>122.20 dBc/Hz</td>
</tr>
</tbody>
</table>

**Trace** - Displays trace number(s).

**Spot Frequency** columns - Displays decade edges and/or specified offset spot frequencies.

**Select Traces** - Select traces for calculating the Spot Noise.

**New Traces...** button - Add a new trace for calculating Spot Noise.

**Define Spot Frequencies** - Select the check boxes and enter specific spot frequencies to measure. A trace must be selected. If **Decade Edges** is selected, the decade edges will also be measured. Duplicate frequencies between Decade Edges and Spot Frequencies are shown only once.

**Decade Edges** - Select to measure the decade edges based on the **Start Offset** and **Stop Offset** settings defined in the Sweep tab. Deselect to measure only the defined/selected spot frequencies. A trace must be selected.

**Defaults** button - Restores default phase noise settings.

**Apply** button - Applies setting changes and leaves the dialog box open to make more setting changes.
Displaying Phase Noise Parameters

Trace Measurement Parameters

How to select and configure Measurement Parameters

Using Hardkey/SoftTab/Softkey

1. Press Trace > Trace Setup > Measure....
2. Select a parameter.

Using a mouse

1. Click Instrument.
2. Select Trace.
3. Select Measure....
4. Select a parameter.

**PN_b2** - Integrated phase noise measured at VNA input b2 (Test Port Receiver 2).
**AM_b2** - Amplitude modulation measured at VNA input b2 (Test Port Receiver 2). This parameter measures the percentage of the modulation of the carrier.

**Note:** The VNA input receiver number can be changed using the **VNA Input** function in the **RF Path** tab or by pressing **Meas > Main > VNA Input**.

**New Trace** - Add a new trace.

**Channel N** - Select a channel number for the new modulation distortion trace.

**Window N** - Select to create the new trace in an existing window or new window.

**Select All** - Select all measurement parameters.
Setting Up a Phase Noise Measurement

The following procedures describe how to set up a phase noise measurement using the Option S93031xB Phase Noise application.

In this topic:

- Hardware Setups
- Create a Phase Noise Channel
- Open the Phase Noise Setup dialog
- Set Up the Carrier
- Set Up the Measurement
- Set Up the RF Path
- Set Up the Source
- Set Up an External Source

Hardware Setups

The following are typical hardware setups for measuring the phase noise of a device. The Option S93031xB Phase Noise application is installed on the PNA/PNA-X.
Create a Phase Noise Channel
1. On the VNA front panel, press **Meas > S-Param > Meas Class...**.

2. Select **Phase Noise**, then either:
   - **OK** delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.

3. In the **Confirm Measurement Class Change** dialog, click **OK** to proceed or **Cancel** to exit.

4. The **Phase Noise Setup** dialog is displayed.

5. The default Phase Noise Measurement Parameter is **PN_b2**. This means that the displayed trace is a Phase Noise measurement trace. The "b2" corresponds to the VNA receiver input to use for the phase noise measurement.

---

### Open the Phase Noise Setup dialog

1. If the Phase Noise Setup dialog is not displayed, press **Freq > Main > Phase Noise Setup...**.

2. The **Sweep**, **RF Path**, **Source**, **Spurious**, **Integrated Noise**, or **Spot Noise** tab can now be selected.
**Set Up the Carrier**

1. In the **Sweep** tab, set the **Carrier Frequency** by either using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad. This is the nominal value of the carrier frequency.

2. Set the **Carrier Threshold** by either using the up/down arrows or by double-clicking in the data entry field and entering the level using the displayed keypad. The carrier will be detected using this threshold value and will be compared to the specified Carrier Frequency. The delta indicator displays any frequency difference.

**Set Up the Measurement**

1. In the **Sweep** tab, set the **Start Offset** and **Stop Offset** by either using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad. This is an offset relative to the carrier frequency to position the measurement about the noise sideband to measure power density in dBc/Hz.

2. Set the **RBW Ratio** by either using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad. The RBW Ratio sets the resolution bandwidth ratio, which is the specified resolution bandwidth percentage of every half decade offset frequency.

3. Set the **FFT Avg Factor** by either using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad. The average factor is multiplied by the default average count, which changes for each frequency range.

4. Select the desired **Noise Mode** using the drop down menu. Select **Fast**, **Normal**, or **Best**. The **Fast** selection has the highest noise floor and fastest measurement speed. The **Best** selection is the lowest measurement speed with lowest noise floor. Refer to the following example showing **Fast**, **Normal**, and **Best** differences:
Set Up the RF Path

1. In the RF Path tab, select the VNA Input using the drop down menu. This selects the VNA receiver input to use for the phase noise measurement. Only one can be chosen.

2. Set the Rcvr Atten using the drop down menu. This sets the VNA receiver attenuation of the selected VNA receiver.

3. Click on the RF Path Configuration... button to configure hardware paths that are available with selected PNA/PNA-X options as necessary. This ensures that the correct path has been selected for the source (for converter measurements) and receiver. Learn more.

Set Up the Source
Set up a source when the DUT requires an RF source at its input.

1. In the **Source** tab, select **ON** in the **State** column next to the desired source displayed in the list. This is the source used at the input of the DUT.

2. Set the **Frequency** for the selected source by clicking in the data entry field, clicking on the **Edit** button, then entering the frequency by either using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad.

3. Set the **Power** for the selected source by clicking in the data entry field, clicking on the **Edit** button, then entering the power by either using the up/down arrows or by double-clicking in the data entry field and entering the power using the displayed keypad.

4. Click on the **Power and Attenuator...** button to define and control the source power and attenuation. Learn more

5. Click on the **RF Path Configuration...** button to configure hardware components that are available with selected PNA/PNA-X options. Learn more.

---

**Set Up an External Source**

An external source can also be configured and selected as the source. Before the external source is displayed in the list it must be set up as shown in the following procedure.
1. Click on the **External Devices**... button.

2. Click on the **New** button.

3. Click in the **Name** field and type a name for the source. For example, **myMXG**.

4. For **Device Type**, select **Source**.

5. Select the **Driver**. For example, **MXG_Vector**.

6. Select **Active - Show in UI**.

7. Ensure that **Enable IO** is checked.

8. In the **I/O Configuration** field, type the VISA address of the external source.

9. Click on the **OK** button. The following is an example:

![External Device Configuration: myMXG](image)

10. The external source should be displayed in the list of sources displayed in the **Source** tab and can be selected using the Set Up the Source procedure.
Spurious Measurement

In this topic:

- Spurious Criteria
- Define a New Trace
- Setting Minimum Spurious Level
- Setting Spurious Sensibility
- Setting Omit Displayed Spur
- Displaying the Spurious Table
- Enable Spur Analysis

Spurious Criteria

The criteria used for a spurious measurement defines the threshold in terms of the standard deviation (σ) value which is calculated from the measurement result moving average.

The threshold can be set as the standard deviation (Γ) X sensibility. When the measured value exceeds the threshold, it is defined as a spur.

Define a New Trace

When the Phase Noise Measurement Class is selected and a Phase Noise channel is created, the default measurement parameter is PN_b2. This indicates that the trace is a Phase Noise measurement that will be made at the b2 receiver of the VNA. The VNA receiver can be changed using the VNA Input function in the RF Path tab. Each defined trace can have independent spurious settings.

1. In the Spurious tab, click on the Select drop down menu and select New Traces...

2. In the New Trace dialog, select PN_bN, where "N" is the VNA receiver number set using the VNA Input function in the RF Path tab.

Setting Minimum Spurious Level
If the spurious data is exceeds the minimum spurious level, it is omitted from the spurious data, or it is displayed as Power (dBc).

For example, there are two peaks (C) and (D) which are detected as a spurious in the figure below. When Power (dBc) is displayed, the peak (C) which is above the minimum spurious level turns white in color.
1. In the **Spurious** tab, click on the **Select** drop down menu and select the desired trace.

2. Set the **Min. Spur Level** by either using the up/down arrows or by double-clicking in the data entry field and entering the level using the displayed keypad.

**Setting Spurious Sensibility**

The following procedure describes how to set spurious sensibility.
1. In the Spurious tab, click on the Select drop down menu and select the desired trace.

2. Set the Spur Sensibility by either using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad.

Setting Omit Displayed Spur

This function will omit all displayed spurs.

1. In the Spurious tab, click on the Select drop down menu and select the desired trace.

2. Check Omit Displayed Spur. The following shows two spurs displayed in the top trace and, after checking Omit Displayed Spur, the bottom trace shows that the spurs are omitted.

Displaying the Spurious Table
The Spurious Table can be displayed below the trace area of the display showing Trace and Spur number, Offset frequency, Power level, and Jitter.

<table>
<thead>
<tr>
<th>Trace</th>
<th>Spur</th>
<th>Offset</th>
<th>Power</th>
<th>Jitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1</td>
<td>10 kHz</td>
<td>-15.228 dBc</td>
<td>38.998 ps</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>20.0543795 kHz</td>
<td>-76.584 dBc</td>
<td>33.352 fs</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>29.8920401 kHz</td>
<td>-76.910 dBc</td>
<td>32.124 fs</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>49.8629281 kHz</td>
<td>-78.318 dBc</td>
<td>27.317 fs</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>950.118507 kHz</td>
<td>-82.061 dBc</td>
<td>17.753 fs</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>987.954269 Hz</td>
<td>-91.471 dBc</td>
<td>6.0090 fs</td>
</tr>
</tbody>
</table>

1. In the Spurious tab, check Show Spurious Table.
2. To sort the table, click on the Table Sort Order drop down menu and select Power to sort by power or Offset to sort by offset frequency.

Enable Spur Analysis

This function enables spurious analysis search markers. See Spurious Search.

1. In the Spurious tab, click on the Select drop down menu and select the desired trace.
2. Check Enable Spur Analysis then click OK to exit the dialog.
3. On the VNA front panel, press Search > Spurious > Spurious Search. A marker is displayed over the first spur:

4. Note the frequency and level from the carrier are displayed in the top-right corner of the display.
5. To move the marker to the next spur on the right, press Spurious Right >> Search.
6. To move the marker to a spur on the left, press **Spurious Left >> Search**.

7. To place markers on all displayed spurs, press **Multi Spurious Search**.

8. To turn off all markers, press **Marker > Marker Setup > All Off**.
Integrated Noise Measurement

Integrated noise is the single sideband phase noise integrated over a measurement bandwidth from a start/stop offset frequency. Up to four offset frequency ranges can be defined.

In this topic:

- Define a New Trace
- Set Up Measurement Ranges
- Set Up Weighting Filter
- Displaying the Integrated Noise Table

Define a New Trace

When the Phase Noise Measurement Class is selected and a Phase Noise channel is created, the default measurement parameter is \( \text{PN\_b2} \). This indicates that the trace is a Phase Noise measurement that will be made at the b2 receiver of the VNA. The VNA receiver can be changed using the VNA Input function in the RF Path tab. Each defined trace can have independent settings.

1. In the Integrated Noise tab, click on the Select drop down menu and select New Traces... .
2. In the New Trace dialog, select \( \text{PN\_bN} \), where "N" is the VNA receiver number set using the VNA Input
The following procedure describes how to set up multiple measurement ranges.

1. In the Integrated Noise tab, click on the Select drop down menu and select the desired trace.

2. Click in the Type column for Range 1, click on the down arrow, then select Custom.

3. In the Start column for the range, set the frequency by either using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad.

4. In the Stop column for the range, set the frequency by either using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad.

5. Repeat steps 2 through 4 to add more ranges (up to four ranges).

6. If Weighting Filter files have been set up and saved, click in the Weighting Filter column for a range, click on the down arrow, then select the file. See below for a procedure on how to set up a Weighting Filter file.
The weighting filter allows weighting values to be entered at specified frequencies. The weighting filters compensate external effects that affect the phase noise in the specified frequency ranges. When the weighting filters are applied, the trace data are corrected by the filter characteristics before the calculation of integrated noise.

1. In the Integrated Noise tab, click on the Weighting Filter Setup... button.

2. Click on the Add button.

3. In the Frequency column, set the frequency by either using the up/down arrows or by double-clicking in the data entry field and entering the frequency using the displayed keypad.

4. In the Weighting Value column, set the value by either using the up/down arrows or by double-clicking in the data entry field and entering the value using the displayed keypad.

5. Repeat steps 2 through 4 to add more weighting values.

6. Click on the Save Table button to save the settings to a file (.csv).

Displaying the Integrated Noise Table

The Integrated Noise Table can be displayed below the trace area of the display showing trace number, start/stop offset, weighting filename (if used) integrated noise, phase modulation measured in degrees and radians, residual FM (Hz) or residual AM (%), and jitter. Residual FM (Hz) will be displayed when the PN_b2 Phase Noise measurement parameter is selected; residual AM (%) will be displayed when the AM_b2 Phase Noise measurement parameter is selected.
1. In the Integrated Noise tab, check Show Integrated Noise Table.
Spot Noise Measurement

Spot Noise measurement are measurements made at specific frequencies. The measurements can be made at each decade, determined by the start/stop offset frequencies, up to six defined frequencies, or a combination of both decade and defined frequencies.

In this topic:

- Define a New Trace
- Define Spot Frequencies
- Displaying the Spot Noise Table

Define a New Trace

When the Phase Noise Measurement Class is selected and a Phase Noise channel is created, the default measurement parameter is PN_b2. This indicates that the trace is a Phase Noise measurement that will be made at the b2 receiver of the VNA. The VNA receiver can be changed using the VNA Input function in the RF Path tab. Each defined trace can have independent settings.

1. In the Spot Noise tab, click on the New Traces... button.
2. In the New Trace dialog, select PN_bN, where "N" is the VNA receiver number set using the VNA Input function in the RF Path tab.
Define Spot Frequencies

The following procedure describes how to define measurements at specific frequencies and/or at the decade edges. The decade edges are determined by the Start Offset and Stop Offset settings in the Sweep tab.

1. In the Spot Noise tab, select the desired trace displayed under Select Traces.

2. To measure phase noise at each decade, check Decade Edges and leave the other spot frequencies unchecked.

3. To define specific frequencies to measure other than at the decade edges, perform the following steps:
   a. Uncheck the Decade Edges check box.
   b. In the data entry field, set the spot frequency to measure (up to six) by either using its up/down arrows or by double-clicking in its data entry field and entering the frequency using the displayed keypad.
   c. Check the corresponding box to enable the measurement at this specific frequency.
   d. Repeat steps a through c for each frequency to measure.

Displaying the Spot Noise Table
The Spot Noise Table can be displayed below the trace area of the display showing the trace number and corresponding measurement frequencies and the phase noise in dBc/Hz.

<table>
<thead>
<tr>
<th>Trace</th>
<th>10 Hz</th>
<th>100 Hz</th>
<th>1 kHz</th>
<th>10 kHz</th>
<th>100 kHz</th>
<th>1 MHz</th>
<th>10 MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>-58.123 dBc/Hz</td>
<td>-107.14 dBc/Hz</td>
<td>-119.77 dBc/Hz</td>
<td>-119.01 dBc/Hz</td>
<td>-121.16 dBc/Hz</td>
<td>-136.44 dBc/Hz</td>
<td>-133.20 dBc/Hz</td>
</tr>
</tbody>
</table>

1. In the **Spot Noise** tab, check **Show Spot Noise Table**.
AM Noise Measurement

Noise that modulates the phase of the carrier is referred to as amplitude modulation of the carrier. The following procedure describes how to measure AM Noise.

1. Set up a Phase Noise measurement as described in Setting Up a Phase Measurement.

2. On the VNA front panel, press Meas > Main > AM_b2. The b2 is the VNA input and could be different depending on the hardware configuration.

3. To create a new AM Noise trace, press Meas > Main > Other. The Measure dialog is displayed:

4. In the Measure dialog, check AM_b2 AM Noise.

5. In the Measure dialog, check New Trace then select the desired Channel and Window.

6. The following example displays an AM measurement at the b4 VNA input:
7. The FM/AM column displays the percentage of modulation of the carrier.
The Spectrum Analyzer function is available with Sx090A/B.

In this topic:

- Features, Requirements, and Limitations
- Specificities versus legacy Spectrum Analyzer setups
- Spectrum Analyzer Setup Overview
- How to make SA Measurements
  - SA Setup Dialog
  - Source Setup Dialog
  - Coherence Setup Dialog
  - Trig. & Pulse Dialog
  - Advanced Settings Dialog
    - IF Dialog
    - Processing Dialog
    - ADC & LO Dialog
    - Data Dialog
  - Source Frequency Settings Dialog
  - Source Power Settings Dialog
  - Measurement Parameters
- Marker -> SA
- SA Analysis Markers (Separate topic)
- Calibrating an SA Channel
- Gated SA
- SA Warning Messages
- Spectrum Analyzer Measurement Examples (Separate topics)
  - Amplifier Harmonics Measurement
  - Converter Spurious Measurement
  - Gated Measurement

See Also

Programming commands
Spectrum Analyzer mmWave Measurements

Noise Power Ratio (NPR) Measurement

Other VNA Applications

Features, Requirements, and Limitations

Features

- General purpose spectrum analysis for component measurements
- Add multi-channel spectrum analysis (multiple receivers/frequencies)
- Perform fast spurious search in broad frequency band
- Enable spectrum analysis at calibration plane using VNA calibration and Fixturing (de-embedding)
- Single-connection, multiple measurements
- Internal, manual, and external triggering - learn more
- Marker -> SA available in Standard, SMC, and Swept IMD channels
- Gating - learn more
- Coherence Image Rejection for multi-tones and repetitive test signals
- Export data
- All VNA are supported.
- Broadband and Banded millimeter-wave systems are supported - learn more
- External DC meter configuration is supported for a SA measurement channel.

Requirements

- Spectrum Analyzer Option S930900A/B.
- Windows 7 or Windows 10 operating system
- Option S930909A/B (10 MHz to = 90 GHz) for broadband spectrum analyzer millimeter-wave measurements
- Option S93093A/B (10 MHz to = 120 GHz) for broadband spectrum analyzer millimeter-wave measurements
- Option S93094A/B (> 110 GHz) for banded spectrum analyzer millimeter-wave measurements
- IF Response Adjustment required after installing Option S93090xA/B, S93093A, or S93094A/B

Limitations

- For PNA-L 4-port models, only one reference receiver can be used at a time
For PNA-L models, millimeter-wave measurements are not available

**Specificities versus legacy Spectrum Analyzer setups**

This Spectrum Analyzer application works on VNA hardware:

- There is no hardware pre-selector filter (it used to be a slow Yig filter in legacy Spectrum analyzer hardware).
- The IF bandwidth is limited by hardware design to 30 MHz (can be increased with advanced settings to 38 MHz). This is because PNA ADCs are baseband running at 100 MHz.
- Image rejection in IF band is not based on hardware filtering but on software comparison of different acquisitions of the same RF frequencies with different analyzer internal LOs.

This has some consequences:

- Sweep time is very fast for low resolution bandwidth compared to legacy spectrum analyzers (they have to wait for Yig filter stabilization).
- Real time wideband signals can only be accurately acquired if they fit in the instantaneous IF bandwidth: 30 MHz. Real communication signal or random noise signals can then be measured with no Image rejection mode (None, LO Low or None, LO High Image Reject Type).
- Repetitive wideband test signals can be measured accurately if the RBW is set low enough, or if the Coherent mode is enabled. Then, any signal bandwidth is possible, up to several GHz wide. If the coherent mode is not possible, then we recommend to set the RBW equal or lower to 1/10 of 1/signal repetition rate. Example: for a 1 ms repetitive test signal, set the RBW to 100 Hz or lower.

**Note:** The wideband repetitive test signal case is a very important and common one, as a VNA is a component test tool, not a signal analysis tool. NPR, and ACPR measurements of active devices can be performed accurately with this Spectrum Analyzer Application.

**SA Setup Overview**

2. Set up the SA source frequency and power.
3. Define the measurement parameters.
4. Select markers.
5. Calibrate the SA channel using the Cal All wizard.

**How to make SA Measurements**
Create a Spectrum Analysis Channel

1. On the VNA front panel, press Meas > S-Param > Meas Class... .

2. Select Spectrum Analysis, then either:
   - OK delete the existing measurement, or
   - New Channel to create the measurement in a new channel.

3. A Spectrum Analysis measurement is displayed.

Configure SA settings

Using Hardkey / SoftTab / Softkey

1. Press Freq > Main > SA Setup... .

Using a mouse

1. Click Stimulus
2. Select SA Setup...

SA Setup Dialog tab help
**Sweep Type** - Sets the spectrum analysis sweep type. See Type (Sweep).

**X-Axis Point Spacing** - Enables or disables the separate segment sweeps in a Dual-Band Configuration for frequency conversion measurements. The span increases to display input and output signals.

**Show segments** - Displays the segment table at the bottom of the display. Number of points for each segment cannot be specified.

**Hide segments** - Hides the segment table.

**Processing**

**Resolution Bandwidth** - Provides the ability to resolve, or see closely spaced signals. The narrower (lower) the Resolution Bandwidth, the better the spectrum analyzer can resolve signals. In addition, as the Resolution Bandwidth is narrowed, less noise is measured by the spectrum analyzer ADC and the noise floor on the display lowers as a result. This allows low level signals to be seen and measured. However, as the Resolution Bandwidth is narrowed, the sweep speed becomes slower.

**Auto** - Check to couple Resolution Bandwidth to the frequency span in a ratio based on the Span/RBW setting. As the frequency span is narrowed, the Resolution Bandwidth is also narrowed providing increased ability to resolve signals. Clear to uncouple the settings.

**Video Bandwidth** - Sets the video averaging factor. The averaging operation is applied after the DFT (Discrete Fourier Transform) and before the image rejection. The trace data is smoothed with the method selected by the Video Averaging Type. More smoothing occurs as the Video BW is set lower. However, as the Video BW is narrowed, the sweep speed becomes slower. The Video Bandwidth can be set from 3 Hz to 3 MHz when Auto is deselected.

**Auto** - Check to couple the Resolution Bandwidth to the Video Bandwidth in a ratio based on the RBW/VBW setting. Clear to uncouple the settings.

**Detector Type** - A "detector" is an algorithm used to map DFT bins into display buckets. There are typically several DFT bins in a single display bucket, and the detector determines how to translate the multiple DFT values into a single display value.

**Peak** - Displays the maximum value of all the measurements in each bucket. This setting ensures that no signal is missed. However, it is not a good representation of the random noise in each bucket.

**Average** - Displays the Root Mean Squared (RMS) average power of all the measurements in each bucket. This is the preferred method when making power measurements.
**Sample** - Displays the center measurement of all the measurements in each bucket. This setting gives a good representation of the random noise in each bucket. However, it does not ensure that all signals are represented.

**Normal** - Provides a better visual display of random noise than Positive peak and avoids the missed-signal problem of the Sample Mode. Should the signal both rise and fall within the bucket interval, then the algorithm classifies the signal as noise. An odd-numbered data point displays the maximum value encountered during its bucket. An even-numbered data point displays the minimum value encountered during its bucket. If the signal is NOT classified as noise (does NOT rise and fall) then Normal is equivalent to Positive Peak.

**NegPeak** - Displays the minimum value of all the measurements in each bucket.

**Peak Sample** - Attempts to determine if the display bucket contains an actual signal, or just noise. If a signal is present, the Peak detector is used, otherwise Sample is applied.

**Peak Average** - Attempts to determine if the display bucket contains an actual signal, or just noise. If a signal is present, the Peak detector is used, otherwise Average is applied.

**Bypass** - Check to bypass the Detector Type to view all display points from the DFT. This is only available if the total number of DFT points can be handled by the display.

**Video Averaging Type** - Determines how to compute the video average. When Auto is selected, the optimum type of averaging for the current instrument measurement settings is selected. It averages the magnitude of the DFT bins. Averaging only applies if the video bandwidth is less than the resolution bandwidth.

**Voltage** - Selects averaging of the detected signal's magnitude and returns the result.

**Power** - Selects averaging of the detected signal's squared magnitude and returns the square root of the result.

**Log** - Selects averaging of the detected signal's natural logarithm of the magnitude and returns the exponentiated value of the result.

**Voltage Max** - Returns the maximum voltage (signal magnitude) measured during the averaging period.

**Voltage Min** - Returns the minimum voltage (signal magnitude) measured during the averaging period.

**Averaging Count** - Reads the number of Video bandwidth sweeps that are averaged together. This readout is displayed to the right of the **Averaging Type** selection (the small "1" shown in
the dialog above). It can be read with the remote interface using the SENS:SA:BAND:VID:AVER:COUNt? command.

**Settings**

Sets the SA (receiver) frequency range when running Linear Frequency sweep type. Use either of the following pairs of settings to determine the frequency range.

**Start /Stop** - Specifies the beginning and end frequency of the swept receiver range. Start is the beginning of the X-axis and Stop is the end of the X-axis. When the Start and Stop frequencies are entered, then the X-axis annotation on the screen shows the Start and Stop frequencies.

**Center /Span** - Specifies the value at the center and frequency range. The Center frequency is at the exact center of the X-axis. The Frequency Span places half of the frequency range on either side of center. When the Center and Frequency Span values are entered, then the X-axis annotation on the screen shows the Center and Span frequencies.

**Number of Points** - Selects the number of trace points on the display. When the Detector is bypassed, the number of display points is read only, it shows the current DFT points to cover the RF span.

*Note:* When running Segments, the frequency ranges are set by the segment table.

**Attenuators**

Receiver Attenuation is used to protect the test port receivers from damage or compression. Receiver attenuation causes the applied power at the receiver to be less than the power at the test port by the specified amount of attenuation.

Receiver Attenuators are offered as an option. Learn more.

Type or select independent attenuation values for each test port receiver.

A preference can be set to mathematically offset (or NOT) the reported power at the test port receivers by the amount of receiver attenuation. By default, All VNA models offset the display. Learn how to set the preference.
**Power On (All Channels)** Check to enable source power for all channels. Only turns power ON if the port State setting is ON.

**Port Powers Coupled**

- **Coupled** (checked) The power levels are the same at each test port. Set power at any test port and all test ports change to the same power level.

- **Uncoupled** (cleared) The power levels are set independently for each test port. Uncouple power, for example, if you want to measure the gain and reverse-isolation of a high-gain amplifier. The power required for the input port of the amplifier is much lower than the power required for the output port. A power sweep can also be performed with uncoupled power. Learn more about Setting Independent Port Power.

**Source Cells**

**Name** - Lists the test ports through which an internal source is available. If an external source has been configured, it will appear at the bottom of the list.

**State**

- **ON** Source power is ALWAYS ON. Only one port can be turn on unless the unit has the second source. Learn about internal second source restrictions.
- **OFF** Source power is never ON, regardless of the measurement requirements. Use this setting to prevent damage to a sensitive DUT test port.

- **No Control** Available ONLY on external sources. The SA application will NOT control the external source.

**Type (Sweep)**

- **CW** - The source is set to a CW frequency.

- **LinFreq** - The source is set to sweep from the Start to Stop frequency.

- **Power** - The source is set to a power sweep.

- **Phase** - The source is set to a phase sweep.

- **LinF+Pwr, Freq+Pha, Pow+Pha, Fr+Pw+Ph** - Multi-parameter sweep for the specified parameters. Set the range of sweep for each parameter.

**Frequency** - Click in the cell, then click **Edit**, to start the Frequency Settings dialog (below).

**Power** - Sets the power level at the output of the source. Click in the cell, then click **Edit**, to start the Power Settings dialog (below).

**Phase** - Control source phase of a VNA source or external source. Learn more.

**Pulse** - Enable/disable pulse measurements. Learn more.

**IQMod.** - Modulated I/Q file. Clicking Edit accesses a dialog for setting up an I/Q modulated file that is sent to a source for measuring Noise Power Ratio (NPR). Refer to the Noise Power Ratio (NPR) Settings topic for descriptions of the modulation dialogs.

**RF source sweep order**

- **Frequency Power Phase** - Sweep from Start to Stop frequency first then sweep power then sweep phase.

- **Power Frequency Phase** - Sweep power first then sweep from Start to Stop frequency then sweep phase.

- **Phase Frequency Power** - Sweep phase first then sweep from Start to Stop frequency then sweep power.

- **Phase Power Frequency** - Sweep phase first then sweep power then sweep from Start to Stop frequency.

- **Frequency Phase Power** - Sweep from Start to Stop frequency first then sweep phase then sweep power.
power.

Power Phase Frequency - Sweep power first then sweep phase then sweep from Start to Stop frequency.

**Buttons**

**Path Configuration**  Learn more

**Power and Attenuator**  Learn more

**External Devices**  Learn more

---

**Multitone**

The set of Multitone properties enable an enhanced Image Rejection mode that takes benefits of the stimulus test signal knowledge.

If the stimulus test signal is repetitive with a repetition rate of $x$ seconds, it only contains tones that are on a $1/x$ frequency grid. Example: if the test signal out of an arbitrary wave generator repeats every 1 ms, then it only contains frequencies on a 1 kHz grid (noise is not considered here). We will take this into account here to make the PNA-SA DFT analysis grid landing exactly on the same grid.
Moreover we will use the test signal knowledge to adjust the PNA-SA LO frequencies in order to avoid having 2 tones from the multitone stimulus signal landing at the same location at the PNA IF side. This makes the image rejection process deterministic.

**Enable multitone image rejection**

- **Enabled** (checked) The other parameters of the Multitone dialog are taken into account. Enabling this mode will set the DFT mode to arbitrary, the RBW shape to No Window, the RBW grid to a set of suitable values, and the list of analysis LOs accordingly. This setting is not compatible with the advanced settings “Force ADC record size” or “Force LO to frequency”.

- **Disabled** (cleared) Legacy stochastic image rejection mode of the PNA-SA application. When disabled, the window type is set back to what it was before enabling, and the RBW list is also set to the previous setting.

**Tone Spacing** - The tone spacing of the multitone signal (Hz).

**Waveform Period** – 1 over the tone spacing. This is the test signal repetition rate (seconds).

**Reference Tone** – If the multitone grid does not start from 0 Hz, its offset is set here. To make this more convenient, this dialog accepts as well the frequency of any tone of the multitone grid (Hz).

**Reject up to harmonic** – Set the number of test signal harmonics you want to be protected against. This adds constraints to the list of LOs used to cover the span.

**Nyquist protect order** – Enhancement for the deterministic coherent image reject mode. It ensures the Nyquist images of the signal tones in the IF bandwidth are not falling back on top of real signal frequencies. To be able to enable Nyquist protection, the tone spacing of the coherent signal cannot be an integer divider of the ADC sampling clock (100MHz for a PNA). Enabling this option often results in a larger ADC recordsize (or a smaller DFT tone spacing) at SA receiver side.

**Vector Averaging** - Average ADC samples by the specified number (=1) in FPGA memory before the DFT processing. For example, if an ADC record size of 1,000 samples is acquired and Vector Averaging is set to 2, then 1,000 samples will be averaged to 1,000 samples and the result (1,000 samples) will be stored in FPGA memory. In other words, we acquire 2,000 samples form ADCs and send 1,000 averaged samples to the next processing stage. Vector averaging helps to reduce noise and increase dynamic range. However, this feature should only be used when the stimulus frequencies are known and coherent with the current ADC record size. A value of 1 means no averaging. Note this feature behaves like ADC Stacking+1. The maximum vector averaging value is 65536 or below. It depends on the RBW and the decimation.
Check box- Check to enable the ADC sample Vector Averaging to be specified manually.

**Note:** Vector Averaging and Video Bandwidth averaging cannot be set together. When enabling vector averaging, if the coherent mode is enabled then Video Bandwidth is turned off. The ADC Record Size x Vector Averaging must be = 64 Mega Samples (or = 32 Mega for some multiple receivers configurations).

**Note:** Vector Averaging is a great averaging method when Coherent Multitone mode is enabled. We recommend increasing it in Coherent Multitone mode instead of reducing the RBW to reduce the noise floor.

**Note:** Coherent mode (and Vector Averaging too) will work well if and only if the signal source and the PNA have their reference clocks synchronized. This is usually done by connecting a 10 MHz reference BNC cable between the signal source and the PNA.

**Note:** Vector Averaging is also known as Stacking. In fact, Vector Averaging = stacking +1.

**Data Display**

- **Show All** - Legacy behavior, shows the noise or the spurious between the tones. Note the RBW shape for multitones (No Window) makes the noise or the out of grid spurious amplitude values not accurate.

- **Zero the non-tones** - All the span frequencies that are not on the multitone grid have their amplitudes set to -200 dBm before correction. This makes band power marker measuring only the power for multitone frequencies, and not the noise power.

- **Discard the non-tones** - Deletes span frequencies that are not on the multitone grid.

**Multitone settings are valid** - Displays status of multitone settings.

**Tone Phases and Phase Stitching**

**Compute Phases** - Check to enable phase computation.

**Display Phases if Tone Power >** - Set the phase display minimum level.
ADC Triggering

**Advanced Trigger Mode** - Check to enable a measurement trigger based on the ADC Level or a period of time.

**ADC Level** - Initiate a measurement trigger event whenever the ADC level of any of the receivers at work is greater than this specified value (0 = ADC Level = 1683). A level of 100 is recommended as a default value. The ADC level is an uncalibrated value that reflects signal peak amplitude. It will detect RF energy only within the current IF bandwidth, so it would make sense to use this feature associated with a Forced LO value or a narrow SA span.

**Periodic Counter** - Initiate a measurement trigger event based on the specified period. For example, if **Periodic Counter** is set to 1,000,000, then an acquisition occurs every 0.01 sec (1,000,000 x ADC Sampling Frequency (10 nsec)).

**Trigger...** - Accesses the Trigger dialog for setting up triggering.

**Hold** - The channel accepts NO trigger signals.

**Single** - The channel accepts ONE trigger signal, then goes into Hold.

**Continuous** - The channel accepts an infinite number of trigger signals.

**Pulse Measurements**
Off - Turns pulse off.

Standard Pulse - Turns pulse on.

Duty cycle - Displays the duty cycle relative to the current settings for Pulse Period and Pulse Width (Pulse Width/Pulse Period).

Meas. Width - Sets the time the measurement pulse is on.

Pulse Period pulldown - Allows the selection of the following:

- **Pulse Period** - The time to make one complete pulse.
- **Pulse Frequency** - The reciprocal of Period (1/ Period).
- **Pulse Duty Cycle** - **Pulse Width** divided by the **Pulse Period**.

RF Pulse Width - Sets the width of the RF Source pulse.

One LO Acquire - The ADC acquisition time required to measure the data for a single FFT.

Full Sweep - Complete cycle time of measurement including background sweeps.

Pulses per LO - Number of pulses which will occur during the sweep for each LO.

Pulse Details

Name - Pulse generator outputs in numerical order.

Device - Indicates the device being controlled by the pulse generator output.

- **Pulse0** - Always set to **Receiver**. Sets the amount of time to wait before triggering the ADC to begin acquisition and is always selected for Pulse0 and cannot be changed. Pulse0 adds (ADC Delay) + (Modulator Delay). The ADC will begin measuring data 250 ns before the rising edge of Pulse0. This delay is indicated in the **Fixed ADC Delay = 250 ns** annotation.

- **Pulse1** through **Pulse4** - Pulse outputs can be set to the following:

  - **RF Src** - Selecting **RF Src** indicates that the pulse signal is used to drive the RF modulator. Only one pulse generator output can be used to drive an RF source. If you try to set more than one pulse generator output to **RF Src**, then the other one will be set to **User N** (where "N" is the pulse generator number).

  - **User 1, User 2, User 3, User 4** - Labels for user convenience. These labels do not connect the pulse generator to any specific hardware. These selections may be used to control a DUT, DC biases, or other signals.
- **Activity** - This selection outputs a signal on Pulse4 when the ADC is active. This is the same as Pulse4 Output Indicates ADC Activity on the Pulse Generators Setup dialog. If ADC is selected for Pulse4, then the pulse width and delay entries are grayed out because Pulse4 is no longer a pulse output.

**Width** - RF Source pulse width. This setting is the same as **Pulse Width** under RF Pulse.

**Meas Delay** - Sets the time before each pulse begins.

**Width Adjust** - Adjusts the measurement pulse width.

**Invert** - Check to cause the pulse ON time to be active low and OFF to be active high.

**Enable** - Check to enable individual pulse generators.

**Plot Pulse Timing...** - Accesses Pulse Timing display to evaluate the pulse timing setup.

**Pulse Generators...** - Accesses the Pulse Generator Setup dialog for setting up pulse measurements.

---

### Advanced Settings Dialog tab help

![Advanced Settings Dialog](image)

**Properties**

![Properties](image)
**RBW Shape** - Selects the digital filter (window) to apply to the time domain IF signal. The filter effectively "shapes" the signal before application of the DFT to help avoid discontinuities which add unwanted frequency content to the spectrum. Each filter has its own advantages and disadvantages.

**Gaussian** - Selects a Gaussian window. The Gaussian window has good frequency separation and moderate amplitude accuracy. This window provides higher dynamic range because it has much lower side lobes. It is used for general-purpose measurements and when high dynamic range is required.

**Flat Top** - Selects the flat top window for amplitude measurement of sinusoidal frequency components. The flat top window has moderate frequency separation and excellent amplitude accuracy. It is typically used for narrowband signals when measuring the amplitude of a particular frequency component with greater amplitude accuracy.

**Kaiser** - Selects the Kaiser window which is an approximation of a Slepian window using Bessel functions. This window has a relatively high dynamic range and is similar to the Blackman window.

**Blackman** - Selects the Blackman window. This window has a relatively high dynamic range and is similar to the Kaiser window.

**No Window** - The No Window selection does not modify the time-domain data in any way before applying the DFT. This selection is very fast but may yield a significant number of side lobes in the frequency domain because of spectral leakage. This selection has a rectangular shape and does not attenuate any portion of the time record.

**Image Reject Type** - Sets the minimum number of distinct DFT acquisitions to use when computing an actual signal. As the number of DFT acquisitions increases from the None, LO Low setting to the Max setting, an increased number of erroneous signals are eliminated. Therefore, the Better and Max settings provide the highest confidence that what remains are actual signals, at the expense of slower measurements.

**None, LO High** - Selects 1 acquisition with the LO higher than the receiver frequency.

*Note: Selecting None, LO High with full span is not possible. See SA Warning Messages.*

**None, LO Low** - Selects 1 acquisition with the LO lower than the receiver frequency.

*Note: Selecting None, LO Low with full span is not possible. See SA Warning Messages.*

**Min** - Selects 2 acquisitions.

**Min, LO High** - Selects 2 acquisitions (like Min) and both acquisitions consider that the LO
is higher than the receiver frequency.

**Min, LO Low** - Selects 2 acquisitions (like Min) and both acquisitions consider that the LO is lower than the receiver frequency.

**Normal** - Selects 4 acquisitions.

**Better** - Selects 6 acquisitions.

**Max** - Selects 8 acquisitions.

**Image Reject Strength** - Sets the image rejection strength. During the image rejection process, several LO acquisitions overlap at the same RF frequency (depending on the Image Reject Type). As a result, different RF signal values can be returned. This feature sets the acceptable power differences between measurements performed with different LOs in determining actual signals. Possible values are Weak, Normal, Strong. Weak accepts more difference between measurements, and strong less difference.

**RBW/VBW** - Sets the ratio of Resolution Bandwidth to Video Bandwidth when the Video Bandwidth is in Auto mode.

**Span/RBW** - Sets the ratio of Span to Resolution Bandwidth when the Resolution Bandwidth is in Auto mode.

**CF Step Size** - Manually sets the amount Center frequency change that occurs when $\Delta f$ is clicked (next to the value).

- **Auto** - Each press of the $\uparrow\downarrow$ arrows increments or decrements the Center frequency by 5% of the current frequency span.

**Occupied BW search min** - Sets the minimum search frequency to use during an Occupied BW search measurement. Power below this frequency is ignored. See Occupied BW Ratio for information about setting up this measurement type.

**DC Sources**

DC source control allows the spectrum to be measured at multiple DC source settings.

**Enable DC Outputs** - Enables all DC source outputs that are turned ON in the DC Source dialog. This same selection is found in the DC Source Dialog.

**Enable DC Sweep** - Enables the DC sources to sweep between their start and stop voltages. If not selected, then the DC sources will be set to their start voltages.

**Number of DC levels** - Defines the number of voltage levels in the DC sweep.

The following settings apply to the measurement loop order. The SA may be programmed to loop through a series of spectrum measurements at multiple RF source frequencies,
multiple RF source powers, and multiple DC voltages. These radio buttons determine whether the DC sources are swept before the RF power and frequencies are swept, or whether the DC sources are swept after the RF power and frequencies are swept.

**Sweep Order**

**DC before RF** - Sweep through each DC voltage step first then sweep through the next frequency.

**RF before DC** - Sweep through each frequency step first then sweep through the next DC voltage.

**External Devices...** button - Learn more.

**DC Sources...** - Configure internal DC sources. Learn more.

**Dual-Band Configuration**

The dual band configuration adds support for frequency conversion measurements. The following shows a down-converter measuring lower and upper ACPR and NPR. For this example, the RF input is 3.1 GHz on Port 1 from an N5182A MXG, the mixer IF output is 2.2 GHz on Port 2, and the mixer LO input connected to Port 4 is 900 MHz.

**Note:** The frequency conversion measurements also supports power sweep types to sweep from a start to a stop power.

![Dual-Band Configuration Diagram]

**Enable segments auto configuration** - Automatically configures separate segment sweeps to display both input and output signals regardless of the frequency separation. In the above down-
converter example, the segment sweep for the output (signal on the left) was set to 2.04 GHz to 2.36 GHz while the input segment sweep (signal on the right) was set to 2.94 GHz to 3.26 GHz. This was configured automatically when the check box is checked.

**Band** - Displays the band numbers corresponding to the input and output.

**Source** - Selects RF signal source from the pull down menu.

**Port** - Selects the port number to assign to the input and output.

**Center** - Sets the center frequency of each band.

**Inverted** - Inverts markers if necessary to position the NPR marker in the notch.

**Band 2 center offset** - LO offset value.

**Embedded LO**

**Enable Embedded LO** - Check to enable measurements of mixers that have a fixed LO inside the DUT.

*Note:* Embedded LO does not support signals with a tone spacing of less than 1 kHz.

**Setup...** - Opens the Embedded LO dialog.
Enable Embedded LO - Check to enable measurements of mixers that have a fixed LO inside the DUT.

Tuning Method - These settings determine the amount of time spent versus the degree of accuracy to which the LO Frequency is measured. Accuracy is compromised when noise starts to appear on the measurement trace.

Broadband and Precise  Does the entire tuning process for each background sweep.

Precise Only  Does NOT perform broadband tuning on each sweep. Use this setting when the embedded LO is stable. The signal (after broadband) must be within ½ the tuning IFBW. If the signal will always be within ½ the IFBW, broadband tuning is not needed. Most satellite components are within 3 kHz absolute so might not need broadband tuning.

Disable Tuning  Only the previously measured LO Frequency Delta is applied to the reference mixer LO and VNA receivers.

Tune every - Set the interval at which tuning is performed before a measurement
sweep. 'Tune every 3 sweeps' means that every third measurement sweep is preceded by tuning sweeps. If the embedded LO drifts, or if regularly changing DUTs, use 'Tune every 1 sweep'.

**Broadband Search** - Set the frequency span over which to measure the embedded LO frequency.

**Noise BW** - Noise Bandwidth used for Broadband and Precise tuning sweeps. This sets the resolution in the Broadband sweeps.

**Max Iterations** - The maximum number of Precise sweeps to make. When this number is reached, the final measurement is used.

**Tolerance** - When two consecutive Precise measurements are made within this value, the final measurement is used. If this is not achieved within the Max Iterations value, then the last measurement is used. This is the best of the 'Tunings settings' to change to improve accuracy.

**LO Frequency Delta** - The absolute difference between the measured embedded LO frequency and the LO setting that is entered in the Mixer Tab dialog.

**Find Now** - The VNA finds and measures the actual LO frequency using the current dialog settings. This data is displayed in the Status box.

**Default** - Resets the LO Frequency Delta and Tuning parameters to their default settings.

**Advanced >> button** - Accesses the IF, Processing, ADC & LO, and Data dialogs.
IF Gain

**Auto** - Selects the appropriate amount for gain versus RF frequency bands for each receiver IF Path.

Or select a specific amount of gain (in dB) for IF receiver paths.

**Couple all IF paths** - When checked, all receivers assume the same setting. When cleared, each receiver can assume an individual setting.

**IFConfig** - Accesses the IF Path Configuration dialog. Learn all about IF Path Configuration.

IF Bandwidth

**ADC Filter** - Selects between a narrow and wide IF filter anti-aliasing path.

- **Narrow 11MHz** - Selects the ADC 11 MHz IF filter path. A warning message will appear if the Narrow IF filter path is selected and the Resolution Bandwidth is > 1 MHz. See SA Warning Messages.

- **Wide 38MHz** - Selects the ADC 38 MHz IF filter path.

**Auto** - Check to automatically set the ADC Filter setting based on the ADC Sampling Frequency.

**DFT Bandwidth Auto** - Enables the default values for DFT bandwidth.
With **Auto** checked, the default values are:

**Narrow** - 1 MHz to 10 MHz

**Wide** - 1 MHz to 34 MHz

With **Auto** unchecked, the values can be entered manually. The ranges are:

**Narrow** - 500 kHz to 11 MHz

**Wide** - 500 kHz to 44 MHz

---

**DFT Type** - Sets the DFT record size type. The types include:

- **Power of 2** - Sets the DFT record size to the next power of 2 greater than or equal to the current ADC record size. This is the fastest DFT processing available; the power of 2 record size allows for very efficient computation shortcuts (also known as the FFT algorithm).

- **Fastest** - Sets the DFT record size as close as possible to the ADC record size (larger or equal) while optimizing processing speed.

- **Optimized Radix** - Sets the DFT record size to the minimum integer number larger or equal than the ADC record size that can be decomposed with 2,3,5,7,11,13
radixes (also known as the 13-smooth numbers). The Intel CPUs have shared-coded trigonometric values for 2, 3, 5, 7, 11, 13 fractional angles; the DFT code takes benefits from that for efficient DFT processing.

**Arbitrary** - Sets DFT record size equal to the ADC record size. If the current ADC record size is a large prime number, then the DFT can be very slow. Sometimes, the record size will be increased more than the minimum number required to match the 13-smooth condition, if the whole processing of the sweep is faster with a record size that has a faster DFT time. There is a trade-off here: increasing the record size to speed up the DFT will increase the amount of data to process for the further steps of the SA processing (image rejection, detection).

**Additional comments:**

There is a given ADC record size that gives a given RBW (for a given window type), the RBW evolves as 1/ADC record size. If the DFT record size must be greater, depending on the DFT mode, some zeros will be added to the ADC record size. This is the difference that can be noted between the ADC record size and the DFT record size on the Advanced Processing dialog.

When running the coherent mode of SA, the Arbitrary mode will always be selected, to make sure the DFT bins frequencies exactly land on the coherent signal tones.

The Power of 2, Optimized Radix, and Fastest mode have the same behavior regarding the RBW setting: These 3 modes are increasing the ADC record size to the next best match. This is more sensitive with Power of 2 mode, as the density of available ADC record size is sparse.

The Power of 2 mode makes the SA computation behave exactly like some tools based on FFT processing; this is a use case of this mode. The other use case is to get the fastest processing time that can be useful for intensive spurious search measurements. The penalty is the small number of RBW values available.

The Optimized Radix mode is currently the default mode, this is the one that gives the most accurate RBW setting. The Fastest is to be tried if speed matters for a given non-coherent SA setting.

For example: running 100 kHz RBW with Gaussian filter, the ADC record size must be 1988 samples. Here are the DFT lengths for the different algorithms:

- Power of 2: 2048
- Fastest: 2048
- Optimized Radix: 2000
- Arbitrary: 1988

**End of Sweep Processing**

The End of Sweep Processing function is used to keep the memory buffer of the last full sweep in memory for further processing. This is not done by default because keeping the whole data in memory requires large
amounts of memory and processing (for example, in the case of wide span or low RBW).

Before the implementation of this function, raw data could be sent to a file (ascii or binary) or the fifo, as this can be done while sweeping with no need to keep the whole data in memory. This option is still available.

With each sweep, the data buffers are filled, and are erased if a new speed is started. In other words, this feature works well in the logic of “Single” sweep. You must ensure that a full sweep is in the buffers before pulling out raw data.

**Receivers** - The receiver list can be either ALL, or a specific valid receiver. Only the receivers currently defined for measurement traces can be kept in memory.

**Keep whole sweep data** - Check to keep last full sweep data in memory.

**Info**

**Acq. Time for 1 LO** - Displays the LO acquisition time which is the ADC Record Size x ADC Sampling Frequency (10 nsec or 40 nsec) x (1 + Stacking) x (Video Averaging Coefficient). When settings affecting this value are changed, the displayed value is not updated automatically and will become grayed out. To update the value, close then open this dialog. The analyzer must be sweeping to update values.

**Span Acq. Time** - Displays the total acquisition time to perform a SA sweep. For simple cases, it is the acquisition time of one LO multiplied by the number of LOs. When running Multiple recording coherent pulse mode, the acquisition time here takes into account the duty cycle of the pulses.

**Span LOs count** - Displays the number of LO acquisitions determined by the Image Reject selection and the span. When settings affecting this value are changed, the displayed value is not updated automatically and will become grayed out. To update the value, close then open this dialog. The analyzer must be sweeping to update values.

**Span bins count** - Displays the current span DFT bin count, the number of DFT points processed across the total RF span. When the Detector is bypassed, this is the number of points that are sent to the display.

**DFT resolution** - Displays the DFT resolution.

**DFT record size** - Displays the current DFT record size.

**ADC record size** - Displays the ADC record size value.

**ADC with average** - Displays the ADC acquisition time of one LO multiplied by the averaging (vector averaging or video averaging) factor. It is the straight ADC acquisition time that has to go into the ADC memory for further processing.

**ADC frequency** - Displays the ADC frequency.
**Coherence ratio** - Displays the coherence ratio value.

**Display image reject traces** - Check to display the data acquired by each LO. The minimum number of meaningful traces is determined by the "Image Reject" setting (described in the Advanced dialog above).

**About Image Reject Traces**

These traces display the spectral content of the measured signal for each LO frequency used in the acquisition. The number of ImageReject traces you want to look at is tied to the 'Image Reject' setting. For example, 'Normal' setting is at least 4 ImageReject traces, and 5 more generally.

This function is intended to be used as a diagnostic tool if something looks suspicious.

**Note**: Mixer calibration and user calibration are not applied to the image rejection traces, thus the amplitude readout value is not accurate.

**About Acquisition Time and Sweep Time**

The Acquisition time for 1 LO is really the duration of the ADC acquisition run for one LO setting. Depending on the SA span and the image rejection mode, the number of LOs required for a given SA frequency span changes. This number is reported as Span LOs count. So the total ADC acquisition time for a given span, aka Sweep Acq. Time, is the product of Acquisition time for 1 LO x Span LO count (unless Multiple Recording mode is running, then the Multiple Recording multiplication factor is to be considered).

The entire sweep time is a significantly larger number than the Sweep Acq. Time. Each time the LO is moved, there is a settling time required for LO stabilization. Then each time some raw ADC data is acquired, there is some time to move the data across the buses (the data amount is multiplied by the number of ADCs at work, when several RF receivers are acquired). All the further processing (windowing, DFT, IF calibration, Image Rejection, User calibration) is CPU time consuming, the lowest the RBW the more time it takes to process the DFT samples (there are more DFT samples).

The Image Rejection mode MIN instead of Normal (when not running coherent mode) is a popular way to speed up the sweep time, as roughly half of the LOs count is required, at the cost of higher likelihood of false spectrum spurious (but real signal spurious are still always detected).

The video averaging is digital, it runs several times the ADC acquisitions and windowing and DFT and then averages. The number of times this extra processing is done for video averaging is indicated at the Video averaging count at the SA main page.

When available, the vector averaging is done at the FPGA side, it multiplies by the averaging factor the ADC acquisition time, and it doubles the data amount across the data buses (32 bits instead of 16 bits ADC data representation). The following parts of the processing are not impacted, the quantity of data to process at host CPU side is not increased. Thus the vector averaging is a very efficient way to get lower noise floor with minimum additional CPU time.
The display processing is a very significant part of the sweep time. The number of display points can be set (defaults to 1001), then the code often has to do a data reduction to go from millions of DFT points to 1001 display points. This is the so-called detector processing. This process is CPU intensive, especially the Peak detector that recomputes an X-axis frequency grid each sweep to align each x-axis point to the local interpolated peak. FastPeak is less frequency accurate for the peak points positions but very significantly faster. If the span is narrow, bypassing the detector can speed up the sweep. Turning off the display is another option, it bypasses most detection and display algorithms, thus speeds up the sweep in remote programming modes.

The status bar of the channel shows the last measured SA sweep time, and if some settings are changed, the first sweep after a change shows an estimate with a ~ preceding the timing.

ADC & LO Dialog tab help

**ADC Sampling**

**ADC Sample Frequency** - Select between 100 MHz and 25 MHz.

**Auto** - Check to automatically set the ADC Sampling Frequency.

**Enable FIR for 25 MHz** - Enables the FIR filter for 25 MHz decimation: reduces the noise floor.

**Dithering** - Check to allow ADC dithering to average out the characteristic "stair steps" produced
during the ADC conversion process.

**Overrange warning percent** - Sets the percentage of the ADC input full scale. SA sweeps require thousands (or millions) of ADC samples processed by the FFT. The maximum ADC sample value is kept for the whole sweep, as it is an image of the peak voltage in the IF chain of the instrument. Instead of calibrating this value to Volts, a percentage value of the ADC input full scale is used. If the value is too high, reduce the IF gain or add RF attenuation to maintain the linearity of the instrument. If the value is too low, the signal amplitude is too low and under-utilizing the ADC range. Increasing the IF gain may increase the measurement quality (reducing the noise floor).

**Show ADC Ranges...** - Displays the ADC ranges of the current receivers from the last sweep if the Trigger mode is currently on Hold.

![ADC Ranges](image)

**ADC Raw data**

**Force ADC record size** - Sets the ADC record size which is dependent on the Resolution Bandwidth and ADC Sampling Frequency:

\[
ADC \text{ Record Size} = \frac{1}{ResBW} \times ADC \text{ Sampling Frequency} \times Window \text{ Expansion Factor}
\]

**Check box** - Check to enable the ADC record size to be specified manually. Doing so sets the resolution bandwidth. The size range is 64 Samples to 32 or 64 MegaSamples depending on the selected receivers. The DFT size will be recomputed accordingly to the DFT Type setting. When not checked, the value displayed is the current ADC record size. This feature is not compatible with Coherent Multitone mode.

**Stacking** - Stack ADC samples by the specified number (=1) and store result in memory. For example, if an ADC record size of 1,000 samples is acquired and Stacking is set to 1, then 1,000 samples will be added to 1,000 samples and the result (1,000 samples) will be stored in memory. In other words, we acquire 2,000 samples form ADCs and send 1,000 stacked samples to the next processing stage. Stacking helps to reduce noise and increase dynamic range. However, this feature should only be used when the stimulus frequencies are known and coherent with the current ADC record size. A value
of 0 means no stacking.

**Check box** - Check to enable the ADC sample stacking to be specified manually.

**Note:** Stacking and Video Bandwidth averaging cannot be set together; Video Bandwidth has precedence. The ADC Record Size x (Stacking + 1) must be =64 Mega Samples.

**Note:** Stacking is a great averaging method when Coherent Multitone mode is enabled. We recommend increasing the stacking in Coherent Multitone mode instead of reducing the RBW in order to reduce the noise floor.

**Multiple Recording** - Check to allow the ADC Record Size to be divided and acquired in smaller "chunks" and also specify a wait period between these acquisitions.

- **Chunk Size** - Sets the size to acquire the ADC record in smaller "chunks". For example, if the ADC Record Size is 2048 and the **Chunk Size** is set to 256, then the ADC record is acquired in 8 chunks (1 = Chunk Size = ADC Record Size).

- **Chunk Period** - Set the period to wait between ADC record chunks.

**Note:** This feature is compatible with Coherent Multitone mode.

**LO**

**Randomized LO** - Check to allow dithering of the LO values used when taking a sweep. Allowing randomized LO makes it less likely that erroneous signals will appear in the final measurement.

**Enable baseband X-axis mode (LO independent sweep)** - Enables baseband sweep independent of the LO sweep to allow signals down to 1 Hz to be analyzed. This function is called baseband because it operates without LO conversion and also does not use the mixer path in the IF. The frequency range that can be measured is 1 Hz to 38 MHz due to the IF filter. Therefore, spectrum analysis can be done within this range.

**The following is a procedure for using this function:**

1. Check the **Enable baseband X-axis mode (LO independent sweep)** check box.

2. Connect the signal through one of the IF Path Inputs on the rear panel. Learn more .

3. Go to the **SA** tab and change the **Stop** frequency to 38 MHz. Notice the **Start** frequency changes to 10 MHz. Change the **Start** frequency if necessary.
4. Go to the IF tab, click on the IF Config... button, then under IF Path for the IF inputs you want to consider, select External.

![IF Path Configuration](image)

**Note:** The SA baseband mode only allows you to run one R receiver. Any R receiver selection (R1 to R4) will get the signal from the R IF input.

5. In the IF tab under IF Bandwidth, deselect ADC Filter Auto and select Wide 38MHz. This allows you to change the Wide frequency range if necessary. If the SA current span is larger than the IF bandwidth, -200 dBm will be displayed outside of the IF filter area.

![IF Bandwidth Configuration](image)

**Force LO to Frequency** - Sets the LO to a specified frequency. This check box can only be set if Image Reject is set to None, LO Low or None, LO High or Enable baseband X-axis mode (LO independent sweep) is checked.

**Check box** - Check to enable the LO frequency to be specified manually.

This function allows the LO source to be used mostly like another source that can be set up relative to a source port. For example, if Port 1 in the Source tab is set to a frequency of 1 GHz and the **Force LO to frequency** is set up as shown below, then the LO will track 1 MHz after the Port 1 source frequency.
Data Dialog tab help

Data Format -

- **Float LogMag (dB)** - Sets the data format to log magnitude in dBm.
- **Float LinMag** - Sets the data format to linear magnitude in volts.
- **Integers** - Sets the data format to Packed Integers (each value is a short 16 bit integer, the equation to compute the dBm value is: dBm = Xshort/200.0 - 36.165.

**Export receivers** - Select the data to export from a specific receiver or all receivers.

**Don't save data below threshold** - Set data level threshold mode and threshold level in dBm. For text file output with verbose mode, only the frequencies with power greater than this threshold setting will be written to the file.

**DFT bins count** - Displays the current DFT bin count, the number of DFT points processed across the total RF span. When the Detector is bypassed, this is the number of points that
are sent to the display.

**Receivers count** - Displays how many receivers are currently being exported. The number here can be less than the number of receivers specified in Export Receivers, if some of them are not selected in the channel.

**Export to binary file** - Set data to be exported to a binary file. Data is not exported until the next new sweep occurs.

**Export to text file** - Exports data only. Data is not exported until the next new sweep occurs.

**Verbose mode** - Exports frequency and data. Data is not exported until the next new sweep occurs.

**Erase files each new sweep** - Selecting this option will erase the data after each sweep. If this option is not checked, the data from each sweep will continue to be appended to the output data file which can create a very large file size (and fill the disk, with many unwanted consequences).

**File name prefix** - The receiver selected in Export receivers will be appended to the prefix name specified in this field with either "*.txt" if a text file is exported or "*.bin" if a binary file is exported. For example, if C:\Temp\SA_DATA_OUT is entered into the File name prefix field and the "B" receiver data is exported to a text file, the data will be exported to a file called SA_DATA_OUT_B.txt.

**Record size (bytes)** - This is the byte size of binary data output.

**Export markers with data files** - Adds marker data and data to the text file (*.txt) output.

**Export all markers to a single file** - Adds all marker data to a single text file (*.txt) output.

**Export to FIFO buffer** - Exports data to the FIFO (First-IN, First-OUT) data buffer. FIFO is a circular buffer that allows very fast Read-Write access.

**Export to shared memory** - Exports data to shared memory (Microsoft Windows feature) which is the fastest way to transfer data between applications. The application that is retrieving data has to register itself to Microsoft Windows with the same share name.

**Share name** - Assigns a specified name to the shared data.
In the SA Setup Source Tab (above):

> When (Sweep) Type = **CW**, set the CW Frequency.

![Frequency Settings](frequency_cw.png)

> When (Sweep) Type = **Linear**, the following dialog appears:

![Frequency Settings](frequency_linear.png)

**Settings**

Sets the source frequency range. Use either of the following pairs of settings to set the frequency range.

- **Start /Stop** - Specifies the beginning and end frequency of the swept range.
- **Center /Span** - Specifies the value at the center and frequency range.

**Source Number of Steps** - Sets the number of steps the source will make across the specified source frequency range.

**SA Sweeps / Source Step** - Sets the number of SA (receiver) sweeps for each Source Step. This setting is common to all sources.
In the SA Setup Source Tab (above):

> When (Sweep) Type = CW, or LinFreq, the following dialog appears:

![Power Settings dialog](Image)

> When (Sweep) Type = Power or LinF+Pwr, the following dialog appears:

![Power Settings dialog](Image)

**Settings**

Sets the source power range. Use either of the following pairs of settings to set the power range.

**Start /Stop** - Specifies the beginning and end power of the swept range.

**Center /Span** - Specifies the value at the center and power range.

**Source Number of Steps** - Sets the number of steps the source will make across the specified source power range.

**SA Sweeps / Source Step** - Sets the number of SA (receiver) sweeps for each Source Step. This setting is common to all sources.
In the SA Setup Source Tab Phase column (above):

> When (Sweep) Type = \textbf{CW} or \textbf{LinFreq}, \textbf{Power}, or \textbf{LinFreq+Pwr}, the following dialog appears:

![Phase Settings dialog](image)

> When (Sweep) Type = \textbf{Phase}, \textbf{Freq+Pha}, \textbf{Pow+Pha}, or \textbf{Fr+Pw+Ph} the following dialog appears:

![Phase Settings dialog](image)

**Settings**

Sets the source phase range. Use either of the following pairs of settings to set the phase range.

- **Start /Stop** - Specifies the beginning and end phase of the swept range.

- **Center /Span** - Specifies the value at the center and phase range.

**Source Number of Steps** - Sets the number of steps the source will make across the specified source phase range.

**SA Sweeps / Source Step** - Sets the number of SA (receiver) sweeps for each Source Step. This
Setting is common to all sources.

**Measurement Parameters**

**How to select and configure Measurement Parameters**

**Using Hardkey / SoftTab / Softkey**

1. Select a trace by pressing `Trace > Trace N > Trace N`.
2. Press `Trace > Trace Setup > Measure...`.
3. Select a parameter.

**Using a mouse**

1. Right-click on a trace.
2. Select a parameter.

**Measure Dialog**

Select one of the test port receivers to make a measurement. Test ports are identified in both traditional VNA notation and Receiver Notation. Learn more.

Or select an ImageReject measurement, if the Display Image Reject Traces checkbox has been set.

**New Trace** - Add a new trace.

**Channel N** - Select a channel number for the new SA trace.
**Window N** - Select to create the new trace in an existing window or new window.

**Select All** - Select all measurement parameters.

**Note:** Configuring an external DC meter for a SA channel is supported. DC is read at the end of a sweep. Only one point is read and the entire trace is filled with this point reading regardless of the number of channel points.

---

**Marker -> SA**

The following marker-related features are unique to SA.

**Marker -> SA**

This feature is supported in Standard, SMC or Swept IMD measurement classes (channels) ONLY. In this section, these are called NA channels.

On a standard channel with a marker residing on a trace in an NA channel, **Marker -> SA** creates a new SA channel in full frequency span and provides access to the SA Analysis Markers.

Refer to the following for swept IMD and SMC:

Swept IMD: span = 10 * delta frequency

SMC: SA channel uses receiver frequency range

A marker is created on the trace at the same frequency as the NA channel marker. This is a quick way to see the frequency spectrum of the NA channel at a specific frequency.

- The same source that is used for the trace in the NA channel is turned ON in the SA channel in CW mode at the marker frequency.

- The same receiver that is used for the NA channel is used for the SA channel.

- For each new NA channel, a new SA channel is created. Subsequent markers in the same NA channel use the same SA channel.

- In general, **Marker -> SA** creates a new measurement on the SA channel only if the measurement does not already exist. For example, if a marker is used on an S11 measurement in a standard channel, **Marker -> SA** creates a measurement using test receiver "A" with port 1 as the source. If **Marker -> SA** on an S12 measurement is then performed, the same test receiver "A" is used except that port 2 becomes the source. In this case, a new SA trace will not be created.
How to use Marker \text{ -> SA}

With a marker residing on a trace in a standard channel...

- With a mouse: Right-click on a marker, then select Functions, then Marker \text{ -> SA}.
- With a keyboard: With the relevant marker active (selected), Alt+M, F, A.
- Without mouse or keyboard: With the relevant marker active (selected), Press Marker, Marker \text{ -> Functions}, then Marker \text{ -> SA}.

\textbf{Band Markers}

Once an SA channel has been created using Marker \text{ -> SA}, the Band Markers are accessed by selecting the SA Analysis softtab. The following marker types provide a readout of the total power, noise, or density within a selectable frequency span. The span is marked by vertical posts that appear on either side of the marker.

- Band Power - These markers provide a readout of the total power within a specified frequency span defined by Band Span.
  
The Average detector type makes the display more consistent with Band Power markers. With this detector type the marker readouts and the estimated power values or delta from the display have a good match. Many legacy spectrum analyzers need to set the detector type to average to do band power measurements. PNA-SA does not need to select the average detector to compute the right band power measurements, it's only useful to check the values between marker readouts and display estimates.

- Band Density - Select Off, Noise (dBm/Hz), Power (dBm/Hz), Tone (dBm/Tone), NPR, or ACPR.

Refer to SA Analysis Markers for a full description of the marker types and their settings.

\textbf{Band Power and Band Density Noise Markers}

\textbf{Note:} If a Band Power or Band Density Noise marker is selected, Discrete mode is turned OFF to allow precise measurements over the desired frequency range.

The span is marked by vertical dotted lines that appear on either side of the marker. The marker's y-axis value is set to the measured power value.
If a Band Power or Band Density Noise marker is in Delta mode, the difference between the Band Power or Band Noise marker and the reference marker is displayed with a leading delta symbol.

**Band Density Power Markers**

The ? 1 marker displays the notch frequency relative to the center frequency. In this example the notch is in the center so the frequency is 0 Hz. Also displayed is the notch span (100 MHz).
The Tones is the difference between the average tone power a the Reference (> R ) marker and the average tone power at the bottom of the notch.

The > R marker displays the center frequency (16 GHz) and the signal span (1.5 GHz).

The Band Power displays the total power within the signal span.

The Tones (1.40 GHz) displays the average tone power across the 1.5 GHz signal span minus the average tone power across the notch (1.5 GHz - 100 MHz = 1.4 GHz)

The total modulation span defined by Band Span is marked by the outer vertical dotted lines.

Each notch is also marked with vertical dotted lines defined by Density BW.
How to select Band Markers

1. Press Marker > SA Analysis.

2. Another method of selecting band markers is to right-click on a marker on the display then select Search then Search... from the menu. The following dialog appears:

![Marker Search dialog](image)

3. For information on the settings for different SA search types, refer to SA Analysis Markers.

If a Band Power or Band Noise measurement cannot be made, the marker readout will display -999.
dBm (for Band Power), or -999 dBm/Hz (for Band Noise). There are two reasons why this may happen:

1. *The band span (from marker frequency – span/2 to marker frequency + span/2) is outside the frequency range of the channel.*

2. *The Band Power or Noise marker was created while the channel was in Hold mode. At least one sweep must be taken after creating such a marker. The marker can be moved taking a sweep while in Hold mode. However, the marker readout will not change. To update the marker readout to the new marker location, a re-sweep is required.*

**Occupied BW Ratio**

The Occupied BW Ratio is the frequency range that contains a defined percentage of the overall band power as specified by OBW Percent. The marker readout provides the occupied band center frequency, percentage of the band span to measure, and the occupied band power. See also Occupied BW search min for setting the minimum frequency to start a search.

The span is marked by vertical dotted lines that appear on either side of the marker indicating the percentage of span. The marker's y-axis value is set to the measured power value.
How to select Occupied BW Ratio

1. Select one of three ways to enable Occupied BW Ratio:
   
   a. Move the cursor on a marker, right-click on the marker, select Search, then Occupied BW.
   
   b. Press Marker, then SA Analysis, then Occupied BW ON.
   
   c. Select Response, then Search...
      The following dialog appears

   ![Marker Search Dialog]

2. If the Marker Search dialog is used, perform the following steps:
   
   a. Select an existing marker.
b. For Search Type, select Occupied BW Ratio.

c. For Percent, enter a percentage of the band span to search.

d. For Search Domain, either select Full Span (default) or define a User Span by selecting User N then specifying the Start and Stop frequencies.

e. Click OK.

---

**Calibrating an SA Channel**

A calibration can be performed on the SA Channel using the Cal All Wizard when corrected results are required to improve amplitude accuracy.

**Note:** Calibration is performed over the currently specified spectrum analyzer frequency range only.

Another method of calibration is to import an existing Cal Set. An imported Cal Set must contain the Receiver Response terms for the measurement port on the SA channel. In addition, if the imported Cal Set covers a narrower frequency range than the SA channel, the error terms in the imported Cal Set are extrapolated.

The Cal Plane Manager can be used to characterize adapters, cables, and fixtures used to connect a DUT to the VNA to remove their effects from the measurement. See the detailed procedure.

---

**Gated SA**

Gated spectrum analyzer measurements are triggered measurements that capture data for the duration of the active level of the trigger. Thus, only level external triggers can be used (High level, Low level) for gated measurements. Edge triggers can be used to start SA measurements, but will not gate them. It is not recommended to use level triggers to start measurements: if the active level is shorter than the required acquisition time (that depends on the RBW and other settings), then the measurement will abort.

A given SA sweep needs a given number of LOs (this depends mainly on the span and on the Image Reject setting). This number can be retrieved at the SA Processing dialog. Each LO needs one ADC acquisition and the duration of one ADC acquisition can be retrieved at the SA Processing dialog. The time for each LO acquisition is based on the ADC Record Size times the ADC Sampling Frequency (defaults to 10 nsec). For gated SA, it is meaningful to gate each individual ADC acquisition. This can be done with the Trigger Setup dialog, with the external trigger configured per channel (Trigger Scope) in point mode (channel trigger state mode), and Meas Trigger set to Level.
For example, if the ADC Record Size is 663, the time required for a full DFT acquisition would be 663 x 10 nsec = 6.63 usec. As long as the width of the measurement trigger is greater than 6.63 usec, measurement data will be acquired. If the measurement trigger width is set to less than 6.63 usec, then no measurement data will be acquired. If the width of the measurement trigger is much larger, for example 20 usec, then up to 3 ADC acquisitions with 663 samples can occur during the active status of the trigger.

**Note:** Internal Pulse0 pulse generator is not suitable for gated SA because the pulse width cannot be set. Instead, use Pulse3 as shown in the Gated Measurement example (pulse 1 to pulse 4 would work too, but without the direct Trigger source to Pulse3 internal connection).

Refer to Gated Measurement for an example of how to set up a gated spectrum analyzer measurement.

**SA Warning Messages**

Warning messages appear when the measurement cannot be performed with the current settings. Messages are displayed in blue for three seconds and the channel is placed in hold (not sweeping).

SA Warning: Image Reject Max and Better are not allowed if RBW > 1MHz and Narrow IF Filter.

SA Warning: Image Reject Max is not allowed if RBW > 1MHz.

SA Warning: Image Reject Better is not allowed if RBW > 2MHz.

SA Warning: Image Reject Max and Better are not allowed below 20 MHz if RBW > 1MHz and Wide IF Filter.

SA Warning: Cannot run Image Reject None LO High at high end of RF frequencies.

SA Warning: Cannot run Image Reject None LO Low at low end of RF frequencies.

SA Warning: Cannot force Narrow IF Filter if RBW > 1MHz.
Noise Power Ratio (NPR) Settings

NPR measures the nonlinear behavior of an RF microwave amplifier under a modulated signal stimulus.

In this topic:

- Requirements
- Create a Spectrum Analysis Channel
- Accessing NPR Settings
- Modulation settings dialog
- Create Modulation dialog (separate topic)
- Modulation Cal - Setup dialog (separate topic)

See Noise Power Ratio (NPR) Measurement Example

Requirements

- Spectrum Analyzer Option S9x090xA/B
- 2- or 4-Port PNA with ability to connect an external modulated source
- Supported external sources:
  - E8267D PSG Vector Signal Generator
  - M8190A with E8267D PSG Vector Signal Generator
  - M9383A/B, M9384B (wideband vector, 44 GHz, with enhanced phase noise)

Note: When setting up an M9383A, M9383B, or M9384B source in the External Device Configuration dialog, select **MXG_Vector** as the driver (they are code compatible).

- N5182B MXG RF Vector Signal Generator
- N5192A and N5194A UXG Vector Adapter
Create a Spectrum Analysis Channel

1. On the VNA front panel, press Meas > S-Param > Meas Class....

2. Select Spectrum Analysis, then either:
   - OK delete the existing measurement, or
   - New Channel to create the measurement in a new channel.

3. A Spectrum Analysis measurement is displayed.

Accessing NPR Settings

Using Hardkey/SoftTab/Softkey

1. Press Freq > Source IQ Mod. > MXG.

Using a mouse

1. Click Stimulus
2. Select SA Setup...
3. Select the Source tab.
4. Click in the IQMod column then select Edit.

Modulation settings dialog help
**Modulation source** - Displays the name of the modulation source.

**Enable Modulation** - Check to enable modulation.

**Create/Edit Mod File...** - See Create Modulation dialog description.

**Show File Properties** - Displays the properties found in the modulation file. This is the active Modulation Filename selected in the Modulation settings dialog. This information cannot be edited.
**Modulation Filename** - Selects the modulation file.

**Autoset Frequencies and Coherence for current modulation.**

**Autoset NPR Markers** Enables and disables NPR markers.

**Autoset ACPR Markers** Enables and disables ACPR markers.

**NPR Band Guard** - Guard band on each side of the notch in an NPR measurement.

**ACPR Band Guard** - Guard band between the end of the modulation signal and where ACPR is measured.

**Immediate Autoset** button - Click to adjust frequencies and markers if modulation settings are changed.

**Enable Modulation Correction** - Automatically checked if user exits the calibration wizard with **Exit & Save Cal**. This option is grayed out if a modulation calibration has not been performed.

**Source Amplification (= Power Offset) = 0.000 dB** - Displays the Power Offset to monitor the power budget between the source and DUT. The Power Offset setting provides a method of compensating port power for added attenuation or amplification in the source path. [Learn more.](#)
Calibrate Modulation... - See Modulation Cal - Setup dialog description.
The Intermodulation Distortion (IMD) App is offered for both Amplifiers and Converters, and also can be viewed in both Swept IMD and IM Spectrum channels.

- **Swept IMD** channels are used to measure over 150 different IMD parameters. These parameters are built from various selections such as: IMD products, tone power levels, intercept points, at the input and/or output of a DUT. In addition, each parameter can be swept using several different methods.

- **IM Spectrum** measurements provide a traditional spectrum analyzer view of the main tones and distortion products.

Learn all about the IMD App (Opt S93087A/B) in the following topics:

**Concepts**

- Swept IMD and IM Spectrum Concepts

**How to Use ALL IMD Apps**

- Swept IMD for Amplifiers
- IM Spectrum for Amplifiers
- Swept IMDx for Converters
- IMx Spectrum for Converters

**Requirements for Opt S93087A/B IMD App**

PNA-X or PNA (N522x) with:

- FOM (Opt S93080A)
- Opt S93087A/B (software option only) must be enabled.
- Configurable test set

**Note:** N522x models require additional configuration and equipment. See Configure External Source and Combiner for IMD Apps.

**See Also**
• Programming commands

• App Note 1408-17 - Making Accurate Intermodulation Distortion Measurements with the PNA-X.

• PNA-X Models and options

• N522x Models and options
Swept IMD and IM Spectrum Concepts

- Swept IMD Concepts
- Swept IMD for Converters (separate topic)
- Swept IMD Parameters
- How the VNA Measures IMD
- How an IM Spectrum Channel Works
- IM Spectrum Parameters

Other IMD topics

Swept IMD (Intermodulation Distortion) Concepts

When a device or system is subjected to multiple input frequencies, the non-linearity of the DUT can generate undesired outputs at other frequencies. Typically, two input tones of equal power separated in frequency by a specified amount are used to stimulate the device while observing the resulting frequency spectra at the output. A variety of measurements can then be utilized to determine the intermodulation distortion characteristics of the device.

The frequencies of the resulting distortion products are predictable. While many mixing products can be generated, the high and low signals of the "odd order" products (3rd, 5th, and so forth) are close enough to the original two signals to potentially interfere with adjacent communication channels. With the exception of the 2nd order product, the higher "even order" products are usually far enough away to be of no interest.

The following image and table shows two equal-power main-tones and the nearby odd-order distortion products, as well as the 2nd order product (not shown in the image). Notice that the frequency separation between adjacent odd-order products is the same as the separation of the main tones (Delta F) frequency. For most devices, these distortion products become worse as the device is pushed further into compression.
Two main tones (f1 and f2) with odd order intermodulation products.

The following table shows the calculations and example frequencies (Blue text) of the intermodulation products that are closest to the two main tones.

<table>
<thead>
<tr>
<th>Product</th>
<th>Low (L)</th>
<th>High (H)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main tones</td>
<td>f1</td>
<td>f2</td>
</tr>
<tr>
<td></td>
<td>100 MHz</td>
<td>120 MHz</td>
</tr>
<tr>
<td>3</td>
<td>2f1-f2</td>
<td>2f2-f1</td>
</tr>
<tr>
<td></td>
<td>80 MHz</td>
<td>140 MHz</td>
</tr>
<tr>
<td>5</td>
<td>3f1-2f2</td>
<td>3f2-2f1</td>
</tr>
<tr>
<td></td>
<td>60 MHz</td>
<td>160 MHz</td>
</tr>
<tr>
<td>7</td>
<td>4f1-3f2</td>
<td>4f2-3f1</td>
</tr>
<tr>
<td></td>
<td>40 MHz</td>
<td>180 MHz</td>
</tr>
<tr>
<td>9</td>
<td>5f1-4f2</td>
<td>5f2-4f1</td>
</tr>
<tr>
<td></td>
<td>20 MHz</td>
<td>200 MHz</td>
</tr>
<tr>
<td>2</td>
<td>f2-f1</td>
<td>f1+f2</td>
</tr>
<tr>
<td></td>
<td>20 MHz</td>
<td>220 MHz</td>
</tr>
</tbody>
</table>

Learn about Swept IMDx for Converters Concepts

Swept IMD Parameters

The following basic parameters, offered for both Amps and Converters, are expanded to over 150 by selecting specific product tones (2,3,5,7,9), the Low-side, High-side, or Average of these tones, measured at the Input or Output of the DUT.
• Tone Power
• Tone Gain
• Intermodulation Distortion
• Intercept Point Parameters
• Composite Triple Beat (CTB)
• Composite Second-order Beat (CSO)
• Cross-Modulation Distortion

Learn how to select these IMD parameters.

**Tone Power Parameters**

Tone Power parameters measure the **absolute** power level of the main tones, odd-order product tones up to the 9th order, and the 2nd order product tones. These tone powers can be measured at the input and output of the DUT. Because the tones come in pairs, the **Low tone**, **High tone**, and the **Average** of the two can be measured and displayed.

The Average Tone Power is calculated as follows:

$$\text{Avg} = \frac{(\text{High tone (dBm)} + \text{Low tone (dBm)})}{2}$$

When measuring the 2nd order products, only the Low tone and High tones are allowed. When the main tones are separated by less than 10 MHz, the Low tone (f2-f1) is below the frequency range of the VNA.

**Tone Gain**

Tone Gain (in dB) calculates the main tone Output Tone power / Input Tone power. Because the tones come in pairs, Tone Gain can be calculated for the **Low tone**, **High tone**, and the **Average** of the two as indicated in the Average Tone Power calculation.

For IMDx for Converters, the Input and the Output tones are typically at different frequencies.

**Intermodulation Distortion Parameters**

IMD parameters measure the **difference** in power level between the specified product tone and the main tones. These IMD parameters are calculated from the Tone Power measurements. IMD parameters can measure the odd-order product tones up to the 9th order, and the 2nd product tones, at the DUT Input or Output. For each specified product, the difference between the **Low** product and main tone, difference between the **High** product and main tone, and difference between the **Averages** of the product and main tone can be measured and displayed.
Swept IMD supports IMD parameters which are calculated as follows:

<table>
<thead>
<tr>
<th>IMxLo</th>
<th>PwrxLo-PwrMainLo</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMxHi</td>
<td>PwrxHi-PwrMainHi</td>
</tr>
<tr>
<td>IMx</td>
<td>Pwrx-PwrMain</td>
</tr>
<tr>
<td>IMxLoIn</td>
<td>PwrxLoIn-PwrMainLoIn</td>
</tr>
<tr>
<td>IMxHiIn</td>
<td>PwrxHiIn-PwrMainHiIn</td>
</tr>
<tr>
<td>IMxIn</td>
<td>PwrxIn-PwrMainIn</td>
</tr>
</tbody>
</table>

where:

- \( \times \) = the IM product of interest (2, 3, 5, 7, 9)
- \( \text{Avg} \) is implied if \( \text{Hi} \) or \( \text{Lo} \) is not stated
- \( \text{Output} \) is implied when \( \text{In} \) is not stated

Learn how to select IMD parameters.

**Intercept Point Parameters**

As the main tone output power increases (black arrow), output power in the specified product tone increases at a predictable, and steeper, rate (green arrow). At some point, the power in the product tone will be equal to the power in the main tone. The power level at which this occurs is known as the intercept point. Measuring this point directly is typically not possible. Therefore, it is calculated by measuring the main tone power and the specified product tone power.
The Swept IMD App can display either the DUT Input power or DUT Output power that is required to achieve the theoretical intercept point. This is called either Input Referred (IIP) or Output Referred (OIP).

This measurement can be made for the 2nd, 3rd, 5th, 7th, and 9th order intercept points. In addition, the measurements can be made for either the Low tone, the High tone, or the Average of the two. However, for the 2nd order intercept point, only Low and High tone parameters are supported; not Average.

Swept IMD supports Intercept Point parameters which are calculated as follows:

\[
\begin{align*}
OIPxHi &= PwrMain - IMxHi/(x-1) \\
OIPxHiIn &= PwrMain - IMxHiIn/(x-1) \\
OIPxLo &= PwrMain - IMxLo/(x-1) \\
OIPxLoIn &= PwrMain - IMxLoIn/(x-1) \\
OIPx &= PwrMain - IMx/(x-1) \\
OIPxIn &= PwrMain - IMxIn/(x-1) \\
IIPxHi &= PwrMainIn - IMxHi/(x-1) \\
IIPxHiIn &= PwrMainIn - IMxHiIn/(x-1) \\
IIPxLo &= PwrMainIn - IMxLo/(x-1) \\
IIPxLoIn &= PwrMainIn - IMxLoIn/(x-1) \\
IIPx &= PwrMainIn - IMx/(x-1) \\
IIPxIn &= PwrMainIn - IMxIn/(x-1)
\end{align*}
\]

where:

- \( x \) = the IM product of interest (2, 3, 5, 7, 9)
- \( PwrMain \) = average power of the main tones at the DUT Input or Output
- \( IMx \) = product tone power - main tone power (from above IMD parameter)
- \( Avg \) is implied if \( Hi \) or \( Lo \) is not stated
- \( Output \) is implied if \( In \) is not stated

Learn how to select IMD parameters.

**Composite Triple Beat (CTB)**
From the NCTA Standard, composite triple beat is defined as the modulation beat of the target channel signal caused by triple beat resulting from the nonlinear characteristic of the DUT. Composite triple beat is expressed as the ratio of the target channel signal level to the maximum mean level of beat components dispersed around the carrier of that target channel.

Swept IMD supports two parameters of this type:

- CTB is based upon an approximation for the number of beats in mid-band
- CTBE is based upon an approximation for the number of beats at the band edge.

The equations for these two parameters are as follows:

Mid-Band CTB (dB) = \(-2(P_i - P_s) + 6 + 10 \log(3N^2/8) + \text{CTB Offset}\)

Band Edge CTBE (dB) = \(-2(P_i - P_s) + 6 + 10 \log(N^2/4) + \text{CTB Offset}\)

Where:

- \(P_i\) = Output power level at the third order intercept point (dBm): OIP3 (Lo | Hi)
- \(P_s\) = One of the following values based upon the Composite Normalization Mode:
  - For PDBM or PDBMV mode, \(P_s = \text{CompositeNormalizedCTBPower}\)
  - For Number of Carriers mode, \(P_s = \text{PwrMain (AVG)} - 10 \log(N/2)\)
  - For None mode, \(P_s = \text{PwrMain (AVG)}\)
- CTB Offset = Offset value for CTB calculation
- \(N\) = Total number of carriers.

**Note:** CTB Offset and \(N\) values can ONLY be set using SCPI or COM commands.

Learn how to select IMD parameters.

**Composite Second-Order (CSO)**

From the NCTA Standard, composite second order is defined as the modulation beat of the target channel signal caused by second order beat resulting from the nonlinear characteristic of the DUT. Composite second order beat is expressed as the ratio of the target channel signal level to the maximum mean level of beat components dispersed around 0.75 MHz and 1.25 MHz above and below the carrier of that channel.

Swept IMD supports a CSO parameter which is calculated as follows:
CSO(dB) = (Pi - Ps) + 10Log(N) + CSO Offset

Where:

- **Pi** = Output power level at 2nd order intercept point: OIP2 (Lo | Hi)
- **Ps** = One of the following values based upon the Composite Normalization Mode:
  - For **PdBm** or **PdBmV** mode, **Ps** = CompositeNormalizedCSOPower
  - For **Number of Carriers** mode, **Ps** = PwrMain (AVG) – 10Log(N/2)
  - For **None** mode, **Ps** = PwrMain (AVG)
- **CSO Offset** = Offset value for CSO calculation
- **N** = Number of distortion products.

**Note:** **CSO Offset** and **N** values can ONLY be set using SCPI or COM commands.

Learn how to select IMD parameters.

**Cross-Modulation Distortion**

From the NCTA Standard, cross modulation is defined as the distortion that causes modulated carrier components of undesired channels to amplitude-modulate the target channel carrier due to the nonlinear characteristic of the unit under test. Cross modulation distortion is expressed as the ratio of the target channel carrier level to the level of modulated components of the carrier of the target channel resulting from modulated signals of undesired channels.

Swept IMD supports an XMOD parameter which is calculated as follows:

\[ XMOD = -2(Pi - Ps) + 6dB + 20Log(N) \]

Where:

- **Pi** = Output power level at third order intercept point: OIP3 (Lo | Hi)
- **Ps** = Power level of each carrier: PwrMain (AVG)
- **N** = Total number of carriers.

Make Cross Modulation settings using SCPI or COM commands.

**How the IMD Application Works**
The following diagram illustrates how the VNA is configured to generate the two main tones. This shows a PNA-X with dual sources and the internal combiner. A 2-port or 4-port N522xA model can also be used. Learn how to Configure External Source and Combiner.

2-port PNA-X generates the f1 and f2 main tones.

Depending on the specified parameters and sweep type, the sources and receivers are tuned to the appropriate frequencies in order to measure all of the required main and product tone powers. For example, an IM3 parameter requires the measurement of both main tones, and the 3rd order High and Low tone powers.

The Narrowband IF path is used for IMD measurements to help reduce spurious responses. Because the narrowband filter has a bandwidth of about 28 kHz, using an IFBW greater than 30 kHz does nothing to improve measurement accuracy. Learn how to set IFBW for IMD.

Limiting Stimulus Settings and Out of Range Product Tones

Because the main tones are generated by the VNA internal sources and external sources, the frequencies of the main tones must always be within the frequency range of the VNA or external source. Sweep parameter values are adjusted when necessary to ensure that f1 and f2 frequencies are within these limits.

However, the VNA DOES allow you to make settings that cause the selected IM products to fall outside the frequency range of the VNA. For example, with the main tones at 10 MHz and 15 MHz, the VNA will allow you to select the parameter IM3Lo (3rd low side product tone). However, the frequency of this product will be at 2f1-f2 or 5 MHz, which is below the frequency range of the VNA. In these cases, the trace data is set to zero, which converts to -200 dB in Log Mag format.
**Limited Number of Acquisitions**

The total number of acquisitions per sweep can not exceed 32,001 points. The number of acquisitions is determined by multiplying the number of trace points, by the number of tones frequencies, then by 2 (for both Input and Output frequencies). The VNA will automatically reduce the number of trace points to ensure the total number of acquisition points does not exceed 32,001.

**How an IM Spectrum Channel Works**

Before reading this topic, you should become familiar with [IMD Concepts](#).

The IM Spectrum channel provides a traditional spectrum analyzer view of the intermodulation distortion behavior of a device. Unlike the Swept IMD channel, the main tones (F1 and F2) are fixed while the receiver is swept over a frequency range of interest in order to generate a display as shown below.

![IM Spectrum trace (bottom) with Swept IMD traces (top)](image)

A typical spectrum analyzer does NOT have a signal source. This one does. The signal source, or stimulus, settings for the F1 and F2 main tones can be set in either the IM Spectrum channel or the Swept IMD channel. These settings include the frequencies and power levels of the main tones.
The settings for the IM Spectrum receiver can be set ONLY in the IM Spectrum channel. These settings include how many tone products to view - which determines the center and frequency span - and the Resolution Bandwidth.

You can choose from several Resolution Bandwidths which also determines the number of data points used in the channel. The higher the Res BW, the fewer the number of data points. The formula for determining the number of points is:

\[ N = \frac{\text{Span}}{\text{ResBW}} \times 3 \]

The IM Spectrum channel performs multiple measurements for each data point in order to reject unwanted images which are generated by the VNA internally. This provides a high degree of confidence that signals captured in a trace are real and are not spurious responses generated in the measurement process.

**IM Spectrum Parameters**

You can select from three different IM Spectrum parameters.

1. The tones OUT of the DUT (default parameter).
2. The tones IN to the DUT (to be sure that the input signals are pure). **NOT supported** in IMx Spectrum
3. Reflected tones off the DUT input. **NOT supported** in IMx Spectrum

Learn how to select IM Spectrum parameters for Amplifiers or Converters.

Learn all about IM Spectrum for Amplifiers or Converters.

See list of all IMD topics.
Features and Limitations

Create a Swept IMD Measurement

- Frequency tab
- Power tab
- Configure tab
- How to specify IMD Parameters

IMD Calibration

Saving Swept IMD Power Data

Other IMD (Opt S93087A/B) Topics

Features and Limitations

See requirements.

Features

- Fast and easy setup for the measurement of a variety of distortion related parameters up to 9th order.
- Measurement at both input and output of a DUT.
- Supports a variety of sweep-modes for the main-tones: center-frequency linear and segment sweep, tone-separation sweep, power sweep, or CW sweep (fixed main-tone).
- Make very fast, accurate measurements using the VNA sources with high-power, high linearity, and low harmonics.
- Supports calibration and correction of Swept IMD parameters.
- Independently set IFBW for measuring main-tones versus product tones.

Limitations

- 2-port non-frequency converters ONLY. For frequency converters, use Swept IMDx.
DUT to VNA port mapping is limited. Port selections are made on the Power tab. Learn more.

When using Integrated Pulse application, the IF Filter setting for the relevant receiver must be changed to 'Wide'. Learn how. The default IF Filter setting in Swept IMD is 'Narrow' in order to avoid spurs and harmonics. The IF Attenuator setting (on the same dialog) may also require adjusting.

The following features are NOT available with Swept IMD:

- Number of points limited to 20,001
- Independent IFBW, Power Levels, or Sweep Time in a segment table is NOT supported.
- Analog Sweep (Stepped sweep mode only)
- Log frequency sweeps
- Unratioed receiver measurements (A, B, R)
- ECAl User Characterization
- Time Domain
- Balanced measurements
- Save Formatted Citifile data.
- Save SnP data.
- Port extensions
- Some Fixturing Features
- External Test Set Control (Option S93551A/B)
- Integrated Narrowband or Narrowband Pulse App
- External DC Sources (DC Meters ARE supported).
- See Frequency limitations in a Swept IMD channel.
- Option 205 or Option 425

Create a Swept IMD Measurement

1. Press Meas > S-Param > Meas Class....
2. Select Swept IMD, then either:
1. **OK** delete the existing measurement, or

2. **New Channel** to create the measurement in a new channel.

3. A Swept IMD measurement is displayed. To select additional parameters to display, click **Response**, then **Meas**, then select a parameter from the list.

### How to start the Swept IMD Setup dialog

**Using Hardkey/SoftTab/Softkey**

1. Press **Freq** > **Main** > **IMD Setup**.

**Using a mouse**

1. Click **Stimulus**
2. Select **IMD Setup**.

### Frequency tab - IMD Setup dialog box help

![Sweep Type and Sweep Settings](image)

Configures the Sweep Type and frequency range for SweptIMD and Swept IMDX measurements.

**Sweep Type and Sweep Settings**
Sweep fc (center frequency)

Maintaining a constant tone spacing (Fixed DeltaF) and tone power, fc is swept from Start fc to Stop fc. Center Frequency can also be specified as Center fc and Span fc.

At each fc, the receivers are tuned to all of the required frequencies to measure the power of the appropriate tones.

Swept DeltaF (tone spacing)

The specified fc (center frequency) and tone power is held constant. The tone spacing is increased from Start DeltaF to Stop DeltaF in the specified number of points.

Power Sweep

The main tone frequencies are specified as either f1 and f2, or as fc and DeltaF. These frequencies are held constant while the power of each main tone is varied from the Start-Power to Stop-Power in the specified number of power points. The power of each tone can be set (on the Power tab) individually or as a pair by checking Coupled Tone Power.

CW

The main tone frequencies and power levels are held constant. Measurements are taken for the specified Number of Points. The X-axis is number of points.

Segment Sweep fc (Swept IMD ONLY)

Same as Sweep fc except that the center frequencies are constructed using the standard segment table. Learn how.

LO Power Sweep (Swept IMDX ONLY)

The main tone frequencies and power levels are held constant. Measurements are taken for the specified number of points. The X-axis is LO Power.

Segment Sweep Notes: (Swept IMD ONLY)

- The segment table shown on the dialog is ‘READ-ONLY’.

- Learn how to Create and edit the Segment Sweep table.
- **Independent IFBW** and **Power** are NOT available.

- **X-axis point spacing** is available beginning with A.09.10.

**Number of Points** Enter the number of data points for each sweep. See Limited Number of Acquisitions.

**IFBW**

The **Narrowband IF path** is used for IMD measurements to help reduce spurious responses. Because the narrowband filter has a bandwidth of about 28 kHz, using an IFBW greater than 30 kHz does nothing to improve measurement accuracy.

**Main Tone and IM Tone IFBW** IF Bandwidth is specified separately for the main tones (f1 and f2) and for the intermodulation tones. This allows the higher-power main tones to be accurately measured at a higher - and faster - IFBW, while the lower-power product tones to be accurately measured a lower - and slower - IFBW.

**Note:** The IFBW is limited to 600 kHz when performing Swept IMD measurements even if the **Wide IF path** is selected.

**Reduce IF BW at Low Frequencies** - On the VNA, the trace noise becomes worse below 748 MHz. This is especially obvious between 10 MHz and 45 MHz. When this box is checked, the VNA uses a smaller IF Bandwidth than the selected value at frequencies below 748 MHz. Learn more about the selected values.

**Note:** For Main Tone IFBW settings of 1 kHz and below, and when the center frequency of the VNA is an exact multiple of 10 MHz, then the tone frequencies are shifted UP by (10 * IFBW) for the entire IFBW sweep. If those frequencies would exceed the maximum frequency of the analyzer, then the frequency is shifted DOWN by (10 * IFBW). This is done to avoid interference with 10 MHz reference signals.

---

**Power tab - IMD Setup** dialog box help
Configures RF power and Power Sweep settings for IMD measurements.

**Power ON (All channels)**  Check to turn RF Power ON or clear to turn power OFF for all channels.

**DUT Input Port**

**Input Port** Choose Port 1 or Port 3. When using Port 3, an external combiner is required. [Learn more.](#)

**Source Attenuator** Specifies the port 1 attenuator. This attenuator affects the range of available power into the DUT. [Learn more about Source Attenuation.](#)

**Receiver Attenuator** This attenuation setting protects the A receiver from damage.

**DUT Output Port**

**Output Port** Choose Port 2 or Port 4 (with limitations). [Learn more.](#)

**Source Attenuator** This setting is used to improve the load match at the DUT output. Select 0 dB for power levels up to 10 dBm, and increase by 10 dB for every 10 dBm more output power.

**Receiver Attenuator** Specifies the attenuator setting for port 2. When the power into the receiver test port is around +10 dBm, the VNA receiver may be in compression. However, with receiver attenuation, lower input power levels may become too noisy to make accurate power measurements. In this case, lower IFBW for the IM tones to reduce noise. [Learn more about Receiver Attenuation.](#)

**Tone Powers**

**Coupled Tone Power** Check to set the same power level for each main tone using the f1 Power setting. Clear to set different f1 and f2 power levels.

**ALC On** Check to use internal ALC hardware (default). Clear to use Open Loop hardware. Open Loop leveling should only be used when doing Wideband Pulse measurements.
Power Leveling

Because the gain of the DUT can be different for the f1 and f2 tone frequencies, you can set tone power at either the input of the DUT OR the output using the following methods. Receiver Leveling will cause slower sweeps.

Set Input Power (Default) The specified f1 and f2 power levels are set at the DUT input. Input power level accuracy is based ONLY on the source power cal that is performed during the IMD cal. The input and output tones may NOT be equal or flat.

Set Input Power, receiver leveling The specified f1 and f2 power levels are set at the DUT input using receiver leveling at the input reference receiver. This ensures the tone power levels are equal at the DUT input. However, the output tones may NOT be flat due to variations in the gain of the DUT at different frequencies.

Set Input Power, equal tones at output The specified f1 and f2 power levels are set at the DUT input and a measurement is made at the output. The inputs are adjusted once at each frequency to ensure the tone power levels are equal at the DUT output. However, the output tones may NOT be flat due to variations in the gain of the DUT at different frequencies.

Set Output Power, receiver leveling The specified f1 and f2 power levels are set at the DUT output. Receiver Leveling at the output receiver is used to accurately set the specified power level of each tone within the tolerance value that is set in the Receiver Leveling dialog. This setting results in the output tones being equal and flat across the frequency range.

Note: Enabling Safe Mode when using receiver leveling may be necessary to ensure stable results.

f1 / f2 Power

Fixed f1 Power Specify the power level for f1 at either the DUT input or output depending on the Power Leveling setting. Choose a value between -30 dBm and +30 dBm. When "Coupled Tone Power" is checked, power is set for both f1 and f2 tones.

Fixed f2 Power Available when Coupled Tone Power is NOT checked. Specify the power level for f2 at either the DUT input or output depending on the Power Leveling setting. Choose a value between -30 dBm and +30 dBm.

Start, Stop, and Step f1 and f2 Power Available when Power Sweep is selected on the Frequency tab. Sets the Start and Stop power levels for f1 and f2, either individually or together with Coupled Tone Power checked.

Path Configuration click to launch the RF Path Configuration dialog.

Highlighted Note: RF2 tone power offset to compensate for combiner loss.
This message appears when the f2 tone is being supplied by an external source. The tone power has been increased on the external source to compensate for loss through the internal combiner. For example, if the tone power at the DUT should be 0 dBm, the power out of the source will be about 15 dB higher.

To accommodate single-source VNA models, an external source can be used for the f2 tone. Learn how to configure an external source and combiner to make Swept IMD and IMDx measurements.

f1 Always uses VNA internal source 1.

f2 Select a source to be used for the f2 tone. This selection is available when an external source is configured and the Active box is checked on the External Source Configuration dialog.

Buttons

Add Source Click to configure an external source using the External Source Configuration dialog.

Path Configuration Click to launch the Path Configuration dialog (PNA-X models only).

How to add IMD Parameters

Using Hardkey/SoftTab/Softkey
1. Press Trace, then select trace.
2. Press Meas, then select a parameter.

Using a mouse
1. Click Instrument, Trace, Add Trace.
2. Click Response, Meas, then select a parameter.

IMD New Trace dialog box help
Use this dialog to select IMD and IMDx parameters to measure and display.

Up to five parameters can be selected at a time, then click **Apply** to create those traces. Then select more without closing the dialog. There is no limit to the number of traces and windows allowed in the VNA.

**Note:** Calculations are NOT performed to determine if the frequency of a selected intermod (Order) product will be within the frequency range of the VNA. Measurements that fall outside of the frequency range of the VNA are displayed as -200 dB.

**Param Name** This name is built from the selected **Type**, **Tone Select**, **Order**, and **Measure At** settings. Once built, check to measure this parameter.

**Type**

Select the type of Swept IMD measurement to make.

The characters below (in parenthesis) are used in the Param Name.

- **Tone Power (Pwr)** Measures the absolute power of the specified (ordered) tone. [Learn more.](#)

- **IMD Relative to Carrier (IM)** Measures the power of the specified (ordered) tone relative to the power of the f1 or f2 tone. [Learn more.](#)

- **Input Referred Intercept Point (IIP)** From DUT measurements, calculates the theoretical power level at which the specified (ordered) intermod product will be the same power level as the carrier at the output of
the DUT. The input power to the DUT at which this output power occurs is reported and displayed. Learn more.

- **Output Referred Intercept Point (OIP)** From DUT measurements, calculates the theoretical power level at which the specified (ordered) product will be the same power level as the main tone at the output of the DUT. This value is reported and displayed. Learn more.

- **CTB Band-Edge Distortion (CTBE)** Composite 'Triple Beat' Distortion - 3rd order DUT OUT only. Learn more.

- **CTB Mid-Band Distortion (CTB)** Composite 'Triple Beat' Distortion - 3rd order DUT OUT only. Learn more.

- **CSO Distortion (CSO)** 2nd order DUT OUT only. Learn more.

- **XMOD 3rd Order Crossmod (XMOD)** 3rd order DUT OUT only. Learn more.

- **Tone Gain (ToneGain)** From tone power measurements, calculates the Output Tone power / Input Tone power for the specified tones. Learn more. The Input and Output tones are different frequencies.

**Tone Select**

Select the tone (High, Low, or Both) to be measured and displayed (See image).

- **High** Measure and display the power of the specified (ordered) tone on the High side of the main tones.

- **Low** Measure and display the power of the specified (ordered) tone on the Low side of the main tones.

- **Avg** Measure the specified (ordered) tones on both the High and Low sides. Then calculate and display the Average power level. **Note:** The Avg power level is actually the mean dB value of the high and low side tones - NOT the true average power level which would be calculated from linear power (watts). This is done to be consistent with the Power Leveling algorithm that is used to set the tone powers equal at the DUT input or output, which is also based on the dB values.

- **Max** Measure the specified (ordered) tones on both the High and Low sides. Then find and display the Maximum power level.

- **Min** Measure the specified (ordered) tones on both the High and Low sides. Then find and display the Minimum power level.

Used to build the parameter name:

- **Hi, Lo, Max, and Min** are appended to the Param Name when selected.

- **Nothing is appended to the Param Name when Avg (default setting) is selected.**
### Important Notes - 2nd-order products

- 2nd-order products are likely to be outside of the VNA frequency range. When that occurs, trace data will show **all zeros** (linear) or **-200 dB** (log magnitude).
- If either the High or Low side falls outside the frequency range of the VNA, then **Avg** is NOT allowed.
- When displaying 2nd-order traces, **Avg** is NOT allowed, even when both the Lo and Hi products are displaying valid data. This is because 2 Low and 2 High products are usually very different from one another.
- When performing a calibration that is meant to include 2nd-order products, be sure **Include 2nd Order Products** is checked in the first Calibration dialog box.

### Order

Specify the intermodulation product to measure. Choose from 1, 2, 3, 5, 7, 9.

- **Main** is appended to the Param Name when 1 is selected.
- Otherwise, the tone number is appended to the parameter name.

### Measure At

Measure the selected parameter at either:

**DUT Input**

- **In** is appended to the Param Name (ex: PwrMainIn).
- The **input port reference receiver** is used to measure the fundamental tones and the required products.

**DUT Output**

- Nothing is appended to the Param Name (ex: PwrMain).
- The **output port measurement receiver** is used to measure the fundamental tones and the required products.
1. At the first page of the IMD Cal Wizard, you tell the calibration routine the frequencies at which the calibration is to be performed. Optionally, you can choose to perform the source power cal at only the center frequency midway between the main tones. For IMDx ONLY, you can also choose to perform a source power cal of the LO source.

2. The VNA calculates an array of source and receiver frequencies that incorporate all the main tone frequencies (low and high) and the specified product tone frequencies.

3. Using a power meter at port 1, a source and receiver calibration is performed to calibrate the R1 reference receiver to be a fast and ‘tunable’ power meter.

4. The R1 reference receiver is then used to perform a source power cal of the main tone frequencies: first the Source 1 / Low tone then the Source 2 / High tone. Both sources are left ON while each tone is measured in order to duplicate the impedance match under which the measurement will be performed.

5. Then a standard 2-port SOLT cal is performed at all frequencies using either an ECal module or mechanical standards. The 2-port cal is used to correct the source calibration R1 tracking terms for the match of the power sensor. It is also used to transfer the R1 tracking term to the B receiver.

Notes

- If the main tone frequencies change but are within the frequency range in which the calibration was performed, then the calibration becomes interpolated C*. This can occur by changing the start/stop/center/span frequencies, the number of points, or the sweep type. Learn more about Interpolation.

- A Preference setting can be made which allows a Swept IMD and IMDx calibration to exceed the stop frequency limit of an ECal module. Learn more.

- Receiver calibrations that are performed in a standard channel can be applied to a Swept IMD channel. However, Source Calibrations can NOT be applied.

See Also

IM Spectrum Calibration

How to start a Swept IMD Calibration

Using Hardkey/SoftTab/Softkey

1. Press CAL > Main > Smart Cal....

Using a mouse

1. Click Response
2. Select Cal
3. Select Smart Cal...
Select Tone Products - IMD Cal dialog box help

Calibration Mode

Match corrected Response Cal  This selection performs a full IMD calibration as described in the above Calibration 'Overview'.

**Response ONLY (Normalization)**  This IMD Cal does NOT correct for the mismatch of the power sensor. Choose this if you have a test configuration that does not easily accommodate making match measurements. Instead of a standard 2 port SOLT cal (Step 5 above), only the transmission tracking term is measured and used to transfer the R1 receiver tracking term (produced by the power sweep) to the B receiver.

**Note:** For the Response (Normalization) Cal, it is assumed that a zero-length THRU standard is being used to connect port 1 to port 2. If an adapter is used, there is NO compensation for delay or loss of the adapter. This can NOT be changed.

Tone Power Cal

**Calibrate only at center frequencies**  The source power cal portion is performed at only the center frequency, which is midway between the main tones. This cuts the source power calibration time (the slowest part of the calibration) in half. The measurement is interpolated although the C* annotation is not shown in the status bar.

**Calibrate at all frequencies**  The source power cal portion is performed at all main tone frequencies.

**Enable LO Power Calibration**  (IMDx ONLY) Check to perform a standard power calibration of the LO source as part of the calibration process.

Select Product Tones

**Max Product**  Select the highest product that you will be measuring. The low and high frequencies for that product, and all lower 'odd' order products will be calibrated. For example,
when 5th Order Product is selected, the frequencies for the Main Tones, and the Low and High order products for the 3rd and 5th order products will be included in the calibration.

**Include 2nd Order Products** Check to calibrate the 2nd-order products in the frequencies to be calibrated. The frequencies of these products are usually far from the main tones.

**Extrapolated ECal Calibration**

A *Preference setting* can be made which allows a Swept IMD and IMDx calibration to exceed the stop frequency limit of an ECal module. The error terms beyond the ECal stop frequency are extrapolated.

The setting is designed to provide correction for order products (especially 2nd order) when the frequencies exceed the ECal module stop frequency.

- The accuracy of the extrapolated error terms is not as good as those that were measured.
- This feature is NOT allowed when using *Cal All Channels*.
- The ECal Extrapolation state and frequencies are saved to the Cal Set and visible in the Cal Set Properties dialog.

---

**Select DUT Connectors and Cal Kits - IMD Cal dialog box help**

If *Response Only* is selected on the previous page, click **Source Cal Settings...** to change the Source Cal settings, or click **Next>** to continue.
Otherwise, this is a standard VNA Cal Wizard page except for the following:

**Modify Cal**  Check, then click **Next**, to Modify Cal (Standards AND Thru Method).

**Source Cal Settings**  Click to launch the **Source Cal Settings** (for Apps) dialog.

Learn more about IMD Calibration

---

**Power Cal Settings - IMD Cal** dialog box help

A power calibration is performed on port 1. This is done to simplify subsequent IMD calibrations since the power calibration will not need to be repeated.

For highest accuracy, connect the power sensor directly to the test port with no adapter.

Learn more about these settings.
**IMD Calibration Step 1** dialog box help

**Power Level** at which to perform the Port 1 Source Power Cal.

It is usually best to perform the Source Power Cal at 0 dBm because the power sensor is calibrated at that level.

However, if a component is used between the VNA source and the calibration reference plane, then adjust the power level so that the power at the sensor is about 0 dBm if possible.

Learn more about IMD Calibration

The remaining dialog pages are the same as the standard Cal Wizard.

**Saving Swept IMD Power Data**

Swept IMD power data, Log Mag ONLY, can be saved to a *csv file. This data type can be read by spreadsheet programs, such as Microsoft Excel. Data from the last complete sweep is saved to the specified *csv file.

**How to save Swept IMD (and IMDx) data**

With a Swept IMD or IMDx channel active...

Using **Hardkey/SoftTab/Softkey**

1. Press **Save Recall** > **Save Other** > **Save Data**...  
2. File Type= CSV Formatted Data (*.csv).

Using **a mouse**

1. Click **File**  
2. Select **Save Data**  
3. File Type= CSV Formatted Data (*.csv).

**Notes:**
- For every tone, six power levels are saved in this order: OUT Avg | OUT Lo | OUT Hi | IN Avg | IN Lo | IN Hi
- Power levels for the Main tones are always saved, regardless of the active measurement.
- All tones that are displayed are also saved. For example, any displayed 3rd order tone parameter causes the 3rd tone power levels to be saved.
- Only tone powers are saved. Calculated parameters, such as IMD Relative to Carrier (IM) are NOT saved.
- If calibration is turned ON when the file is saved, then all data is calibrated. Otherwise, raw data is saved.

This image shows the 6 Main tone power levels that are always saved.

In the power parameter labels, Output and Avg are implied as in the parameter selection. For example: PwrMain = Average Output power of the main tones.
Intermodulation Spectrum measurements can be made independently, or coupled with Swept IMD measurements. They are a distinct measurement class. Therefore, IM Spectrum measurements are always made in a separate channel.

- Limitations

- Create an IM Spectrum Channel

- Select IM Spectrum Parameters

- Start the IM Spectrum Setup dialog
  - Frequency tab
  - Power tab
  - Tracking tab
  - Configure tab

- Calibration

See Also

- IM Spectrum Concepts
- IMx Spectrum for Converters
- Swept IMD Measurements
- Programming commands

Limitations with IM Spectrum

The following VNA features are NOT available in an IM Spectrum channel:

- Unratioed receiver measurements (A, B, R)
- ECal User Characterization
- FOM or FCA
- Time Domain
- Balanced measurements
- Save Formatted Citifile data.
- Save SnP data.
- External sources
- External Test Set Control (Option S93551A/B)
- Port Extensions
- Integrated Narrowband or Narrowband Pulse App
- DC Meter parameters
- Option 205 or Option 425

Create an IM Spectrum Channel

An IM Spectrum channel can be created independently from a Swept IMD channel or coupled with the stimulus settings of an existing Swept IMD measurement.

To create an independent IM Spectrum channel

1. Press **Meas > S-Param > Meas Class**.
2. Select **IM Spectrum**, then either:
   - **OK** delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.

To create an IM Spectrum channel from an existing Swept IMD channel

1. With a Swept IMD channel active, press **Marker** on the front panel.
2. Move the marker to the data point of interest. It can be moved later.
3. Press **Marker > Marker Functions > Marker -> IM Spectrum**.

This creates or configures an existing IM Spectrum channel based upon the configuration of the Swept IMD channel.
To perform this function remotely:

Execute the VBScript file located on the VNA.

1. Create a macro. Either send this command or it can be done manually:
   `SYST:SHOR10:PATH "C:\Program Files (x86)\Keysight\Network Analyzer\Applications\IMD\imd.vbs"`

2. Execute the macro using: `SYST:SHOR10:EXEC` or the equivalent COM command: Execute Shortcut Method

Notes:

- 10 is the macro number. If you are already using this number for a macro, it will be overwritten.
- The marker must already be on the IMD trace and be the active marker.
- The path name to imd.vbs may be different on your VNA.

Select IM Spectrum Parameters

How to add IM Spectrum traces

With the IM Spectrum channel active and a Tr1 Output trace displayed:

Using Hardkey/SoftTab/Softkey

1. Press Trace, then select trace.
2. Press Meas, then select a parameter.

Using a mouse

1. Click Instrument, Trace, Add Trace.
2. Click Response, Meas, then select a parameter.

New Trace - IM Spectrum - dialog box help

Choose from:

- **Output Spectrum** View signals OUT of the DUT and into VNA port 2 (B receiver).
- **Input Spectrum** View signals IN to the DUT (R1 receiver). Use this when measuring IM product frequencies to determine the power level of spurious signals into the DUT at those frequencies.
- **Reflection Spectrum** View signals reflected off the DUT input and back into VNA port 1 (A receiver)
IM Spectrum Setup Dialog

How to start the IM Spectrum Setup dialog

With an IM Spectrum measurement active:

Using Hardkey/SoftTab/Softkey

1. Press Sweep > Main > IMS Setup....

Using a mouse

1. Click Stimulus
2. Select IMS Setup...

Frequency tab - IM Spectrum dialog box help

![Image]

**Note:** The number of data points in an IM Spectrum channel = 3 * SPAN / ResBW. This can NOT be changed directly.

Configures the Sweep Type and frequency range for IM Spectrum measurements.

**Sweep Type**

Provides several methods to tune the IM Spectrum **receivers** (NOT stimulus) to view the power spectrum of arbitrary frequency ranges or various distortion products.

**Linear**  Allows the start/stop or center/span receiver frequencies to be set arbitrarily for f1 and f2. Enter the Response Settings below.

**2nd Order**  Couples the receiver frequency range to f1 and f2 to provide a convenient means of observing the spectrum surrounding ONLY the high-side 2nd order harmonic where Center = (f1 + f2), Span = 3 * (f2 – f1). **Note:** The center frequency is NOT set on the main tones.

**3rd Order**  Couples receiver frequency range to f1 and f2 to provide a convenient means of observing the spectrum surround the main-tones including the 3rd Order products where Center = (f1 + f2) / 2, Span = 4 * (f2 – f1).
**Nth Order** Couples receiver frequency range to \( f_1 \) and \( f_2 \), providing a convenient means of observing the spectrum surrounding the main-tones for an arbitrary span where \( \text{Center} = \frac{(f_1 + f_2)}{2} \), and \( \text{Span} = N \times (f_2 - f_1) \). This allows you to set the span arbitrarily, but have the center frequency track the main tone frequencies.

**Resolution BW**

The IM Spectrum channel utilizes a set of Gaussian filters instead of the standard VNA IF filters in order to provide similar behavior to traditional spectrum analyzers.

The comprehensive list of filters that are available for IM spectrum are: 3 MHz, 1 MHz, 600 kHz, 300 kHz, 150 kHz, 100 kHz, 60 kHz, 10 kHz, 3 kHz, and 1 kHz.

**Note:** The 10 kHz Res BW filter can generate image signals which are not a product of the DUT. You can verify the integrity of a questionable signal by switching to the 3 kHz or 60 kHz filter and looking for the image signal in the same location.

Not all of these filters are available for all measurements. Narrower filters are available for use with narrower frequency spans, and wider filters are available for wider spans.

**Stimulus Settings**

Allows configuration of the main-tone frequencies. Available ONLY when Tracking is OFF.

- \( fc \) (main-tone center frequency) = \( \frac{(f_1 + f_2)}{2} \)
- \( \Delta f \) (main-tone frequency separation) = \( f_2 - f_1 \)
- \( f_1 \) = Low-side main-tone frequency
- \( f_2 \) = High-side main-tone frequency

**Response Settings**

Allows configuration of the frequency range to sweep the receivers in terms of start/stop or center/span. Available for Linear sweep types ONLY.

- **Start Spectrum** = first frequency point of the power spectrum sweep
  - **Note:** The minimum **Start Spectrum** frequency that can be set is 40 MHz.
- **Stop Spectrum** = last frequency point of the power spectrum sweep
- **Center Spectrum** = \( \frac{(\text{Start Spectrum} + \text{Stop Spectrum})}{2} \)
- **Span Spectrum** = (Stop Spectrum – Start Spectrum)

---

### Power tab - IM Spectrum - dialog box help

![Power tab screenshot]

Learn about this dialog

---

### Tracking tab - IM Spectrum - dialog box help

![Tracking tab screenshot]

These selections are NOT available when there is no Swept IMD channel.

Allows the IM Spectrum channel to use (track) the **stimulus** (Main Tone) settings of an existing Swept IMD channel.

**Tracking Enable**  Check to use the frequency and power stimulus settings from the specified IMD channel. When enabled, stimulus settings on the Frequency Tab are disabled and ALL stimulus settings, such as frequencies, power and attenuator settings, and calibration, are copied from the Swept IMD channel to the IM Spectrum channel.
In the top Swept IMD window the main tones are swept from 4 GHz to 6 GHz with some specified delta F tone separation. The marker is on the center data point at 5 GHz.

The bottom IM Spectrum window center frequency is the same as the above marker: 5 GHz, but it has a much narrower frequency range of +/- 500 kHz. The IM Spectrum channel sets the receiver to see the two main tones, plus the third, fifth, and seventh-order products.

**Step Mode**

When tracking is enabled, set the method by which the IM spectrum measurement tracks the IMD channel.

**Manual Step** When selected, IM Spectrum measurements occur at only the specified IMD channel data point. **Stimulus Point** specifies the data point, by number, in the Swept IMD channel which has the stimulus settings to use for the IM Spectrum sweep.

**Automatic Step** When selected, causes the IM Spectrum channel to sequentially setup each of the stimulus conditions through which the Swept IMD channel sweeps the DUT. Each sweep of IM Spectrum is performed using the next set of stimulus conditions.

For example, in the above image, the first Swept IMD data point is at 4 GHz. The first IM
Spectrum sweep uses a center frequency of 4 GHz. The following IM Spectrum sweep would be at the second Swept IMD data point or 4.01 GHz, and so forth. After the last data point in the sweep is reached, the IM Spectrum channel begins again at the first Swept IMD data point. The only indication that Tracking is enabled is in Automatic mode, you can see the center frequency increment with each IM Spectrum sweep.

**Configure tab- IM Spectrum Setup - dialog box help**

To accommodate single-source VNA models, an external source can be used for the RF2 tone.

Learn how to configure an external source and combiner to make Swept IMD, IMDx, IM Spectrum, and IM Spectrum for Converters measurements.

**RF1** Always uses VNA internal source 1.

**RF2** Available for selection when an external source is configured and Active.

**Add Source** Click to configure an external source using the External Source Configuration dialog.

**Path Configuration** Click to launch the Path Configuration dialog (PNA-X models only).

**Calibration**

A calibration corrects the IM Spectrum source and receiver power level accuracy of the displayed Tones.

A calibration of the IM Spectrum channel is NOT performed using the Calibration wizard. An IM Spectrum channel is calibrated by applying a Cal Set in one of the following ways:

- A cal that was used on a Swept IMD channel. The Cal Set can be applied to the IM Spectrum channel using
the Manage Cal Sets dialog (Learn how) or from the Marker =>IM Spectrum softkey (Learn how). However, a Swept IMD channel with Sweep Type = Power Sweep can NOT be applied to a IM Spectrum channel. This is because a Cal Set for power sweep contains only a CW frequency and the IM Spectrum channel requires a swept frequency range. Zero Span is not supported in an IM Spectrum channel.

- A Source Power and Receiver Cal Set from a standard channel calibration. This can be a full calibration, but must include a source power cal for the source port (1) and receiver cal for the B receiver. Only one source can be corrected. Learn how to apply a standard Cal Set.

See Swept IMD Calibration
Conceptually, Swept IMD for Converters (IMDx) is like IMD for Amplifiers, except that there are two sets of products on the DUT output.
This image shows the simplest measurement configuration: the Input, LO, and Output are all CW frequencies.

- **DUT INPUT** - two fundamental tones (f1 and f2)
- **DUT LO** - a single frequency.
- **DUT OUTPUT** - two sets of frequencies

With IMDx, you can view EITHER the High side or Low side products; NOT BOTH.

Make this selection on the **Mixer Frequency tab**.

- High side (Input **PLUS** LO and all ordered products)
- Low side (Input **MINUS** LO and all ordered products)

### Requirements and Limitations

Swept IMDx requires **Swept IMD** (Opt S93087A/B).

- When using **Integrated Pulse application**, the IF Filter setting for the relevant receiver must be changed to 'Wide'. Learn how. The default IF Filter setting in Swept IMD is 'Narrow' in order to avoid spurs and harmonics. The IF Attenuator setting (on the same dialog) may also require adjusting.

The following VNA features are **NOT** available with Swept IMDx:

- Analog Sweep (**Stepped sweep** mode only)
- **Log frequency** sweeps
- **Unratioed receiver measurements** (A, B, R)
- **ECal User Characterization**
- **Time Domain**
- **Balanced measurements**
- **Save Formatted Citifile data.**
- **Save SnP data.**
- **Port extensions**
- **Some Fixturing Features**
• External Test Set Control (Option S93551A/B)

• Integrated Narrowband or Narrowband Pulse App

• Independent IFBW, Power Levels, or Sweep Time in a segment table is NOT supported.

• External DC Sources (DC Meters ARE supported).

• See Frequency Limitations

• Option 205 or Option 425

**Note:** Beginning with A.09.00, **Embedded LO** measurements are allowed in IMDx and IMSpectrum. Configure the measurement as you would with SMC. Learn how.

### How to Configure your Hardware

The VNA is extremely versatile, and can be configured in many ways to make IMDx measurements. While not all conceivable configurations are documented here, a few of the most common examples are provided to show the basic concepts.

#### DUT Configuration

- The DUT Input must be connected to VNA Port 1 which supplies the f1 and f2 tones.
- The DUT Output must be connected to VNA Port 2 which uses the VNA B receiver.
- See LO Source configuration below.

#### Source Configuration

Three sources are required to make IMDx measurements: Two sources are VNA internal; the third is an external source.

- **F1 tone** - Must come out VNA Port 1 Default is Src 1
- **F2 tone** - Must come out VNA Port 1 through the internal combiner. This source can come from an internal or external source. Default is the internal Src 2.
- **LO** - Can come from internal or external source. Default is Not controlled, set to 0 Hz.

  - If using the internal Src 2 source, the f2 tone must come from an external source through the rear-panel.
If using an external source, it can be connected directly to the DUT, or through the VNA Port 3 or Port 4 using the Path Configurator.

**Configuration Examples**

Three configurations are shown below to illustrate how to use the Path Configurator to manually make switch settings.

IMDx and IMx Spectrum channels are ALWAYS configured using the default configuration switch settings shown in the image below. This is NOT the same default configuration that is used for an S-parameter channel. A manual switch setting is required whenever a custom configuration is used.

The manual switch setting, which must be done every time an IMDx or IMx Spectrum channel is created, can be saved using the Store button in the Path Configurator. Then save the entire IMDx setup as an Instrument State. This will load the custom Path Configuration when the Instrument State is recalled.

1. **Default Configuration**  No manual switching required.
   
   - **f1** - Internal Src1
   - **f2** - Internal Src2 through combiner in Normal
   - **LO** - Connect external source directly to the DUT LO.

2. **External LO though Port 3 or Port 4 (4-port VNA only)**

Use this configuration to monitor LO power using R3 or R4. (Future parameters)
- **f1** - Internal Src1
- **f2** - Internal Src2 through combiner in Normal
- **LO** - Connect external source through the rear-panel (J7 for Port 3; J3 for Port 4).
  - Connect the DUT LO to VNA Port 3 or Port 4.
  - For Port 3, NO switching is required.
  - For Port 4, switch Port 4 Bypass Switch to **Rear Panel**.

3. **Internal LO through Port 3 or Port 4; External f2 through rear-panel (4-port VNA only)**

   This configuration is commonly used for FCA measurements where the internal second source is used as the LO.

   - **f1** - Internal Src1
   - **f2** - External f2 through rear-panel J9. No switching required.
   - **LO** - Internal Src2 through:
     - Port 3  Switch Port 3 Bypass Switch to **Thru Path**. Connect the DUT LO to VNA Port 3
     - Port 4  Switch Port 4 Bypass Switch to **Thru Path**. Connect the DUT LO to VNA Port 4

### Create a Swept IMDx Measurement

1. Press **Meas > S-Param > Meas Class....**

2. Select **Swept IMD Converters**, then either:
   - **OK** delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.

3. A Swept IMDx measurement is displayed. To select additional parameters to display, click **Response**, then **Measure**, then select a parameter from the list.
How to start the Swept IMDx Setup dialog

Using **Hardkey/SoftTab/Softkey**

1. Press  **Freq > Main > IMDX Setup...**

Using a mouse

1. Click  **Stimulus**
2. Select  **IMDX Setup...**

### Programming Commands

---

**Valid Mixer Configuration / Sweep Type Combinations**

#### Tips

Although you will soon become comfortable navigating these tabs, at first it may be best to complete the dialog in the following order:

1. On the **Tone Frequency tab**, set the Sweep Type and Tone Frequencies.
2. On the **Mixer Frequency tab**, set a valid mixer configuration.

- Input center / fixed frequency CAN be set on both the **Tone Frequency tab** and the **Mixer Frequency tab**. When you set one, the other is updated automatically.

---

The following are the **Valid Mixer Configurations** in table format:

<table>
<thead>
<tr>
<th>Sweep Type</th>
<th>Input</th>
<th>LO</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Swept Input (Fixed Tones)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sweep fc</td>
<td>Swept</td>
<td>Swept</td>
<td>Fixed</td>
</tr>
<tr>
<td></td>
<td>Swept</td>
<td>Fixed</td>
<td>Swept</td>
</tr>
<tr>
<td><strong>All Ranges Fixed</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CW</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sweep Delta F</td>
<td>Fixed</td>
<td>Fixed</td>
<td>Fixed</td>
</tr>
<tr>
<td>(Tone) Power Sweep</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LO Power Sweep</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Fixed Input / Swept LO &amp; Output</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CW</td>
<td>Fixed</td>
<td>Swept</td>
<td>Swept</td>
</tr>
</tbody>
</table>
For determining a valid mixer configuration with 2 LOs, one Fixed LO and one Swept is equivalent to having a single-stage Swept LO.

If you create an invalid Sweep Type / Mixer Configuration, a red message appears like the following:

**ERROR: Input range must be swept in Linear sweep mode.**

- If this occurs, change the **Sweep Type** on the Tone Frequency tab.
- See other rules for configuring a mixer.
- The following is an explanation of the table:

**SWEPT Input** - Sweeps the center frequency of the tones, but the tone spacing remains fixed.

On the Tone Frequency tab, select **Sweep fc**

- Either the LO or Output MUST also be swept.
- On the Mixer Frequency tab, select Start/Stop, or Center/Span for each range to be swept.

**FIXED Input** - The center frequency of the tones is fixed.

On the Tone Frequency tab select **Fixed** for the following sweep type:

- **CW** - Tones do NOT change. The LO and Output frequencies CAN be swept.

**All Ranges FIXED** - The center frequency of the tones is fixed.

On the Tone Frequency tab select **Fixed** for the following sweep types:

- **Sweep Delta F** - Tone separation changes.
- **Power Sweep** - Tone power changes.
- **LO Power Sweep** - The LO power is swept.
- **CW** - Tones do NOT change.

**Setup Dialog**

The following tabs are shared with the Swept IMD Setup dialog:

- **Tone Frequency tab**
• Tone Power tab

The following tabs are shared with all Mixer / Converter Applications

• Mixer Frequency tab
• Mixer (LO) Power tab
• Mixer Setup tab (NOT shared)

**Tone Frequency tab - Swept IMDx Setup -dialog box help**

![Tone Frequency tab screenshot]

Learn about this dialog.

**Tone Power tab - Swept IMDx Setup -dialog box help**

![Tone Power tab screenshot]

Learn about this dialog.
Mixer Frequency tab - Swept IMDx Setup - dialog box help

Conduct frequency sweep

Learn about this dialog

Mixer Setup tab - Swept IMDx Setup - dialog box help

Converter Stages Choose from 1 or 2 stage DUT (# of LOs).

Converter Model

Learn how to configure an external source and combiner to make Swept IMD and IMDx measurements.

**f1** Always uses VNA internal source 1.

**(DUT Input) Port N** (Red circle in image) Normally Port 1 is used. This selection is available to allow the two tones to be routed through port 3 on 4-port VNA models. Learn more.

**f2** Select a source to be used for the f2 tone. This selection is available when an external source is configured and the Active box is checked on the External Source Configuration dialog.

**LO1 (and LO2 for 2-stage DUTs)** Select the source to use for the specified LO. Available for
selection when an external source is configured and the Active box is checked on the External Source Configuration dialog.

(DUT Output) Port N  Available for selection on 4-port VNA models. Select the VNA port to connect to the DUT Output.

Fractional Multipliers - Learn more.

Hardware Configuration

Add Source  Click to configure an external source using the External Source Configuration dialog.

Path Configuration  Click to launch the Path Configuration dialog (PNA-X models only).

Mixer (LO) Power tab - Swept IMD Setup - dialog box help

Learn about this dialog

How to add Swept IMDx Parameters

Using Hardkey/SoftTab/Softkey

1. Press Trace, then select trace.
2. Press Meas, then select a parameter.

Using a mouse

1. Click Instrument, Trace, Add Trace.
2. Click Response, Meas, then select a parameter.
This dialog is shared with Swept IMD for Amplifiers.

Learn about this dialog.

**IMDx Calibration**

Calibration for IMDx is exactly the same as calibration for IMD with the following exception:

- You can choose to perform a source power cal of the LO source. If the LO is a fixed frequency, this step is performed very fast.

The results of an IMDx calibration are very similar to the results that are achieved from an SMC calibration.
IMx Spectrum for Converters

IMx Spectrum measurements are a distinct measurement class and therefore always made in a separate channel. This topic discusses all aspects of an IMx Spectrum measurement.

- Limitations
- How an IMx Spectrum Channel Works
- Create an IMx Spectrum Channel
- Select IMx Spectrum Parameters
- Start the IMx Spectrum Setup dialog
  - Frequency tab
  - Tone Power tab
  - Mixer Frequency tab (separate topic)
  - Mixer Power tab (separate topic)
  - Mixer Setup tab
- Calibration

See Also

- Swept IMD Measurements
- Programming commands

Limitations

IMx Spectrum requires IMD (Opt S93087A/B).
Note:

The **minimum** frequency of this application is **40 MHz**. (10 MHz plus 30 MHz).

The **maximum** frequency of this application is the MAX frequency of the VNA **MINUS 30 MHz**.

This is due to the use of the various IF frequencies.

The following VNA features are NOT available in an IMx Spectrum channel:

- Unratioed receiver measurements (A, B, R)
- ECal User Characterization
- FOM or FCA
- Time Domain
- Balanced measurements
- Save Formatted Citifile data.
- Save SnP data.
- External sources
- External Test Set Control (Option S93551A/B)
- Interface Control
- Port Extensions
- Fixturing Simulator
- Integrated Narrowband or Narrowband Pulse App
- Option 205 or Option 425

**Note:** Opt S93084A/B (**Embedded LO measurements**) is NOT necessary to make an IMx Spectrum measurement on converters with an embedded LO. This is because the embedded LO can be slightly off-frequency and the output tones will still be visible on the IMxSpectrum display. You may have to select **Linear** sweep type, then set the response frequency span wide enough to allow for a more off-frequency embedded LO. **Learn how.**
How an IMx Spectrum for Converters Channel Works

Before reading this topic, you should become familiar with IMD Concepts.

The IMx Spectrum channel provides a traditional Spectrum Analyzer view of the intermodulation distortion behavior of a converter output. Unlike the Swept IMDx channel, the main tones (F1 and F2) are fixed while the receiver is swept over a frequency range of interest in order to generate a display as shown below.

A typical Spectrum Analyzer does NOT have a signal source. This one does. The signal source, or stimulus, settings for the F1 and F2 main tones can be set in either the IMx Spectrum channel or the Swept IMDx channel. These settings include the frequencies and power levels of the main tones.

**Receiver Settings**

Although the stimulus settings can be set in either the IMx Spectrum channel or the Swept IMDx channel, the receiver settings are set ONLY in the Frequency tab of the IMx Spectrum dialog. These settings include how many tone products to view, which determines the center and frequency span and the resolution bandwidth. You can also set a Linear sweep type, then enter an arbitrary receiver frequency range.

You can choose from several resolution bandwidths which also determines the number of data points used in the channel. The higher the Res BW, the fewer the number of data points.

The IMx Spectrum channel performs multiple measurements for each data point in order to reject
unwanted images which are generated by the VNA internally. This provides a high degree of confidence that signals captured in a trace are real and are not spurious responses generated in the measurement process.

**Differences from IM Spectrum for Amplifiers**

An IMx Spectrum for Converters channel has the following important differences from an IM Spectrum channel for Amplifiers.

- In an IMx Spectrum channel, **all converter frequencies are fixed**. (Input, Output, LO, Tone Spacing, and Tone Power).

- The IMx Spectrum channel **CANNOT track** with the IMDx channel, as it can in IMD for Amplifiers.

**Create an IMx Spectrum Channel**

An IMx Spectrum channel can be created independently, or from an existing Swept IMDx channel.

**To create an independent IMx Spectrum channel**

1. Press **Meas > S-Param > Meas Class....**

2. Select **IMx Spectrum**, then either:
   - **OK** delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.

**To create an IMx Spectrum channel from an existing Swept IMD channel**

This creates or configures an existing IMx Spectrum channel based upon the configuration of the Swept IMD channel at the Marker frequency.

1. With a Swept IMD channel active, press the **Marker** key on the front panel.

2. Move the marker to the data point of interest. It can be moved again later.

3. Press **[Marker Functions]** then **[Marker -> IMx Spectrum]**

**Select IMx Spectrum Parameters**
How to add IMx Spectrum traces

With the IMx Spectrum channel active and a Tr1 Output trace displayed:

Using **Hardkey/SoftTab/Softkey**

1. Press **Trace**, then select trace.
2. Press **Meas**, then select a parameter.

Using a mouse

1. Click **Instrument, Trace, Add Trace**.
2. Click **Response, Meas**, then select a parameter.

### New Trace - IMx Spectrum -dialog box help

**Output Port 2, Incident**

View signals OUT of the DUT and into VNA port 2 (B receiver).

**Note:** Only viewing the OUTPUT tones is supported.

### IMx Spectrum Setup Dialog

**How to start the IMx Spectrum Setup dialog**

With an IMx Spectrum measurement active:

Using **Hardkey/SoftTab/Softkey**

1. Press **Sweep > Main > IMSX Setup....**

Using a mouse

1. Click **Stimulus**
2. Select **Sweep**
3. Select **IMSX Setup...**
Learn about this dialog and about *.mxr files.
Mixer Frequency tab - IMx Spectrum - dialog box help

Learn about this dialog and about *.mxr files.

Mixer (LO) Power tab - IMx Spectrum - dialog box help

Learn more about this dialog and about *.mxr files.
To accommodate single-source VNA models, an external source can be used for the RF2 tone.

Learn how to configure an external source and combiner to make Swept IMD and IMDx measurements.

**Converter Stages** Choose from 1 or 2 stage DUT (# of LOs).

**Converter Model**

- **f1** Always uses VNA internal source 1.

- **(DUT Input) Port N** Available for selection on 4-port VNA models. Select the VNA port (1 or 3) to connect to the DUT Input.

- **f2** Available for selection when an external source is configured and Active.

- **LO1 (and LO2 for 2-stage DUTs)** Select the source to use for the specified LO. Available for selection when an external source is configured and Active.

- **(DUT Output) Port N** Available for selection on 4-port VNA models. Select the VNA port to connect to the DUT Output.

**Hardware Configuration**

- **Add Source** Click to configure an external source using the External Source Configuration dialog.

- **Path Configuration** Click to launch the Path Configuration dialog (PNA-X models only).

**Calibration**

A calibration of the IMx Spectrum channel is NOT performed using a calibration wizard.
An IMx Spectrum channel is calibrated from a Cal Set that is used on a Swept IMDx channel. The Cal Set can be applied to the IMx Spectrum channel using the Manage Cal Sets dialog (Learn how) or from the Marker =>IMx Spectrum softkey (Learn how).

However, a Swept IMDx channel with Sweep Type = Power Sweep can NOT be applied to a IMx Spectrum channel. This is because a Cal Set for power sweep contains only a CW frequency and the IMx Spectrum channel requires a swept frequency range. Zero Span is not supported in an IMx Spectrum channel.

A calibrated IMx Spectrum trace corrects the source and receiver power level accuracy of the displayed Tones.

See Swept IMD Calibration

Learn how to apply a Cal Set to the IMx Spectrum channel
Time Domain

Note: Time Domain measurements are only available on analyzers with S9x010A. See Configurations.

Time Domain allows you to view a device response as a function of time. The following are discussed in this topic:

- Overview
- How the Analyzer Measures in the Time Domain
- Calibration for Time Domain
- Transmission Measurements
- Measurement Response Resolution
- Measurement Range and Alias Responses
- How to make Time Domain Settings
- Gating
- Advanced Settings

See the updated App Note: Time Domain Analysis Using a Network Analyzer.

Overview

In normal operation, the analyzer measures the characteristics of a test device as a function of frequency. With Time Domain (opt S9x010A), the frequency information is used to calculate the inverse Fourier transform and display measurements with time as the horizontal display axis. The response values appear separated in time, allowing a different perspective of the test device's performance and limitations.

The graphic below compares the same cable reflection measurement data in both the frequency and time domain. The cable has two bends. Each bend creates a mismatch or change in the line impedance.
The frequency domain S11 measurement shows reflections caused by mismatches in the cable. It is impossible to determine where the mismatches physically occur in the cable.

The time domain response shows both the location and the magnitude of each mismatch. The responses indicate that the second cable bend is the location of a significant mismatch. This mismatch can be gated out, allowing you to view the frequency domain response as if the mismatch were not present. Distance Markers can be used to pinpoint the distance of the mismatch from the reference plane.

How the Analyzer Measures in the Time Domain

Time domain transform mode simulates traditional Time-Domain Reflectometry (TDR), which launches an impulse or step signal into the test device and displays the reflected energy on the TDR screen. By analyzing the magnitude, duration, and shape of the reflected waveform, you can determine the nature of the impedance variation in the test device.

The analyzer does not launch an actual incident impulse or step. Instead, a Fourier Transform algorithm is used to calculate time information from the frequency measurements. The following shows how this occurs.

A single frequency in the time domain appears as a sine wave. In the following graphic, as we add the fundamental frequency (F0), the first harmonic (2F0), and then the second harmonic (3F0), we can see a pulse taking shape in the Sum waveform. If we were to add more frequency components, the pulse would become sharper and narrower. When the analyzer sends discrete frequencies to the test device, it is in effect, sending individual spectral pieces of a pulse separately to stimulate the test device.

During an S11 reflection measurement, these incident signals reflect from the test device and are measured at the A receiver. This is when the time domain transform calculations are used to add the separate spectral pieces together.
For example, consider a short length of cable terminated with an open. All of the power in the incident signal is reflected, and the reflections are 'in-phase' with the incident signal. Each frequency component is added together, and we see the same pattern as the simulated incident would have looked (above). The magnitude of the reflection is related to the impedance mismatch and the delay is proportional to the distance to the mismatch. The x-axis (time) scale is changed from the above graphic to better show the delay.

Alternately, the same cable terminated with a short also reflects all of the incident power, but with a phase shift of 180 degrees. As the frequency components from the reflection are added together, the sum appears as a negative impulse delayed in time.

Calibration for Time Domain

For simplicity, we have discussed incident signals reflecting off discontinuities in the test device. By far the most common network analyzer measurement to transform to time domain is a ratioed S11 measurement. An S11 reflection measurement does not simply display the reflections measured at the A receiver - it displays the ratio (or difference) of the A receiver to the Reference receiver. In addition, the S11 measurement can also be calibrated to remove systematic errors from the ratioed measurement. This is critical in the time domain as the measurement plane, the point of calibration, becomes zero on the X-axis time scale. All time and distance data is presented in reference to this point. As a result, both magnitude and time data are calibrated and very accurate.

The following shows where the time domain transform occurs in the analyzer data flow: (see Data Access Map)
1. Acquire raw receiver (A and R1) data
2. Perform ratio (A/R1)
3. Apply calibration
4. Transform data to time domain
5. Display results

Therefore, although a time domain trace may be displayed, a calibration is always performed and applied to the frequency domain measurement which is not displayed.

Transmission Measurements

The most common type of measurement to transform is an S11 reflection measurement. However, useful information can be gained about a test device from a transformed S21 transmission measurement. The frequency components pass through the test device and are measured at the B receiver. If there is more than one path through the device, they would appear as various pulses separated in time.

For example, the following transmission measurement shows multiple paths of travel within a Surface Acoustic Wave (SAW) filter. The largest pulse (close to zero time) represents the propagation time of the shortest path through the device. It may not be the largest pulse or represent the desired path. Each subsequent pulse represents another possible path from input to output.

![Transmission Measurement Diagram](image)

Triple travel is a term used to describe the reflected signal off the output, reflected again off the input, then finally reappearing at the output. This is best seen in a time domain S21 measurement.

Measurement Response Resolution

In the previous paragraphs, we have seen that using more frequency components causes the assembled
waveform to show more detail. This is known as measurement response resolution, which is defined as the ability to distinguish between two closely spaced responses.

**Note:** Adjusting the *transform time settings* improves *display* resolution, but not measurement resolution.

The following graphic shows the effect of both a narrow and wide frequency span on the response resolution. The wider frequency span enables the analyzer to resolve the two connectors into separate, distinct responses.

**Resolution Formula**

For responses of equal amplitude, the response resolution is equal to the 50% (-6 dB) points of the impulse width, or the step rise time which is defined as the 10 to 90% points as shown in the following image.

The following table shows the *approximated* relationship between the frequency span and the window selection on response resolution for responses of equal amplitude.
<table>
<thead>
<tr>
<th>Window</th>
<th>Low-pass step (10% to 90%)</th>
<th>Low-pass impulse (50%)</th>
<th>Bandpass impulse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum</td>
<td>0.45 / f span</td>
<td>0.60 / f span</td>
<td>1.20 / f span</td>
</tr>
<tr>
<td>Normal</td>
<td>0.99 / f span</td>
<td>0.98 / f span</td>
<td>1.95 / f span</td>
</tr>
<tr>
<td>Maximum</td>
<td>1.48 / f span</td>
<td>1.39 / f span</td>
<td>2.77 / f span</td>
</tr>
</tbody>
</table>

For example, using a 10 GHz wide frequency span and a normal window in Bandpass impulse mode, response resolution (in time) equals:

- Time Res = 1.95 / frequency span
- Time Res = 1.95 / 10 GHz
- Time Res = 195 ps

To calculate the physical separation (in distance) of the responses which can be resolved, multiply this value times the speed of light (c) and the relative velocity (Vf) of propagation in the actual transmission medium. In this case, Vf = 0.66 for polyethylene dielectric.

- Distance Res = 195 ps x c x Vf
- Distance Res = 195 ps x (2.997925 E8 m/s) x .66
- Distance Res = 38 mm

For reflection measurements, because of the 2-way travel time involved, this means that the minimum resolvable separation between discontinuities is half of this value or 19 mm.

Although a wider frequency span causes better measurement resolution, the measurement range becomes limited. Also, increasing the frequency range can cause a measurement calibration to become invalid. Be sure to adjust the frequency span BEFORE performing a calibration.

**Measurement Range and Alias Responses**

Measurement range is the length in time in which true time domain responses can be seen. The measurement range should be large enough to see the entire test device response without encountering a repetition (alias) of the response. An alias response can hide a true time domain response.

To increase measurement range in both modes, change either of these settings:

- Increase the number of points
• Decrease the frequency span

Notes:

• After making these settings, you may need to adjust the transform time settings to see the new measurement range.
• Decreasing the frequency span degrades measurement resolution.
• Make frequency span and number of points settings BEFORE calibrating.
• Maximum range also depends on loss through the test device. If the returning signal is too small to measure, the range is limited regardless of the frequency span.

Alias Responses

An alias response is not a true device response. An alias response repeats because each time domain waveform has many periods and repeats with time (see How the Analyzer Measures in the Time Domain). Alias responses occur at time intervals that are equal to 1/frequency step size.

The analyzer adjusts the transform time settings so that you should only see one alias free range on either side (positive and negative) of zero time. However, these settings are updated only when one of the toolbar settings are changed.

To determine if a response is true, put a marker on the response and change the frequency span. A true device response will not move in time. An alias response will move.

For example, in the above graphic, the marker 1 response occurs at 14.07 inches. When the frequency span is changed, this response remains at 14.07 inches. The marker 2 response moves.

Range Formula

You can calculate the alias-free measurement range (in meters) of the analyzer using the following
formula for TDR (reflection) measurements:

**Range (meters) = \( (1/\Delta f) \times V_f \times c \)**

Where:

- \( \Delta f \) = frequency step size (frequency span/number of points-1)
- \( V_f \) = the velocity factor in the transmission line
- \( c \) = speed of light = 2.997925 E8 m/s

For example: For a measurement with 401 points and a span of 2.5 GHz, using a polyethylene cable (\( V_f = 0.66 \))

- Range = \( (1 / (2.5E9 / 400)) \times 2.997925 \times 0.66 \)
- Range = 6.25E6 \times 2.997925 \times 0.66
- Range = 32 meters

In this example, the range is 32 meters in physical length. To prevent the time domain responses from overlapping or aliasing, the test device must be 32 meters or less in physical length for a transmission measurement.

To calculate the one-way distance for a reflection measurement rather than round-trip distance, simply divide the length by 2. In this case, the alias-free range would be 16 meters.

### How to make Time Domain Settings

**Using Hardkey/SoftTab/Softkey**

1. Press **Math > Time Domain > Time Domain Setup....**

### Transform dialog box help
Transform On  Turns time domain transform ON and OFF.

Time Settings

The following settings adjust the display resolution, allowing you to zoom IN or OUT on a response. They do NOT adjust measurement range or measurement resolution.

These settings automatically update (when one of these values are updated) to limit the display to one alias-free response on either side of zero time.

Start  Sets the transform start time that is displayed on the analyzer screen.

Note: Zero (0) seconds is always the measurement reference plane. Negative values are useful if moving the reference plane.

Stop  Sets the transform stop time that is displayed on the analyzer screen.

Center  Sets the transform center time that is displayed in the center of the analyzer screen.

Span  Sets the transform span time that is split on either side of the Center value.

Transform Mode

Transform modes are three variations on how the time domain transform algorithm is applied to the frequency domain measurement. Each method has a unique application.
<table>
<thead>
<tr>
<th>Mode</th>
<th>Benefit - application</th>
<th>Limitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low pass Impulse</td>
<td>Highest resolution.</td>
<td>In both Low pass modes, frequencies down to DC and negative frequencies are extrapolated.</td>
</tr>
<tr>
<td></td>
<td>Most useful for seeing small responses in devices that pass low frequencies,</td>
<td>Therefore, the Start frequency is adjusted when you click Set Freq.Low Pass</td>
</tr>
<tr>
<td></td>
<td>such as cables.</td>
<td>Because this will affect calibration accuracy, be sure to calibrate AFTER completely setting up</td>
</tr>
<tr>
<td></td>
<td></td>
<td>your time domain measurement.</td>
</tr>
<tr>
<td>Low pass Step</td>
<td>Easiest to identify inductive and capacitive discontinuities in devices that pass</td>
<td></td>
</tr>
<tr>
<td></td>
<td>low frequencies, such as cables.</td>
<td></td>
</tr>
<tr>
<td>Band pass Impulse</td>
<td>Easiest method - can be used with any frequency sweep.</td>
<td>Does NOT show capacitive and inductive reactance</td>
</tr>
<tr>
<td></td>
<td>Most useful for measuring band limited devices such as filters and DC blocked cables.</td>
<td>For the same frequency span and number of points, band pass mode has twice the impulse width,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>which hides closely spaced responses degrading the response resolution.</td>
</tr>
</tbody>
</table>

The following chart shows how to interpret results from various discontinuity impedances using Low pass Step and either Low pass or Band pass Impulse modes.

<table>
<thead>
<tr>
<th>IMPEDANCE</th>
<th>STEP RESPONSE</th>
<th>IMPULSE RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPEN</td>
<td>Unity Reflection</td>
<td>Unity Reflection</td>
</tr>
<tr>
<td>SHORT</td>
<td>Unity Reflection = 180</td>
<td>Unity Reflection = 180</td>
</tr>
<tr>
<td>RESISTOR</td>
<td>Unity Reflection</td>
<td></td>
</tr>
<tr>
<td>R &gt; Z₀</td>
<td>Unity Reflection</td>
<td></td>
</tr>
<tr>
<td>R &lt; Z₀</td>
<td>Unity Reflection</td>
<td></td>
</tr>
<tr>
<td>INDUCTOR</td>
<td>Unity Reflection</td>
<td></td>
</tr>
<tr>
<td>CAPACITOR</td>
<td>Unity Reflection</td>
<td></td>
</tr>
</tbody>
</table>

**Effect on Measurement Range**

**Band pass mode** - measurement range is inversely proportional to frequency step size.
**Low pass mode** - measurement range is inversely proportional to the fundamental (start) frequency AFTER clicking Set Freq. Low Pass.

**Set Low Pass Frequencies** USE ONLY IN LOW PASS MODES

Recomputes the start frequency and step frequencies to be harmonics of the start frequency. Start frequency is computed by the following formula: **Low Pass Start Frequency = Stop Frequency / Number of points.**

The computed value must always be greater than or equal to the analyzer's minimum frequency.

**Note:** The number of points or stop frequency may be changed in order to compute this value.

**Gating**

Perhaps the most beneficial feature of time domain transform is the Gating function. When viewing the time domain response of a device, the gating function can be used to "virtually" remove undesired responses. You can then simultaneously view a frequency domain trace as if the undesired response did not exist. This allows you to characterize devices without the effects of external devices such as connectors or adapters.

**Note:** When a discontinuity in a test device reflects energy, that energy will not reach subsequent discontinuities. This can "**MASK**", or hide, the true response which would have occurred if the previous discontinuity were not present. The analyzer Gating feature does NOT compensate for this.

The following measurements images show a practical example how to use and perform gating. The test device is a 10 inch cable, then a 6 dB attenuator, terminated with a short. The following four discontinuities are evident in window 2, from left to right:

1. A discontinuity in the test system cable which appeared after calibration. It is identified by marker 2 at -10.74 inches (behind the reference plane).

2. A discontinuity in the 10 inch device cable shortly after the reference plane.

3. The largest discontinuity is the attenuator and short shown by marker 1 at -12.67 dB (6 dB loss in both forward and reverse direction).

4. The last discontinuity is a re-reflection from the device cable.

We will gate IN the attenuator response. All other responses will be gated OUT.

**Window 1.** Create original S11 frequency domain trace. Shows ripple from all of the reflections.
Window 2. Create a new S11 trace - same channel; new window. Turn Transform ON.

Window 3. On the transformed trace, turn gating ON. Center the gate on the large discontinuity (2.500ns). Adjust gate span to completely cover the discontinuity. Select Bandpass gating type.

Window 4. On the original frequency measurement, turn Gating ON (Transform remains OFF). View the measurement without the effects of the two unwanted discontinuities. The blue trace is a measurement of the 6 dB attenuator with the unwanted discontinuities PHYSICALLY removed. The difference between the two traces in window 4 is the effect of "masking".

Learn how to launch the Transform dialog box

Programming Commands

Transform Gating dialog box help
Gating  Turns Gating ON and OFF.

Start  Specifies the start time for the gate.

Stop  Specifies the stop time for the gate.

Center  Specifies the value at the center of the area that is affected by the gating function. This value can be anywhere in the analyzer range.

Span  Specifies the range to either side of the center value of area that is affected by the gating function.

Gate Type  Defines the type of filtering that will be performed for the gating function. The gate start and stop flags on the display point toward the part of the trace you want to keep.

- **Bandpass** - KEEPS the responses within the gate span.
- **Notch** - REMOVES the responses with the gate span.

Gate Shape  Defines the filter characteristics of the gate function. Choose from Minimum, Normal, Wide, Maximum
<table>
<thead>
<tr>
<th>Gate Shape</th>
<th>Passband Ripple</th>
<th>Sidelobe Levels</th>
<th>Cutoff Time</th>
<th>Minimum Gate Span</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum</td>
<td>±0.1 dB</td>
<td>-48 dB</td>
<td>1.4/Freq Span</td>
<td>2.8/Freq Span</td>
</tr>
<tr>
<td>Normal</td>
<td>±0.1 dB</td>
<td>-68 dB</td>
<td>2.8/Freq Span</td>
<td>5.6/Freq Span</td>
</tr>
<tr>
<td>Wide</td>
<td>±0.1 dB</td>
<td>-57 dB</td>
<td>4.4/Freq Span</td>
<td>8.8/Freq Span</td>
</tr>
<tr>
<td>Maximum</td>
<td>±0.01 dB</td>
<td>-70 dB</td>
<td>12.7/Freq Span</td>
<td>25.4/Freq Span</td>
</tr>
</tbody>
</table>

**Cutoff time** -- is the time between the stop time (-6 dB on the filter skirt) and the peak of the first sidelobe. The diagram below shows the overall gate shape and lists the characteristics for each gate shape.

- T₁ is the gate span, which is equal to the stop time minus the start time.
- T₂ is the time between the edge of the passband and the 6 dB point, representing the cutoff rate of the filter.
- T₃ is the time between the 6 dB point and the edge of the gate stopband.
- For all filter shapes T₂ is equal to T₃, and the filter is the same on both sides of the center time.

**Minimum gate span** -- is twice the cutoff time. Each gate shape has a minimum recommended gate span for proper operation. This is a consequence of the finite cutoff rate of the gate. If you specify a gate span that is smaller than the minimum span, the response will show the following effects:

- distorted gate shape that has no passband
- distorted shape
- incorrect indications of start and stop times
- may have increased sidelobe levels

To launch the Coupling dialog box, click Coupling tab on the Time Domain Setup dialog box.

**Programming Commands**

**Trace Coupling Settings** dialog box help
Trace coupling allows you to change time domain parameters on a measurement, and have the same changes occur for all other measurements in the channel.

For example:

If you are simultaneously viewing a frequency domain measurement and time domain measurement,

and **Coupling** is enabled in this dialog box,

and ALL **Gating Parameters** are checked in this dialog box,

and on the time domain measurement you change the **Gate Span** parameter,

Then the frequency domain measurement will automatically change to reflect the time domain gated span.

**Note:** Trace coupling applies ONLY to the Y-axis scale/reference settings. There are no changes to your data as a result of trace coupling.

**Coupling On** Check to enable coupling. All of the measurements in the active channel are coupled.

The following parameters are available for coupling:

**Transform Parameters**

- **Stimulus** Start, Stop, Center, and Span TIME settings.
- **State** (On/Off) Transform ON and OFF
Window  Kaiser Beta / Impulse Width

Mode  Low Pass Impulse, Low Pass Step, Band Pass

Gating Parameters

Stimulus  Start, Stop, Center, and Span TIME settings.

State  (On/Off)  Gating ON and OFF

Shape  Minimum, Normal, Wide, and Maximum

Type  Bandpass and Notch

To launch the Distance marker dialog box, click Dist. Marker Settings on the Transform dialog box.

Programming Commands

Distance Marker Settings  dialog box help

When markers are present on a time domain measurement, distance is automatically displayed on the marker readout, marker table, and print copy. To learn how to create markers on your measurement see marker settings.

You can read out impedance versus time by creating a marker on a Time Domain trace, then changing the marker format to R+jX. Learn how.
This dialog box allows you to customize the time domain distance marker readings.

These settings affect the display of ALL markers for only the ACTIVE measurement (unless **Distance Marker Unit** is coupled on the Trace Coupling dialog box.

**Marker Mode**  Specifies the measurement type in order to determine the correct marker distance.

- Select **Auto** for S-Parameter measurements.
- Select **Reflection** or **Transmission** for arbitrary ratio or unratioed measurements.

**Auto**  If the active measurement is an S-Parameter, automatically chooses reflection or transmission. If the active measurement is a non S-Parameter, reflection is chosen.

**Reflection**  Displays the distance from the source to the receiver and back divided by two (to compensate for the return trip.)

**Transmission**  Displays the distance from the source to the receiver.

**Units**  Specifies the unit of measure for the display of marker distance values.

**Velocity Factor**  Specifies the velocity factor that applies to the medium of the device that was inserted after the measurement calibration. The value for a polyethylene dielectric cable is 0.66 and 0.7 for PTFE dielectric. 1.0 corresponds to the speed of light in a vacuum. This is useful in Time Domain for accurate display of time and distance markers.

This setting can also be made from the Electrical Delay and Port Extensions dialog boxes.

---

**Advanced Settings**

**Programming Commands**
The following methods set window size. For best results, view the time domain response while making these settings.

- **Kaiser Beta** Changes window size using a Kaiser Beta value
- **Impulse Width** Changes window size using an Impulse Width value

**Transform Alignment**

**Alignment Type**

- **Legacy** - The DC value is extrapolated using three data points. The transform offset is calculated using the delay of the first frequency point. This is the same algorithm used in the HP 8510 network analyzer.

- **Normalize** - The DC value is extrapolated using three data points. The transform offset is set to zero at t=0 minus six rise-times. This mode requires that a good S-parameter calibration has been performed, which can be verified by observing a flat time-domain response at t=0 when measuring a load located at the physical point corresponding to t=0. Setting the time domain trace to zero at a time before t=0 stabilizes the trace for determining impedances after time t=0, resulting in improved behavior compared to Legacy mode. This method is similar to that used with PLTS, and is very useful in determining the time-domain-transform response of transmission lines and printed-circuit-board characteristics.

**Limit Start/Stop Times to Avoid Aliasing** - Check to limit the start/stop times to avoid aliasing.
How to launch the Time Domain Toolbar

Using Hardkey/SoftTab/Softkey

1. Math > Time Domain > TD Toolbar

On the toolbar, click More... to launch the Time Domain dialog box
Enhanced Time Domain Analysis (Option S9x011A/B)

- Overview
- Setting Up Measurement (Setup Tab)
- Making Measurements
- Eye Diagram and Mask Test (Eye/Mask Tab)
- Storing Data and Setting
- Advanced Waveform Analysis
- Advanced Mode
- Measurement Examples

- TDR Programming Commands:
  - CALCulate:TDR Commands
  - DISPlay:TDR Commands
  - MMEMory:TDR Commands
  - SENSE:CORRection:TDR Commands
  - SENSE:TDR Commands
  - SOURce:TDR Commands
  - SYSTem:TDR Commands

- TDR Programming Examples:
  - TDR/TDT Measurement
  - Simulated Eye Diagram
  - 2 Channel Measurement
- Command Finder
Overview

- Features
- TDR Screen Area
- TDR Measurement Considerations
- Starting and Exiting TDR Application

Other topics about Enhanced Time Domain Analysis
Features and Limitations

Features

The TDR option (S9x011A/B) provides the following features:

- Up to 67 GHz of bandwidth with 6.66 ps (10%-90%) or 4.73 ps (20%-80%) rise time enables measurement on the latest high speed serial standards
- Wide dynamic range to observe the true performance of the DUT
- Low noise floor for accurate and repeatable measurements
- Fast measurement speed for real-time analysis
- State-of-the-art calibration techniques reduce measurement errors
- Automatic deskew ensures easy removal of fixture and probe effects
- Full calibration available for the utmost in measurement accuracy
- Quickly obtain accurate TDR/TDT and S-parameter measurements
- Easily locate source of loss, reflections and crosstalk by simultaneous analysis of both time and frequency domain
- Single connection forward and reverse transmission and reflection measurements
- All possible modes of operation (single-ended, differential and mode conversion)
- Measure just the device by utilizing advanced calibration techniques to remove cable, fixtures and probe effects
- Gain insight into high speed interconnect performance through simulated eye diagram analysis and manual scale of eye diagram
- Apply industry standard (PRBS, K28.5) or used specified patterns using the virtual bit pattern generator
- Pre-defined masks for many high speed serial standards
- No need for pulse generators as the eye diagram is synthesized from measurement results
- Hot TDR measurement which allows TDR measurement to be performed while the device is powered ON
Limitations

TDR is not supported in the following configurations:

- If LFE (Low Frequency Extension) is ON, launching TDR will force LFE mode OFF.
- Millimeter Wave configurations do not support TDR either remotely or from the GUI even if TDR is licensed.
- Multiport mode does not support TDR either remotely or from the GUI even if TDR is licensed.

Other topics about Overview
TDR Quick Start

TDR Quick Start helps you to understand the TDR option's operation quickly.

- TDR/TDT Measurement
- Simulated Eye Diagram
TDR/TDT Measurement

To better understand how to use the TDR option, this section describes the basic TDR/TDT measurement procedure using the VNA.

Operation procedure

1. Connect cables to all test ports.
2. Click the **Setup** tab.
3. Click the **Setup Wizard** (under Basic).
4. Set the measurement condition using the **Setup Wizard**:
   a. Select **Deskew & Loss Compensation** (under **Error Correction**) then click **Next >**.
   b. Click the **Differential 2-Port** button, then click **Next >**.
   c. Click the **Deskew** button, then click **Next >**.
   d. Connect a Thru between Port 1 and Port 3. Click the **Measure** button, and click **Next >**.
   e. Connect a Thru between Port 2 and Port 4. Click the **Measure** button, and click **Next >**.
   f. Connect a Load to Port 1 and click the Port 1 button to measure the Load. Repeat this step for Port 2, Port 3 and Port 4. When finished, click **Apply** to save the measurement.
   g. Click **Next >**.
   h. Connect the DUT to cables. Click the **Measure** button, then click **Next >**.
   h. Set the Rise Time to "35 ps" and select "10-90%" from the **Definition** drop-down list. When finish, click **Apply** to save the settings.
   i. Click the **Finish** button.
5. Click the **Auto Scale** then **All Traces** from the drop-down list.
6. Select Trace 3.
7. Click the **Marker Search** button.
8. Select **Rise Time (10-90%)** from the drop-down list.

**Expected result**

Trace 1 shows the TDR measurement in terms of impedance and Trace 3 shows TDT measurement in terms of voltage. The rise time of Trace 3 will be shown.
Simulated Eye Diagram

This section describes the procedure of simulated eye diagram using the VNA with TDR option.

Operation procedure

1. Connect cables to all test ports.
2. Click the Setup tab.
3. Click the Setup Wizard (under Basic).
4. Set the measurement condition using the Setup Wizard:
   a. Select Deskew & Loss Compensation (under Error Correction) then click Next >.
   b. Click the Differential 2-Port button, then click Next >.
   c. Click the Deskew button, then click Next >.
   d. Connect the DUT to cables. Click the Measure button, then click Next >.
   e. Set the Rise Time to 35 ps and select 10-90% from the Definition drop-down list. When finished, click Apply to save the settings.
   f. Click the Finish button.
5. Click Trace button on the tool bar and select 3 to activate Trace 3.
6. Click Eye/Mask tab.
7. Under Stimulus:
   a. Select PRBS from the Type drop-down list.
   b. Select 2^7-1 bits from the Length drop-down list.
   c. Set the One Lv. (amplitude) to 200 mV.
   d. Set the Data Rate to 1 Gb/s.
8. Click the Draw Eye button under Trigger.

Expected result

The eye diagram (Trace 3) similar to the one shown below is displayed.
Other topics about TDR Quick Start
TDR Screen Area

TDR screen area appears at the bottom of the standard screen.

Graph Area

The results of measurements such as Time Domain, S-Parameter, and Eye are displayed here. Operation with mouse is accepted.

Setup Wizard

Using Setup Wizard

Instrument Status Bar

Instrument Status Bar

Channel Window

In addition to the standard status bar, there is a TDR indicator at the right corner of the channel window. This indicator is turned ON only when the TDR option is turned ON. The TDR indicator is blue in basic mode and changes to yellow when advanced mode is turned ON. This indicator also changes according to the Cal status, as shown in the following table:
Cal | Indicator
--- | ---
Off | TDR
Deskew | TDR [Deskew]
DLC | TDR [DLC]
ECal | TDR [Full]
ECal&Deskew | TDR [Full+]

<table>
<thead>
<tr>
<th>Others</th>
<th>Indicator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hot TDR Mode</td>
<td>TDR ?</td>
</tr>
</tbody>
</table>

**Tool Bar**

Frequently used standard functions are displayed here. These functions are synchronized with TDR options and modes.

<table>
<thead>
<tr>
<th>No.</th>
<th>Functions</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Active Trace</td>
<td>Setting up Parameters on Each Traces</td>
</tr>
<tr>
<td>2</td>
<td>Auto Scale</td>
<td>Using Scale/Zooming</td>
</tr>
<tr>
<td>3</td>
<td>Run</td>
<td>Controlling Trigger</td>
</tr>
<tr>
<td>4</td>
<td>Stop/Single</td>
<td>Controlling Trigger</td>
</tr>
<tr>
<td>5</td>
<td>Data Mem</td>
<td>Using Data/Memory</td>
</tr>
<tr>
<td>6</td>
<td>Marker</td>
<td>Using Marker/Marker Search</td>
</tr>
<tr>
<td>7</td>
<td>Marker Search</td>
<td>Using Marker/Marker Search</td>
</tr>
<tr>
<td>8</td>
<td>File</td>
<td>Storing Data and Setting</td>
</tr>
<tr>
<td>9</td>
<td>Minimize</td>
<td>Minimizes the Setting Area. Only tool bar is displayed.</td>
</tr>
<tr>
<td>10</td>
<td>Help</td>
<td>Executes TDR Online Help</td>
</tr>
<tr>
<td>11</td>
<td>Exit</td>
<td>Exiting TDR application GUI</td>
</tr>
</tbody>
</table>

**Note:** Buttons 1 to 5 and 9 to 12 are always displayed regardless of the selected mode. Buttons 6 to 8
do not appear in the Eye mode, and the measurement parameter is displayed on the surface of Button 3 (instead of Auto Scale).

Mode Selector

You can select one of the three available TDR modes:

- Setup
- TDR/TDT
- Eye Mask

The mode changes when one of these tabs are selected. Once selected, the mode is highlighted. The displayed setting area changes corresponding to the selected mode.

Setting Area

The setting area changes and is displayed according to the selected mode.

Other topics about Overview
TDR Measurement Considerations

- Channel and Trace Concepts
- Device Under Test (DUT) Considerations
- Test Cable and Connector Considerations

Channel and Trace Concepts

It is important for oscilloscope users to note the difference in the concept of channel in oscilloscope and network analyzer.

In network analyzer, channel is referred to the window for displaying traces. Because a channel corresponds to a window, it is called a channel window.

On the other hand, the points on the front panel of the network analyzer where cables and DUTs are connected are called test ports. These test ports are not channels and do not directly associate with channel windows.

The VNA, for example allows you to use up to 150 channels to perform measurements under 150 different stimulus conditions. All traces that are assigned to a channel share the same channel settings.

To learn more about setting the channel and traces, refer to Traces, Channels, Windows, and Sheets.

Enhanced Time Domain Analysis option allows you to use channel 1 only. Up to 16 traces can be displayed in this option (when Differential 2-Port DUT topology and all T or all S traces are selected).

Device Under Test (DUT) Considerations

Maximum DUT Lengths

To convert from DUT length in seconds to distance in free space, multiply the value in time by c, the speed of light in free space. To calculate the actual physical length, multiply this value in free space by Vf, the relative velocity of propagation in the transmission medium. (Most cables have a relative velocity of 0.66 for a polyethylene dielectric or 0.7 for a PTFE dielectric.)

\[ \text{DUT length (m)} = \text{DUT length (s)} \times c \times Vf \]

AC coupled (DC cut) DUTs
Caution is required when measuring AC coupled (DC cut) DUTs.

The response after the capacitor cannot be measured correctly, because the capacitor response overlaps the response from the input and may cause measurement errors.

When measuring AC coupled devices, make sure that the following two conditions are satisfied for proper measurements.

- Capacitance should be equal to or greater than 10 nF
- DUT length should be equal to or less than 6.25 nsec (1.25 meters at $\varepsilon=2.25$)

DUT length is the actual length of the DUT, not DUT Length in the Setup tab in the TDR GUI.

When the above conditions are not satisfied, either:

- Short circuit the capacitor
- Provide open termination at one end of the capacitor and measure from the opposite side.

**Test Cable and Connector Considerations**

Using high quality cables to connect the DUT is recommended in order to minimize measurement degradation. The cables should have low loss, low reflections, and minimum performance variation when flexed.

**Note:** When performing deskew and loss compensation, the same type of cable must be used for all test ports, but they do not necessarily have to be of the same length.

When performing deskew or deskew and loss compensation, use a single connector interface type, such as SMA, or Type-N, for all of the ports to obtain the best results.
Starting andExiting TDR Application

- Starting TDR Application
-Exiting TDR Application

Taking a look at Other topics about Overview

Starting TDR Application

1. On the VNA, press Meas > S-Param > Meas Class....
2. Select TDR, then click OK.
3. In the Confirm Measurement Class Change dialog, click OK to proceed or Cancel to exit.
4. The TDR application is displayed.

Starting TDR application remotely

- SYST:TDR:PRESet launches TDR application without VNA-TDR GUI.

Exiting TDR Application

There are two ways to exit the TDR application.

1. Select a different Measurement Class for the currently active channel.
2. Select the Exit button in the TDR Setup dialog.

Exiting TDR application remotely
- **SYST:PRESet** quits TDR application (VNA-TDR GUI is closed if exists.)
Setting Up the Measurement

- Using Setup Wizard
- Performing Manual Setup
- Performing Error Corrections

Other topics about Enhanced Time Domain Analysis
Setup Wizard guides you to perform the Enhanced Time Domain Analysis measurement setup step-by-step. The wizard appears automatically the first time you execute the Enhanced Time Domain Analysis option. If you check the Show this wizard next start-up option, the setup wizard will appear automatically in the next start-up as well.

Another option to start up the Setup Wizard is from the Setup tab:

1. Click the **Setup** tab.

2. Click the **Setup Wizard** button under **Basic**, as shown below.

3. If you do not want to perform the Enhanced Time Domain Analysis setup by using the wizard, click **Close** to close the **Setup Wizard**.

4. Otherwise, select the type of **Error Correction** among the available options:
   - Deskew
5. If you check the **Show this wizard next start-up** option, the setup wizard will appear automatically in the next start-up.

6. Click **Next >** to start the setup process.

7. Select the topology of the device under test (DUT).

8. Click **Next >**.

9. The next screen of the wizard varies depending on the selection of **Error Correction** option and topology of the DUT.

10. Follow the instructions on the wizard to complete the setup.

**Other topics about Setting Up the Measurement**
Performing Manual Setup

Besides Using the Setup Wizard, you can also perform the Enhanced Time Domain Analysis measurement setup manually. The manual setup is performed at the Setup tab.

- Preset
- Setting DUT Topology
- Setting Stimulus Amplitude Level
- Setting DUT Length
- Performing Error Correction
- More Functions
- Average
- Advanced Waveform
- Hot TDR

Other topics about Setting Up the Measurement

Preset

1. Click **Preset** under **Basic** to preset the VNA.
2. A dialog box appears requesting confirmation. Click **OK** to proceed.
3. All the settings shown in the **Basic** area are changed to default except for the DUT Topology.
4. When you click **Preset**, the calibration and deskew data is deleted. Preset sets all of the settings at default except for the DUT Type.

Setting DUT Topology
1. At the **DUT Topology** under **Basic**, select one of the available options from the drop-down list box. This is the same function as Step 1/4: DUT Topology in the **Setup Wizard**.

2. A dialog box appears requesting confirmation. Click **OK** to proceed.

Selecting the DUT topology executes the preset of the VNA. Therefore, when you change from one DUT topology to another, the calibration and deskew data is deleted.

### Setting Stimulus Amplitude Level

**Note:** Stimulus Amplitude Level is the value for scaling the time domain measurement result.

1. Stimulus Amplitude Level is not related to the actual applied voltage.

2. Click in the text box of **Stim. Ampl.** (Stimulus Amplitude) under **Basic**. An **Entry** dialog box appears. Type the stimulus amplitude level value then click **OK**. The new value is displayed at **Stim. Ampl.** text box.

### Setting DUT Length

The DUT length setting is used to set the time span for time domain measurements. Measurements can be made on longer DUTs, but minimum rise time values may be limited.

**Automatic measurement** - recommended
1. Click **Auto** next to **DUT Length** under **Basic**. The **DUT Length** dialog box of the **Setup Wizard** appears.

2. Click **Measure**.

3. The length of the DUT is measured automatically and used to set the time base.

4. Once complete, a check mark appears beside the **Measure** button.

The automatic measurement feature is available as one of the steps in the **Setup Wizard**.

**Manual entry**

If the length of the DUT is known, the DUT length can be set manually.

1. Enter the DUT length in the **DUT Length** text box under **Basic**.

Any DUT shorter than the DUT length setting can be measured. Therefore, when testing multiple DUTs with different lengths, set the DUT length using the longest DUT to allow for the use of the
same instrument settings for all measurements.

**Performing Error Correction**

There are three error correction options available, as stated below. Refer to Performing Error Correction for more information.

- Deskew
- Deskew and Loss Compensation
- Full Calibration (ECal)

**More Functions**

**Ref. Z**

1. At the Ref. Z (Port Reference Impedance) text box in the More Functions tab, left-click once. An Entry dialog box appears. Type the post reference impedance value then click OK. The new value is displayed in the Ref. Z text box.

**Dielectric Const. and Velocity Factor**

Velocity Factor = 1 / sqrt(Dielectric Constant).

As such, when you change either one, the value of the other changes automatically.

1. To change the value of the dielectric constant, at the Dielectric Const. text box in the More Functions tab, left-click once. An Entry dialog box appears. Type the Dielectric Constant value then click OK. The new value is displayed in the Dielectric Const. text box.

2. The value of the velocity factor is changed in a similar way at Velocity Factor text box in the More Functions tab.

**Source Power**

The signal source level is changed during HOT TDR measurement to avoid device malfunction:

1. At the Source Power text box in the More Functions tab, left-click once. An Entry dialog box appears. Type the source power value then click OK. The new value is displayed in the Source Power text box.

**Bal Port Config**

This button opens the balanced source setting dialog.
The default port allocation is different between PNA/ENA VNAs and PXI/USB VNAs. In PXI/USB VNAs, the left hand DUT ports are odd and the right hand DUT ports are even.

![Port Allocation Diagram]

**Freq Limits Config**

The Enhanced Time Domain Analysis measurement start and stop frequencies can be specified directly using the **Frequency Limits** dialog. This feature allows the Enhanced Time Domain Analysis measurement frequency range to be set narrower than the hardware's frequency range. For example, when using an ECal with a narrower frequency range than the hardware.

1. Click on the **Freq Limits Config** button. The **Frequency Limits** dialog is displayed.

![Frequency Limits Dialog]

2. Specify the start frequency in the **Start Limit** field.
3. Specify the stop frequency in the **Stop Limit** field.
4. Click the **Restore Defaults** button to set the start/stop limits back to the default values.
5. Clicking the **OK** button will apply changes and close the dialog. This causes a TDR preset.
6. Clicking the **Cancel** button will cancel any changes and close the dialog.

**Advanced Mode**

Refer to Advanced Mode.

**Average**
The averaging function allows you to reduce the trace noise. It executes the sweep the number of times specified by the averaging factor when the sweep averaging function is turned ON. To activate the averaging option, go to Average.

1. To turn ON the averaging factor, select the **Averaging** check box in the **Average** tab.

2. Type the averaging factor (number of times the sweep needs to be executed).

3. To turn ON the averaging trigger, select the **Average Trigger** check box.

4. At the **IF Bandwidth** text box, left-click once. An **Entry** dialog box appears. Type the IF bandwidth value then click **OK**. The new value is displayed in the **IF Bandwidth** text box. Reducing IF bandwidth increases the dynamic range.

**Advanced Waveform**

Refer to Advanced Waveform Analysis.

**Hot TDR**

Refer to HOT TDR measurement.
Performing Error Corrections

- Overview
- Deskew
- Deskew and Loss Compensation
- Full Calibration (ECal)

Other topics about Setting Up Measurement

Overview

There are many different approaches of removing the effects of the test fixture and cables from the measurement. The level of difficulty for each error correction technique is related to the accuracy of each method. As such, TDR gives you the flexibility of selecting the desired error correction.

There are three error correction options available, as stated below:

- Deskew
- Deskew and Loss Compensation
- Full Calibration (ECal)

If you use the Setup Wizard, you will be guided through step-by-step to perform the Enhanced Time Domain Analysis measurement setup. This includes performing the error correction. Optionally, you can perform the error correction manually using the Setup tab.

Note: The TDR indicator at the channel window also shows the TDR deskew and calibration status.

Deskew
Deskew automatically compensates the electrical length of the cables and fixtures. Deskew mathematically extends the calibration reference plane to the DUT, hence removes the delay from the test setup effectively. This error correction technique provides good results if the cable and fixture are very well designed.

As deskew does not compensate the loss of cables in the measurement, cables should be low loss and as reasonably short as possible.

1. Click the **Deskew** button under **Basic**.
2. The **Deskew** dialog box of the Setup Wizard appears.
3. Follow the instructions on the wizard and click the **Deskew** button in the dialog box.
4. Deskew is performed and a check mark appears beside the **Deskew** button.
5. Alternatively, you can deskew one port at a time. To perform this, instead of clicking the **Deskew** button, click the **Options** button in the dialog box.
6. At the **Deskew Options** dialog box, you can click one port at a time to deskew it.
8. Click **Measure All** to perform deskew on all the ports. This action is the same as clicking the **Deskew** button. You must deskew all the active ports.

9. At **Standard Type**, select either **Open** or **Short** deskew.

10. Click **OK** to close the **Deskew Options** dialog box.

11. Once complete, click **Finish** on the dialog box.

**Deskew and Loss Compensation**

Deskew and loss compensation mathematically extends the calibration reference plane to the DUT, hence removes the delay and loss from the test setup effectively. This error correction technique is a good compromise between level of difficulty and accuracy.

The same type of cable must be used for all test ports. However, they do not necessarily have to be of the same length.

1. Click the **Deskew & Loss** button under **Basic**.

2. The **Deskew & Loss Compensation** dialog box of the **Setup Wizard** appears.

3. The Deskew & Loss Compensation is a three-step process.

4. Follow the instructions on the wizard and click the **Deskew** button in the dialog box.
5. Alternatively, you can deskew one port at a time, just as in Deskew. To perform this, instead of clicking the Deskew button, click the Options button in the dialog box.

6. At the Deskew Options dialog box, you can click one port at a time to deskew it.

7. Clicking Measure All performs deskew on all the ports at the same time. This action is the same as clicking the Deskew button. You must deskew all the active ports.

8. At Standard Type, select either Open or Short deskew.

9. Click OK to close the Deskew Options dialog box.

10. At Step 2, connect a thru between the ports as per the instruction on the dialog box then click Measure.

   - Use thru with a short and low loss.

11. At Step 3, connect a load to the ports one by one then click the associate button accordingly. Once complete, click Apply to save the measurement.

12. Click Finish on the dialog box.
**Full Calibration (ECal)**

Full Calibration refers to calibration using the Electronic Calibration (ECal) module. It is a complete solid-state calibration solution, which makes calibration fast and easy.

**ECAL Minimum Frequency Check**

The performance of ECal modules with a start frequency of 10 MHz affects time domain accuracy. The firmware will issue the following warning:

The selected ECal does not have optimum low frequency performance. For higher accuracy, use either the DC version of the ECal, or perform mechanical calibration in Advanced Mode.

Also, some ECal modules have a low maximum power limit (-15 dBm) that would degrade time domain performance. Hence, these ECal modules are not recommended for calibration. In this case, the firmware will issue the following warning:

The selected ECal does not have optimum performance. For higher accuracy, use either the DC version of the N4690 Series ECal, or perform mechanical calibration.

**Note:** These warning messages are TDR mode only.

1. Click the ECal button under Basic.
2. The Full Calibration (ECal) dialog box of the Setup Wizard appears.
3. Follow the instructions on the wizard then click the Calibrate button in the dialog box.
4. Follow the instructions in the dialogs that will guide you through the calibration.

5. Click Next >.

6. Fixture Compensation is an optional process.

7. If you choose to perform fixture compensation, follow the instructions on the wizard then click the **Fixture Comp** button in the dialog box. This automatically compensates the electrical length of the cables and fixtures of all the ports.

8. Alternatively, you can compensate one port at a time. To perform this, instead of clicking the **Fixture Comp** button, click the **Options** button in the dialog.

9. At the **Fixture Compensation** dialog box, you can click one port at a time.

10. Clicking **Measure All** compensates all the ports at the same time. This action is the same as clicking the **Fixture Comp** button. You must compensate all the active ports.

11. At **Standard Type**, select either **Open** or **Short** compensation.

12. Click **OK** to close the **Fixture Compensation** dialog box.
13. Once complete, click **Finish** on the dialog box.
Making Measurements

- Setting up Parameters on Each Trace
- Controlling Trigger
- Using Scale/Zooming
- Using Marker and Marker Search
- Using Data and Memory
- Using Gating
- Using Trace Control
- Hot TDR Measurement

Other topics about Enhanced Time Domain Analysis
Setting Up Parameters on Each Trace

- Selecting Trace
- Changing Displayed Trace
- Selecting Parameters
- Defining the Stimulus
- Peeling
- Smoothing

Other topics about Making Measurement

Selecting Trace

Using Mouse

- Double-click on any area of the graph plot to exit from full view.
- Click on the desired trace to select the trace.

Using Hardkey /SoftTab /Softkey

- Press the Prev or Next hardkey.

Using Softkey in TDR GUI

Click on the Trace button and select the trace number from the list as shown below.

Note: The number of traces is changeable and up to 16 traces can be displayed in this option. 16 traces are displayed when Differential 2-port DUT topology, and all time domain (T) and all S-parameter (S) traces are selected.
Changing Displayed Trace

Changing Displayed Trace

Using Mouse

1. Double-click on the single trace graph window to exit from full view of active trace as shown below.

2. Double-click on the desired trace for full view.
Selecting Parameters

Click on the TDR/TDT tab in order to get to the Parameters tab selection as shown below.

The table below shows the Measurement and Format available under the Parameter setting:

<table>
<thead>
<tr>
<th>Measure</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-Parameter</td>
<td>Log Mag</td>
</tr>
<tr>
<td></td>
<td>Linear</td>
</tr>
<tr>
<td></td>
<td>Real</td>
</tr>
<tr>
<td></td>
<td>Imaginary</td>
</tr>
<tr>
<td></td>
<td>Group Delay</td>
</tr>
<tr>
<td></td>
<td>SWR</td>
</tr>
<tr>
<td></td>
<td>Phase</td>
</tr>
<tr>
<td></td>
<td>Expand Phase</td>
</tr>
<tr>
<td></td>
<td>Positive Phase</td>
</tr>
<tr>
<td></td>
<td>Smith (Re/Im)</td>
</tr>
<tr>
<td></td>
<td>Smith (G+jB)</td>
</tr>
<tr>
<td></td>
<td>Polar (Re/Im)</td>
</tr>
<tr>
<td>Time Domain</td>
<td>Impedance</td>
</tr>
<tr>
<td></td>
<td>Volt</td>
</tr>
<tr>
<td></td>
<td>Log Mag</td>
</tr>
<tr>
<td></td>
<td>Linear</td>
</tr>
<tr>
<td></td>
<td>Real</td>
</tr>
</tbody>
</table>
The table in the Parameter area changes as the selection of Measure changes. The changes can be referred as below:

<table>
<thead>
<tr>
<th>Measure</th>
<th>Table Content starts with</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-Parameter</td>
<td>Single-Ended S</td>
</tr>
<tr>
<td></td>
<td>Differential Sc, Sd</td>
</tr>
<tr>
<td>Time Domain</td>
<td>Single-Ended T</td>
</tr>
<tr>
<td></td>
<td>Differential Tc, Td</td>
</tr>
</tbody>
</table>

**Defining the Stimulus**

There are two options to choose from under the Stimulus:

- Lowpass Step
- Lowpass Impulse

Only for the Lowpass Step the selection for Rise Time is active. The two options for Rise Time are:

- 10-90%
- 20-80%

The rise time settings in TDR/TDT mode and Eye/Mask mode are independent.
Peeling

When a device has two or more impedance discontinuities, reflections from the second discontinuity reflects off the first discontinuity. This complex interaction of secondary reflections from the stimulus pulse compromises the measured impedance profile and decreases the measurement performance.

TDR peeling compensates for the complicated interaction between the discontinuities. TDR peeling analysis reflects the signals at the source and de-convolves the time domain reflections to create an impedance profile of the device being tested. This option is available when the measurement parameter is set to time domain reflection and the format is impedance.

Limitation

- TDR peeling does not account for frequency response losses (for example, PC board transmission lines are lossy devices). The sum of the waves that are incident on a node are assumed to be equivalent to those exiting the node.

- TDR peeling assumes a lossless transmission line (resistance of 0). Any actual resistance (which causes loss, even at DC) degrades the accuracy of peeling.

- Initial impedance mismatch is the most accurate; as distance increases from initial impedance mismatch down the transmission line, the impedance accuracy decreases.

- TDR peeling cannot be used on TDT responses, because the lack of reflections invalidates the algorithm.

How to set Peeling

1. Click TDR/TDT tab.
2. Select the Peeling check box in the Parameter tab.

Smoothing

Smoothing is only available when S-Parameter is measured and only applicable to reflection coefficient only (S11, S22, S33, S44).

How to set Smoothing

1. Click TDR/TDT tab.
2. Select the Smoothing check box in the Parameter tab.
Controlling Trigger

Making Single/Continuous measurement

Single measurement

- Click on the **Stop Single** button to enable single measurement.

Continuous measurement

- Click on the **Run** button to enable continuous measurement.

Other topics about Making Measurement
Using Scale/Zooming

- Using Autoscale
- Changing Scale Manually
- Zooming

Other topics about Making Measurement

Using Autoscale

1. Select the trace which you want to change the scale.
2. Click on the Auto Scale button on the tool bar or right-click in the graph plot area.
3. Select the desired scale. Refer to the table below for the details of each selection.

<table>
<thead>
<tr>
<th>Scale</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto Scale X</td>
<td>Auto scaling X axis only, on the active trace</td>
</tr>
<tr>
<td>Auto Scale Y</td>
<td>Auto scaling Y axis only, on the active trace</td>
</tr>
<tr>
<td>Auto Scale X &amp; Y</td>
<td>Auto scaling both X &amp; Y axis, on the active trace</td>
</tr>
<tr>
<td>Auto Scale All</td>
<td>Auto scaling both X &amp; Y axis, on all traces</td>
</tr>
</tbody>
</table>

Changing Scale Manually

Changing Horizontal Scale

1. Select the trace which you want to change the scale.
2. Click the **TDR/TDT** tab.

- **Using Virtual knob**
  
a. Click the left knob under **Horizontal**. The center of the knob turns blue.
  
b. Scroll up or down the wheel mouse button to change the scale.

- **Type in Entry box**
  
a. Click in the box below the left knob under **Horizontal**. An **Entry** dialog box appears.
  
b. Type the precise value.

- **Clicking Wave icons**
  
a. Click the wave icons above the left knob under **Horizontal**. Details are as table below:

<table>
<thead>
<tr>
<th>Function Icon</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="wave1.png" alt="Wave Icon" /></td>
<td>Click to increase the scale in a 1-2-5 sequence. The width of the waveform compresses. The default value for basic time units is 1.0 ns/div</td>
</tr>
<tr>
<td><img src="wave2.png" alt="Wave Icon" /></td>
<td>Click to decrease the scale in a 1-2-5 sequence. The width of the waveform expands.</td>
</tr>
</tbody>
</table>

**Reference Position for Horizontal Axis**

The reference point of horizontal axis can be selected from the center line or left edge of the displayed graticule.

Click the icon as shown in table below under Horizontal to change the reference point:

<table>
<thead>
<tr>
<th>Function Icon</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="ref1.png" alt="Reference Icon" /></td>
<td>Reference point of the display graticule to the left edge. The position of left edge is not changed even if you change the scale.</td>
</tr>
<tr>
<td><img src="ref2.png" alt="Reference Icon" /></td>
<td>Reference point of the display graticule to the center. The position of center is not changed even if you change the scale.</td>
</tr>
</tbody>
</table>

**Changing Horizontal position**
1. Select the trace which you want to change the horizontal position.

2. Click the **TDR/TDT** tab.

   • **Using Virtual knob**

     a. Click the right knob under **Horizontal**. The center of the knob turns blue.
     b. Scroll up or down the wheel mouse button to change the position.

   • **Type in Entry box**

     a. Click the box below the right knob under **Horizontal**. An **Entry** dialog box appears.
     b. Type the precise value.

   • **Clicking Arrow icons**

     a. Click the arrow icons above the right knob under **Horizontal**. Details are shown in the table below:

     | Function Icon | Details                                                                 |
     |---------------|-------------------------------------------------------------------------|
     |               | Click to increase the delay from trigger; the waveform moves to the left of the display. |
     |               | Click to decrease the delay from trigger; the waveform moves to the right of the display. |

**Changing Vertical Scale**

1. Select the trace which you want to change the scale.

2. Click the **TDR/TDT** tab.

   • **Using Virtual knob**

     a. Click the left knob under **Vertical**. The center of the knob turns blue.
     b. Scroll up or down the wheel mouse button to change the scale.

   • **Type in Entry box**
a. Click the box below the left knob under **Vertical**. An **Entry** dialog box appears.
b. Type the precise value.

- **Clicking Wave icons**

  a. Click the wave icons above the left knob under **Vertical**. Details are as table below:

<table>
<thead>
<tr>
<th>Function Icon</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Click to increase the scale in a 1-2-5 sequence; the waveform height compress.</td>
</tr>
<tr>
<td></td>
<td>Click to decrease the scale in a 1-2-5 sequence; the waveform height will expand.</td>
</tr>
</tbody>
</table>

**Changing Vertical Position**

1. Select the trace which you want to change the vertical position.
2. Click the **TDR/TDT** tab.

- **Using Virtual knob**

  a. Click the right knob under **Vertical**. The center of the knob turns blue.
b. Scroll up or down the wheel mouse button to change the position.

- **Type in Entry box**

  a. Click the box below the right knob under **Vertical**. An **Entry** dialog box appears.
b. Type the precise value.

- **Clicking the Arrow icons**

  a. Click the arrow icons above the right knob under **Vertical**. Details are as table below:
<table>
<thead>
<tr>
<th>Function Icon</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>▼</td>
<td>Click to increase the offset value; the waveform shift downward.</td>
</tr>
<tr>
<td>▲</td>
<td>Click to decrease the offset value; the waveform shift upward.</td>
</tr>
</tbody>
</table>

- Drag and drop

1. Click and hold the y-axis of the graph plot.
2. Drag up or down to the desired position and release the button.

**Zooming**

To zoom in the graph plot:

1. Left-click and hold the mouse button.
2. Drag the area you would like zoom and release the mouse button.
3. Select zoom as shown below.
Using Marker and Marker Search

- Activating Marker
- Moving Marker
- Using Reference Marker
- Searching Max/Min Points on Trace
- Measuring Rise Time
- Measuring Delta Time

Other topics about Making Measurement

Activating Marker

1. Click on the **Marker Off** button.

2. Select 1 from the list to activate Marker 1.

3. The marker button shows Marker Off when there is no active markers. The marker button shows Marker Ref when reference marker is active. This applies to all markers. Active marker has an arrow head pointing down. Inactive markers have an arrow head pointing up. Only markers 1 and 2 have vertical dotted lines. All marker values are displayed at the upper right corner of the graph view. There are a maximum of 9 markers that can be activated.
Moving Marker

- Click and hold on the marker, drag the marker to the desired point and release the mouse button.

Using Reference Marker

When reference marker is used, all 9 markers use reference marker as reference point.

1. Click on the Marker Off button.
2. Select Ref from the list.

Moving reference marker is similar to moving marker. Simply choose the reference marker instead of the marker number.

Searching Max/Min Points on Trace

Searching Max point on trace

1. Click on the desire trace.
2. Click on the Marker Search button.
3. Select **Max** from the list.

**Searching Min point on trace**

1. Click on the desire trace.
2. Click on the **Marker Search** button.
3. Select **Min** from the pull down menu.
4. The marker is constantly on tracking mode when **Max** or **Min** is selected. Even moving the marker with the mouse, the marker tracks the **Max** or **Min** of the trace and points to it.

**Measuring Rise Time**

1. Click on the desire trace.
2. Click on the **Marker Search** button.
3. Select **Rise Time (10-90%)** or **Rise Time (20-80%)** from the list.

The data is displayed at the top right corner of the graph plot.

*Rise Time: 151.48 ps (45.413 mm)*

**Measuring Delta Time**

Delta Time measurement available for Time Domain. Traces other than time domain are not able to use for delta time measurement. Delta time is compared with trace save in memory, if available, else it is compared with the trace selected.

1. Click on the trace that is the starting point for the Delta Time.
2. Click on the **Marker Search** button.
3. Select **Δ Time** from the list and select the **Δ Time** checkbox.
4. Select the trace that will be the stopping point.
5. The trace of the starting point can not be changed, therefore it is selected at the beginning. You can compare Data with Memory, by selecting the same trace. (only if **Data & Memory** is selected using the **Data Mem** button)

5. Select the Position to be measured.
6. Click on the OK button.

The data is displayed at the top right corner of the graph plot.

*Delta Time (Tr7): 624.08 fs (187.09 μm)*
Using Data and Memory

- Storing Data Traces to Memory
- Display the Memory Traces

Other topics about Making Measurement

Data displayed on the graph plot can be stored in the memory and recalled for later use. Data traces stored in the memory will be deleted once TDR option restarts.

Storing Data Traces to Memory

1. Click on the **Data Mem** button.
2. Select **Data -> Mem** from the list.

Display the Memory Traces

1. Click on the **Data Mem** button.
2. Select **Memory** from the list.

Table below explains other functions in the Data Mem list:
<table>
<thead>
<tr>
<th>Function</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data -&gt; Mem</td>
<td>Save Data traces to Memory</td>
</tr>
<tr>
<td>Off</td>
<td>Display off</td>
</tr>
<tr>
<td>Data</td>
<td>Display Data traces only</td>
</tr>
<tr>
<td>Memory</td>
<td>Display Memory traces only</td>
</tr>
<tr>
<td>Data &amp; Memory</td>
<td>Display Data and Memory traces</td>
</tr>
</tbody>
</table>
Using Gating

- Overview
- Coupling Gate on Several Traces
- Setting Start/Stop Points
- Selecting Gating Type
- Activating Gate

Other topics about Making Measurement

Overview

Gating provides the ability to observe the effect of a particular circuit element on frequency domain response by virtually removing undesired responses. When you define a gate on time domain plot, the gated section is removed and replaced mathematically with an ideal transmission line having the same electrical delay as the removed section. Gating is applied to time plots of individual parameters. While gating the time domain plot of a parameter, you can observe the effect that gating has on the frequency domain of the same parameter by coupling the relevant traces using the gate coupling feature. By observing the original frequency domain response and the transformed frequency domain response, the effect of the gating operation on the S-parameter data can be seen.

When a discontinuity in a test device reflects energy, that energy will not reach subsequent discontinuities. This can "MASK", or hide, the true response which would have occurred if the previous discontinuity were not present. The Gating feature does NOT compensate for this.

Coupling Gate on Several Traces

Gate Coupling allows two or more traces to share the same gating values. The gating value of the active trace is coupled to other trace(s).

It is used to observe the gated frequency response while setting a gate on time response.

Operational procedure

1. Click the TDR/TDT tab.
2. Under TDR/TDT, click the Gating tab.

3. Under Gating, click the Gate Coupling button.

4. The Gate Coupling dialog box appears.

5. Select the time domain trace and the associated S-Parameter trace. For example, trace of T11 and trace of S11.

6. Click OK.

The gate coupling should not be applied to unrelated responses. For example, TDR (T11) and Insertion Loss (S21).

Selecting a lot of traces the gate coupling makes the response of virtual knobs slow. Type your desired value instead of rotating virtual knobs to specify the value.

Setting Start/Stop Points

Gating is set on the active trace. There are several methods to set the start and stop points of the gate:

Setting points at gating tab
1. Click the **TDR/TDT** tab.

2. Under **TDR/TDT**, click the **Gating** tab.

3. Click on the **Start** or **Stop** virtual knob.

4. The knob is highlighted in blue once its enabled.

5. Click on the virtual several times clockwise to increase the value and anti-clockwise to decrease the value.

6. You can observe the start or stop time in the text box below the virtual knob and gate markers on the active trace.

7. Optionally, you can click the text box under the **Start** or **Stop** virtual knob.

8. An **Entry** dialog box appears.
9. Type the start or stop time and click OK.

10. The new value will be displayed at text box and x-axis of the active trace.

The **Start** virtual knob and text box sets the start time and the **Stop** sets the stop time.

The start time and stop time should be set at a point on which the impedance is close to the reference impedance.

**Setting points at active trace graph plot**

1. On the time domain active trace, drag your mouse across the area to set the gating.

2. Select **Set Gating Range**.

3. You can observe two dashed lines indicating the gate on the graph plot. These are gate markers.

4. You can also observe the time value in the text box below the virtual knob under the **Gating** tab in the **TDR/TDT** area.
Selecting Gating Type

The Enhanced Time Domain Analysis module allows you to choose from the following two gate types:

<table>
<thead>
<tr>
<th>Gate type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Band pass</td>
<td>Removes response outside the gate range</td>
</tr>
<tr>
<td>Notch</td>
<td>Removes response inside the gate range</td>
</tr>
</tbody>
</table>

Operational procedure

1. Click the TDR/TDT tab.
2. Under TDR/TDT, click the Gating tab.
3. Under Gating, select one of the gating types from the Type area.

Activating Gate

Ensure that you have set the start and stop time. Then follow the following procedure:

1. Click the TDR/TDT tab.
2. Under TDR/TDT, click the Gating tab.
3. Under Gating, select the Gating check box.

Switch Gating State

1. Right-click on the time domain active trace.
2. Select Switch Gating State.
3. If gating was turned ON earlier, this step will turn it OFF and vise versa. You can observe the changes in the Gating check box.
Changing Trace Allocation

Changing the trace allocation affects the data display on the graph plot. Table below shows the details of each selection:

<table>
<thead>
<tr>
<th>Allocation</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixed</td>
<td>Display mixed of commonly measured time domain and S-parameter data</td>
</tr>
<tr>
<td>All T</td>
<td>Display all Time Domain data for selected device topology</td>
</tr>
<tr>
<td>All S</td>
<td>Display all S-Parameter data for selected device topology</td>
</tr>
</tbody>
</table>

1. Click the **TDR/TDT** tab.
2. Click on the desired trace allocation in the **Trace Control** tab under **Allocation**.

Coupling Marker/Time

1. Click the **TDR/TDT** tab.
2. Click on the desired check box in **Trace Control** tab under **Coupling**.

- Selecting **Marker** under **Coupling** enables all the marker on other traces to be moved in same alignment.
- Selecting **Time** under **Coupling** enables all other traces using the same X axis (Time).
- Selecting **Rise Time** under **Coupling** enables all other traces using the same rise time.
- Selecting **Scale** under **Coupling** enables all other traces using the same scale.

**Target Annotation/Readout**

- Selecting **Trace Annotation** under **Active Trace Only** enables the trace annotation only on an active trace.
- Selecting **Marker Readout** under **Active Trace Only** enables the marker readout only on an active trace.

**Copying Trace Setting**

1. Click on the **Trace Settings Copy** button. The **Trace Settings Copy** dialog box appears.
2. Select the source trace in the **From** list. Select the desired destination trace in the **To** list.
3. Click on the **>> Copy >>** button.

**Copied Parameter**

The following parameters for the following functions are copied by the Trace Settings Copy.

<table>
<thead>
<tr>
<th>Functions</th>
<th>SCPI Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter, Time</td>
<td>CALCulate:TDR:MEASure:PARameter</td>
</tr>
<tr>
<td>Domain/S-</td>
<td></td>
</tr>
<tr>
<td>Parameter, Single-</td>
<td></td>
</tr>
<tr>
<td>Ended/Differential</td>
<td></td>
</tr>
<tr>
<td>Format</td>
<td>CALCulate:TDR:MEASure:FORMat</td>
</tr>
<tr>
<td>Marker</td>
<td>CALCulate:TDR:MEASure:ACTive:MARKer</td>
</tr>
<tr>
<td>Peeling</td>
<td>CALCulate:TDR:MEASure:PEELing:STATe</td>
</tr>
<tr>
<td>Delta Time Dialog</td>
<td>CALCulate:TDR:MEASure:DTIME:POSition</td>
</tr>
<tr>
<td>Delta Time Target</td>
<td>CALCulate:TDR:MEASure:DTIME:TARGET</td>
</tr>
<tr>
<td>[Target is trace for stop]</td>
<td></td>
</tr>
<tr>
<td>Gating Start</td>
<td>CALCulate:MEASure:FILTER:GATE:TIME:STARt</td>
</tr>
<tr>
<td>Gating Stop</td>
<td>CALCulate:MEASure:FILTER:GATE:TIME:STOP</td>
</tr>
<tr>
<td>Gating Type</td>
<td>CALCulate:MEASure:FILTER:GATE:TIME:TYPE</td>
</tr>
</tbody>
</table>
Marker Search
[ON/OFF], marker [0-9, ref]
Marker Search
[MIN/MAX], marker [0-9, ref]
Reference Marker
[ON/OFF]
Marker
CALCulate:MEASure:MARKer:STATe
Marker [x-axis value]
CALCulate:MEASure:MARKer:X
Smoothing
CALCulate:MEASure:SMOothing:STATe
Impulse Width
Value for Lowpass Impulse
Rise Time (for all traces)
Stimulus
Search Rise Time
Horizontal Scale
Horizontal Position
Vertical Scale
Vertical Position
CALCulate:MEASure:TRANsform:TIME:IMPulse:WIDTh
CALCulate:MEASure:TRANsform:TIME:STEP:RTIME
CALCulate:MEASure:TRANsform:TIME:TYPE
CALCulate:TDR:MEASure:TTIME:STATe
CALCulate:TDR:MEASure:TTIME:THReshold
DISPlay:TDR:MEASure:X:SCALe:PDIVision
DISPlay:TDR:MEASure:X:SCALe:RLEVel
DISPlay:TDR:MEASure:Y:SCALe:PDIVision
DISPlay:TDR:MEASure:Y:SCALe:RLEVel
Hot TDR Measurement

- Overview
- Checking Device Malfunction
- Reducing Measurement Error

Other topics about Making Measurement

Overview

Hot TDR refers to TDR measurement while the device is powered ON. During measurement, the measurement signal is applied from the VNA to the transmitter. However, this may cause device malfunction. In addition, the transmitter signal from the DUT device into the VNA ports may cause measurement error. When you measure Hot TDR, the transmitter should be connected with Port 1 for Single END, Port1/2 for differential.

Checking Device Malfunction

The following procedure checks if the Signal Source level from the VNA does not affect the device operation:

1. Measure the reflection on the VNA.
2. Save the results in memory trace.
3. Reduce the signal source level of the VNA at Source Power.
4. Check for significant change on the trace. If the change is significant, reduce the signal source level further to avoid device malfunction.

Reducing Measurement Error
The following procedure reduces the measurement error produced by the device signal.

**Case 1 (For Periodic Bit Pattern)**

If the output signal from device is periodic, set the data rate and execute "Avoid Spurious". This will reduce error due to spurious:

1. Click **Setup** tab and select the **Hot TDR** tab.

2. At the **Data Rate** text box, left-click once. An **Entry** dialog box appears. Type the data rate value and click **OK**. The new value is displayed in the **Data Rate** text box. The Data Rate accuracy should be within ±0.5%.

3. Click the **Avoid Spurious** button to execute the option. The VNA searches for spurious and changes the stimulus setting to avoid the spurious. If the Avoid Spurious is successfully executed, a check mark appears next to the **Avoid Spurious** button. At this point, measurement mode is changed from TDR/TDT to Hot TDR Mode and this is indicated at the channel window as "TDR ?". The blue **SVC** indicator is also turned ON.
   
   a. Eye/Mask option tab is disabled in HOT TDR mode.
   
   b. To reset the HOT TDR mode, simply execute preset or change the DUT topology.

**Case 2 (For Random Data)**

If the output signal from device is random, increase averaging to reduce the measurement error.

**About Avoid Spurious**

If Avoid Spurious fails, the "Spurious Not Found" warning message is displayed. Check mark next to the Avoid Spurious button will not be displayed.

When the following parameter is changed, Avoid Spurious option should be executed again:

- Data Rate
- DUT Length
- Deskew

The setting of Avoid Spurious cannot be stored. To recall the condition of Avoid Spurious, you must execute the Avoid Spurious again after recalling the status setting.

At the execution of Avoid Spurious:

- If the IF bandwidth is over 10 kHz, the IF bandwidth value will be set at 10 kHz.
- If the source power is over -20 dBm, the source power will be set at -20 dBm.
### Eye Diagram and Mask Test (Eye/Mask Tab)

- Performing Eye Diagram Measurements
- Selecting Bit Pattern
- Using Mask Test
- Available Masks

### Other topics about Enhanced Time Domain Analysis
Performing Eye Diagram Measurements

- Overview
- Showing Eye Diagram
- Displaying Results
- Scaling the Eye Diagram
- Injecting Jitter

Other topics about Eye Diagram and Mask Test

Overview

In the oscilloscope, an eye diagram is often used to analyze signal quality. You can diagnose problems, such as attenuation, noise, jitter, and dispersion that arise or characterize specific parts of the system with one display.

The VNA option TDR provides simulated eye diagram analysis capability, eliminating the need for a hardware pulse pattern generator. The virtual bit pattern generator is used to define a virtual bit pattern. The defined bit pattern is then convolved with the device impulse response to create an extremely accurate measurement based eye diagram.

Showing Eye Diagram

1. Select the trace number which you want to observe the eye diagram.
2. Click the Eye/Mask tab.
3. Click Draw Eye to display the eye diagram.
4. Whenever you change the setting of data pattern, it is required to click Draw Eye to reflect the setting on the waveform.

Displaying Results

1. Select Rise Time Def in the Results tab.
2. Click Draw Eye to display the result.
   Whenever you change the setting of Rise Time Def, it is required to click Draw Eye to reflect on the result.
The following results are displayed on the table under Results.

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rise Time</td>
<td>Second</td>
<td>Rise Time Def=10%-90%: Time at 90% level - Time at 10% level</td>
</tr>
<tr>
<td>Fall Time</td>
<td>Second</td>
<td>Rise Time Def=20%-80%: Time at 80% level - Time at 20% level</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rise Time Def=10%-90%: Time at 90% level - Time at 10% level</td>
</tr>
<tr>
<td>Jitter RMS</td>
<td>Second</td>
<td>1σ width of the histogram at the eye crossing point</td>
</tr>
<tr>
<td>Jitter p-p</td>
<td>Second</td>
<td>Full width of histogram at the eye crossing point</td>
</tr>
<tr>
<td>Crossing Percentage</td>
<td>%</td>
<td>Crossing Height / Amplitude × 100</td>
</tr>
<tr>
<td>Opening Factor</td>
<td>None</td>
<td>(Level One - σ_{one}) - (Level Zero + σ_{zero}) / Amplitude</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Level One - Level Zero)/( σ_{one} + σ_{zero})</td>
</tr>
<tr>
<td>Signal/Noise Ratio</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Duty Cycle Distortion</td>
<td>Second</td>
<td></td>
</tr>
<tr>
<td>Duty Cycle Distortion</td>
<td>%</td>
<td>Duty Cycle Distortion (s)/ Bit period × 100</td>
</tr>
<tr>
<td>Level Zero</td>
<td>Voltage</td>
<td>Histogram mean for level zero</td>
</tr>
<tr>
<td>Level One</td>
<td>Voltage</td>
<td>Histogram mean for level one</td>
</tr>
<tr>
<td>Level Mean</td>
<td>Voltage</td>
<td>(Level Zero + Level One) / 2</td>
</tr>
<tr>
<td>Amplitude</td>
<td>Voltage</td>
<td>Level One - Level Zero</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Level One - 3 σ_{one}) - (Level Zero + 3 σ_{zero})</td>
</tr>
</tbody>
</table>
Height Voltage

Width Second Bit Period - 2 × 3 × Jitter RMS

- Bit Period = 1/Bit Rate
- Input Amplitude = Setting of Level One - Setting of Level Zero
- $T_{\text{rise middle}}$ = The time at which the rising edge cross the middle threshold (50%)
- $T_{\text{fall middle}}$ = The time at which the falling edge cross the middle threshold (50%)

**Overlaying the results on the waveform**

1. Select the Overlay check box in the Results tab.
2. The following results are displayed on the screen.
   - Jitter p-p, Level Zero, Level One, Amplitude, Height (V), Width

**Saving Results into File**

You can save the results as a text file.

1. Click Export button in the Results tab. The Save Eye Result dialog box is displayed.
2. Type your desired file name, then click Save.

Example of Result
# Option VNA-TDR Simulated Eye Results

# 7/1/2010 3:10:06 PM

#

Level Zero, 0.00337131636124

Level One, 0.392246236818

Level Mean, 0.19780877659

Amplitude, 0.388874920457

Height, 0.372870737968

Width, 9.94598885146E-10

Opening Factor, 0.986281636548

Signal / Noise, 72.894992429

Duty Cycle Distortion, 6.12868274149E-14

Duty Cycle Distortion (%), 0.00612868274149

Rise Time, 4.69421997336E-11

Fall Time, 4.69412762334E-11

Jitter (PP), 6.25E-12

Jitter (RMS), 9.00185809062E-13

Cross Point (%), 49.9268781576

---

**Scaling The Eye Diagram**

By default, the eye diagram is set to Auto Scale. You can also set the scale manually.

1. Click on the **Scale/Mask** tab.

2. Select the **Manual** radio button. This will activate the **Scale / Div** and **Offset** options.
3. Click in the **Scale / Div** text box and input the Y axis scale value.

4. Click in the **Offset** text box and input the Y axis offset value.

**Mask Pattern**

See *Using Mask Test* for more information.

**Injecting Jitter**

See *Using Jitter Injection* for more information.
Selecting Bit Pattern

- Overview
- Bit Pattern Type
- Settings Parameters of Bit Stream

Other topics about Eye Diagram and Mask Test

Overview

TDR can provide simulated eye diagram analysis capability, eliminating the need for a hardware pulse pattern generator. The virtual bit pattern can be selected from:

- Pseudo-Random Bit Sequence
- K 28.5
- User Custom
- Statistical

Bit Pattern Type

The following Bit Patterns can be used to develop an Eye Diagram:
Bit Pattern | Description
--- | ---
PRBS | Pseudo-Random Bit Sequence. An industry standard created from a specified pattern length. For example, when $2^7$ is selected, 127 $(2^7 - 1)$ unique data 'words' are assembled according to the industry standard.
K 28.5 | Industry standard developed by IBM which includes comma (control) characters. The pattern is "00111110101100000101" (20 bits).
User | Bit Patterns that you have created.
Statistical | Bit Patterns produced via statistical calculations of jitter specification. When this option is selected, eye diagram is displayed as "Statistical" type. When Jitter Injection is turned ON, this option is set as the default selection. Refer to Using Jitter Injection.

In the user bit pattern, you can set the same bit pattern as the Pseudo-Random Bit Sequence. However you can get much better resolution in result when you use PRBS.

### Selecting Bit Pattern

1. Select the Eye/Mask tab.
2. Select your desired bit pattern at Type under Bit Pattern.
3. If you select the PRBS option, length is activated. Then, select length under Bit Pattern.

### Using a User Bit Pattern

You can easily create user (custom) bit patterns. The length of bit should be from 2 to 32768 ($2^{15}$). The pattern with only either 0 or 1 can not be accepted (ex. 00, 111, 0000).

### Defining/Saving User Bit Pattern

1. Select the Eye/Mask tab.
2. Select User at Type under Bit Pattern, then User Pattern is activated.
3. Click User Pattern, then Bit Pattern Editor is displayed.
4. Type "0" or "1" to create your bit pattern.
5. Click OK, then the Save Bit Pattern dialog box is displayed.
6. Type your desired file name, then click Save.
7. Saving pattern to the file must be required when you use the user pattern.
Recalling User Bit Pattern

1. Select the **Eye/Mask** tab.

2. Click **User Pattern**, then **Bit Pattern Editor** is displayed.

3. Click **Load**, then **Load Bit Pattern** dialog box is displayed.

4. Select your desired file name, then click **Open**.

5. Click **Ok** to exit **Bit Pattern Editor**.

Settings Parameters of Bit Stream

The following parameter can be set for the bit stream.

<table>
<thead>
<tr>
<th>Label</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>One Lv.</td>
<td>Eye Diagram Y-axis scaling for bit &quot;1&quot; in volts. Negative voltages are allowed. For Differential Eye Diagrams, these scale values are doubled.</td>
</tr>
<tr>
<td>Zero Lv.</td>
<td>Eye Diagram Y-axis scaling for bit &quot;0&quot; in volts. Negative voltages are allowed. For Differential Eye Diagrams, these scale values are doubled.</td>
</tr>
<tr>
<td>Data Rate</td>
<td>The speed in bits/second which data is transferred over a circuit or a communications line.</td>
</tr>
<tr>
<td>Rise Time</td>
<td>The time that it takes a signal to transition from a low to a high condition. Maximum value is 40% of Bit width (Bit width =1/Bit Rate). The time can be defined by either &quot;10-90%&quot; or &quot;20-80%&quot;. The rise time settings in EYE/MASK mode and TDR/TDT mode are independent.</td>
</tr>
</tbody>
</table>

Defining the parameters

1. Select the **Eye/Mask** tab.

2. Click the text box of desired parameter under **Stimulus**, then the **Entry** dialog box is displayed.

3. Type your desired number by clicking numeric keys on the **Entry** dialog box.
Using Mask Test

- Overview
- Defining Mask
- Executing Mask Test

Other topics about Eye Diagram and Mask Test

Overview

The mask test allows you to verify that a displayed waveform complies with industry-standards definitions for electrical waveforms. To comply with the industry standard, the input waveform must remain outside the shaded mask regions. The mask testing is available in Scale/Mask under the Eye/Mask tab. The TDR uses the same format as Infiniium DCA (86100C), therefore, you can use the MASK file (.msk) stored by DCA (86100C).

The some masks with industry-standards definitions are available in the VNA directories.

Defining Mask

Opening Mask File
<table>
<thead>
<tr>
<th>Items</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File Name</td>
<td>Shows the file name and location which is currently selected. The pre-defined files and templates files are available under C:\Program Files (x86)\Keysight\Network Analyzer\masks.</td>
</tr>
<tr>
<td>Type</td>
<td>Shows the Mask File Identifier. In case of Infiniium DCA, this identifier should be &quot;MASK_FILE_861XX&quot;. However, the VNA does not care about the identifier. The VNA will accept it even if this is other than &quot;MASK_FILE_861XX&quot;. As the pre-installed MASK files are compatible with Infiniium DCA, &quot;MASK_FILE_861XX&quot; is displayed when you use pre-installed MASK files. It is not possible to change this in the Mask Pattern Dialog box.</td>
</tr>
<tr>
<td>Description</td>
<td>Shows the description of MASK file. You can change this in the Mask Pattern Dialog box.</td>
</tr>
</tbody>
</table>

**Loading Mask File**

1. Select the **Eye/Mask** tab.

2. Click the **Mask Pattern** button in the **Scale/Mask** tab, then the **Mask Pattern** dialog is displayed.

3. Select the **File** tab.
4. Click **Load**, then the **Load Mask Pattern** dialog is displayed.

5. Select your desired mask file, then click **Open**.
   
   - If the message prompting you to include absolute values for zero/one level is displayed, define the values of the logic 1 and logic 0 levels in the **Logic 1 Lv (V)** and **Logic 0 Lv (V)** in the **Scale** tab then save the file.

6. Click **Close** to exit the **Mask Pattern** dialog box.

The pre-defined files are read-only file. As you cannot overwrite on them, save the file by clicking **Save As**.

**Scaling Mask**
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Margin (%)</td>
<td>Set the size of the mask margin. Mask margins are used to determine the margin of compliance for a standard or scaled mask</td>
</tr>
<tr>
<td>Position (s)</td>
<td>Move X-axis location of mask</td>
</tr>
<tr>
<td>Delta T (s)</td>
<td>Change X-axis width of mask</td>
</tr>
<tr>
<td>Logic 1 Lv (V)</td>
<td>Change voltage of logical 1.</td>
</tr>
<tr>
<td>Logic 0 Lv (V)</td>
<td>Change voltage of logical 0</td>
</tr>
</tbody>
</table>

**Changing Margin, Position, Delta T and Logic 0/1 Lv**

1. Load the Mask Pattern.
2. Select the Scale tab.
3. Change the margin number using slider, or typing number.
4. Click the box at your desired parameter of position, delta T and logic 0/1 level.
5. Type the number you want to set.
6. Click **Update** to apply the entered number.
7. Click **Close > Yes** to save the modified scale.

**Set Value at Default**

Click **Default Scale** to set the parameter at default.

**Editing Mask**
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Region</td>
<td>The currently selected region number. The region number defines a mask violation area (or polygon).</td>
</tr>
<tr>
<td>Point</td>
<td>The currently selected point number. The point number defines a point in the region.</td>
</tr>
<tr>
<td>X (s) and Y (V)</td>
<td>The X and positions for the selected point. You can enter the number to change the selected position location.</td>
</tr>
<tr>
<td>Std.X and Std.Y</td>
<td>The X and Y positions when Margin in Scale Tab is selected at 0%. This shows the positions of STD in the .msk file.</td>
</tr>
<tr>
<td>Max.X and Max.Y</td>
<td>The X and Y positions when Margin in Scale Tab is selected at 100%. This shows the positions of MARGIN_MAX in the .msk file.</td>
</tr>
<tr>
<td>Min.X and Min.Y</td>
<td>The X and Y positions when Margin in Scale Tab is selected at -100%. This shows the positions of MARGIN_MIN in the .msk file.</td>
</tr>
</tbody>
</table>

1. **Load the Mask Pattern.**

2. **Select the Edit tab.**

3. **Move the position of points by either way.**
Using Mouse

a. Click the desired point on the figure of mask, then the point is selected (the point color becomes red).
b. Move the point with drag and drop on the mouse to your desired position.

Entering Position

a. Click desired point on the figure of mask, then the point is selected.
b. Click entry box of X(s) under position, then type number for X axis.
c. Click entry box of Y(V) under position, then type number for Y axis.
d. Click Update to apply the entered number on the selected point.

4. Click Close > Yes to save the modified pattern.

Executing Mask Test

1. Select the Mask Test check box in the Scale/Mask tab.
2. Click Draw Eye to redraw eye pattern and mask.
3. The mask and pass/fail result is displayed on the screen
Available Masks

The following tables list the available standard masks files that you can use. The TDR uses the same format as the Infinium DCA (86100C). The following available masks are the same as ones stored in DCA and stored under C:\Program Files (x86)\Keysight\Network Analyzer\masks.

- SDH/SONET
- Fibre Channel
- Ethernet
- Infiniband
- Serial ATA
- XAUI
- Electrical
- G_984_2

Other topics about Eye Diagram and Mask Test

SDH/SONET

<table>
<thead>
<tr>
<th>File Name</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>STM000_OC1.msk</td>
<td>STM0/OC1, 51.8 Mb/s</td>
</tr>
<tr>
<td>STM001_OC3.msk</td>
<td>STM1/OC3, 155.5 Mb/s</td>
</tr>
<tr>
<td>STM004_OC12.msk</td>
<td>STM4/OC12, 621.8 Mb/s</td>
</tr>
<tr>
<td>STM008_OC24.msk</td>
<td>STM8/OC24, 1244 Mb/s</td>
</tr>
<tr>
<td>STM016_OC48.msk</td>
<td>STM16/OC48, 2.488 Gb/s</td>
</tr>
<tr>
<td>STM016_G.691_V2.0.msk</td>
<td>STM16/G.691</td>
</tr>
<tr>
<td>STM064_OC192.msk</td>
<td>STM64/OC192, 9.953 Gb/s</td>
</tr>
<tr>
<td>STM256_OC768.msk</td>
<td>STM256/OC-768, 39.812 Gb/s</td>
</tr>
<tr>
<td>STM64_OC192FEC_10_664.msk</td>
<td>STM-64/OC-192 FEC, 10.664 Gb/s</td>
</tr>
<tr>
<td>STM64_OC192FEC_10_709.msk</td>
<td>STM-64/OC-192 FEC, 10.709 Gb/s</td>
</tr>
<tr>
<td>STM64_OC192SuperFEC_12_5.msk</td>
<td>STM-64/OC-192 Super FEC, 12.5 Gb/s</td>
</tr>
</tbody>
</table>
### Fibre Channel

<table>
<thead>
<tr>
<th>File Name</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>FC0133.msk</td>
<td>FC133, 133 Mb/s</td>
</tr>
<tr>
<td>FC0266.msk</td>
<td>FC266, 266 Mb/s</td>
</tr>
<tr>
<td>FC0531.msk</td>
<td>FC531, 531 Mb/s</td>
</tr>
<tr>
<td>FC1063.msk</td>
<td>FC1063, 1063 Mb/s (Revision 1 of the standard)</td>
</tr>
<tr>
<td>FC1063_PI_R13_Dec01.msk</td>
<td>FC1063, 1063 Mb/s (Revision as of Dec., 2001)</td>
</tr>
<tr>
<td>FC2125.msk</td>
<td>FC2125, 2125 Mb/s (Revision 1 of the standard)</td>
</tr>
<tr>
<td>FC2125_PI_R13_Dec01.msk</td>
<td>FC2125, 2125 Mb/s (Revision as of Dec., 2001)</td>
</tr>
<tr>
<td>FC4250_PI_R13_Dec01.msk</td>
<td>FC4250, 2350 Mb/s</td>
</tr>
<tr>
<td>008.5000-FC-PI-4 Multimode.msk</td>
<td>FC8.5 Gb/s Fibre Channel</td>
</tr>
<tr>
<td>10xFiberChannel.msk</td>
<td>10X Fibre Channel, 10.51875 Gb/s</td>
</tr>
<tr>
<td>010.51875-SFP+_Rx_C'_10GE.msk</td>
<td>10X Fibre Channel, SFP+Rx C'</td>
</tr>
<tr>
<td>010.51875-SFP+_Tx_B_10GE.msk</td>
<td>10X Fibre Channel, SFP+Tx B</td>
</tr>
<tr>
<td>010.51875-XFP_B'_10GE.msk</td>
<td>10X Fibre Channel, XFP B'</td>
</tr>
<tr>
<td>010.51875-XFP_C'_10GE.msk</td>
<td>10X Fibre Channel, XFP C'</td>
</tr>
</tbody>
</table>

### Ethernet

<table>
<thead>
<tr>
<th>File Name</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>01xGbEthernet.msk</td>
<td>GB Ethernet, 1250 Mb/s</td>
</tr>
<tr>
<td>02xGbEthernet.msk</td>
<td>2XGB Ethernet, 2.500 Gb/s</td>
</tr>
<tr>
<td>10GbE_9.953_May02.msk</td>
<td>10GB Ethernet, 9.953 Gb/s</td>
</tr>
<tr>
<td>10GbEthernet_10_3125.msk</td>
<td>10GB Ethernet, 10.3125 Gb/s (Revision 1 of the standard)</td>
</tr>
<tr>
<td>10GbE_10_3125_May02.msk</td>
<td>10GB Ethernet, 10.3125 Gb/s (Revision as of May, 2002)</td>
</tr>
<tr>
<td>10_G_Base_LRM_May_2006.msk</td>
<td>10G BASE LRM, 10 Gb/s</td>
</tr>
<tr>
<td>10xGbEthernet_12_5.msk</td>
<td>10XGB Ethernet, 12.5 Gb/s</td>
</tr>
<tr>
<td>1000BASE-LX10_September_2004.msk</td>
<td>1000BASE-LX10, 1.25 Gb/s</td>
</tr>
<tr>
<td>100BASE-BX10_September_2004.msk</td>
<td>100BASE-BX10_September_2004.msk</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>100BASE-BX_LX10.msk</td>
<td>100BASE-LX10, 100 Mb/s</td>
</tr>
<tr>
<td>100BASE-BX_LX10.msk</td>
<td>100BASE-BX_LX10.msk</td>
</tr>
<tr>
<td>10GBASE-LX4.msk</td>
<td>10BASE-LX4</td>
</tr>
<tr>
<td>010.3125-SFP+_Rx_C'_10GE.msk</td>
<td>10.3125 Gb Ethernet SFP+ Rx C'</td>
</tr>
<tr>
<td>010.3125-SFP+_Tx_B'_10GE.msk</td>
<td>10.3125 Gb Ethernet SFP+ Tx B</td>
</tr>
<tr>
<td>010.3125-XFP_B'_10GE.msk</td>
<td>10.3125 Gb Ethernet XFP B'</td>
</tr>
<tr>
<td>010.3125-XFP_C'_10GE.msk</td>
<td>10.3125 Gb Ethernet XFP C'</td>
</tr>
</tbody>
</table>

**Infiniband**

<table>
<thead>
<tr>
<th>File Name</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>InfiniBand_Jun01.msk</td>
<td>2.500 Gb/s</td>
</tr>
</tbody>
</table>

**Serial ATA**

<table>
<thead>
<tr>
<th>File Name</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>SATA 1.5Gb TX 250 Cycles.msk</td>
<td>SATA TX 250 Cycles, 1.5 Gb/s</td>
</tr>
<tr>
<td>SATA 1.5Gb TX 5 Cycles.msk</td>
<td>SATA TX 5 Cycles, 1.5 Gb/s</td>
</tr>
</tbody>
</table>

**XAUI**

<table>
<thead>
<tr>
<th>File Name</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>XAUI-E_Far_May02.msk</td>
<td>XAUI-E Far</td>
</tr>
<tr>
<td>XAUI-E_Near_May02.msk</td>
<td>XAUI-E Near</td>
</tr>
</tbody>
</table>

**Electrical**

<table>
<thead>
<tr>
<th>File Name</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>STS1Eye.msk</td>
<td>STS1Eye, 51.8 Mb/s</td>
</tr>
<tr>
<td>STS3Eye.msk</td>
<td>STS3Eye, 155.5 Mb/s</td>
</tr>
</tbody>
</table>

**G.984.2**
<table>
<thead>
<tr>
<th>File Name</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>2488.32_G.984.2_Downstream_March_2003.msk</td>
<td>2488.32 Mb/s ONU</td>
</tr>
<tr>
<td>1244.16_G.984.2_Downstream_March_2003.msk</td>
<td>1244.16 Mb/s ONU</td>
</tr>
<tr>
<td>1244.16_G.984.2_UpStream_March_2003.msk</td>
<td></td>
</tr>
<tr>
<td>622.08_G.984.2_Upstream_March_2003.msk</td>
<td>622.08 Mb/s ONU</td>
</tr>
<tr>
<td>155.52_G.984.2_Upstream_March_2003.msk</td>
<td>155.52 Mb/s ONU</td>
</tr>
</tbody>
</table>
Storing Data and Setting

- Saving/Recalling Setting
- Saving Data
- Saving Touchstone Data
- Saving Displayed Image

Other topics about Enhanced Time Domain Analysis
## Saving/Recalling Setting

- Saving Setting
- Recalling Setting
- Compatibility of State Files

### Saving Setting

The setting of TDR can be saved and recalled. The state file of TDR measurements has .tdr file extension.

1. Setup your configuration which you want to save.
2. Click **File** in the tool bar.
3. Select **Save State** from menu, then the **Save State As** dialog is displayed.
4. Type desired file name.
5. Click **Save**.

When you use 2-channel measurement in the **Advanced Mode**, the channel 2 setting is also saved in the .tdr file.

### Recalling Setting

1. Click **File** in the tool bar.
2. Select **Recall State** from menu, then the **Recall State As** dialog is displayed.
3. Select the file name of state file whose file extension is .tdr.
4. Click **Open**.

### Compatibility of State Files
# Compatibility between Mode

The following table shows the compatibility of state file between mode.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Recalling</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Basic</td>
</tr>
<tr>
<td>Saved File By</td>
<td></td>
</tr>
<tr>
<td>Basic</td>
<td>Y</td>
</tr>
<tr>
<td>Advanced</td>
<td>N</td>
</tr>
</tbody>
</table>

Y: Recall is possible.

N: Recall is not possible.
**Saving Data**

The trace data can be saved in `.csv` format.

1. Select the trace which you want to get the data.

2. Click **File** in the tool bar.

3. Select **Save Active Trace Data** or **Save All Trace Data** from menu, then the **Save Trace As** dialog is displayed.

4. Type desired file name.

5. Click **Save**.

**Example of File**

```
# Channel 1
# Trace 4
 Frequency  Formatted Data  Formatted Data
8.49E+06  -5.84E-02  0.00E+00
1.70E+07  -8.27E-02  0.00E+00
2.55E+07  -1.04E-01  0.00E+00
3.40E+07  -1.32E-01  0.00E+00
4.25E+07  -1.52E-01  0.00E+00
5.09E+07  -1.84E-01  0.00E+00
5.94E+07  -2.18E-01  0.00E+00
6.79E+07  -2.70E-01  0.00E+00
```

(data is continued until the end)

---

**Other topics about Storing Data and Setting**
Saving Touchstone Data

- File Extension
- Saving Touchstone Data

Other topics about Storing Data and Setting

File Extension (SnP Format)

The SnP file format is changed depending on the selected DUT topology.

<table>
<thead>
<tr>
<th>File Extension</th>
<th>DUT Topology</th>
</tr>
</thead>
<tbody>
<tr>
<td>.s1p</td>
<td>Single Ended 1</td>
</tr>
<tr>
<td>.s2p</td>
<td>Single Ended 2, Differential 1</td>
</tr>
<tr>
<td>.s4p</td>
<td>Single Ended 4, Differential 2</td>
</tr>
</tbody>
</table>

See the Saving data in Touchstone format for touch stone format.

Saving Touchstone Data

The trace data can be saved in touch stone format.

1. Click File in the tool bar.
2. Select Save Touchstone from menu, then the Save Touchstone As dialog is displayed.
3. Type desired file name.
4. Click Save.
Saving Displayed Image

- Saving Image
- Inverting Display Color

Other topics about Storing Data and Setting

Saving Image

The screen image can be saved in bit map or png format.

1. Click File in the tool bar.
2. Select Save Image from menu, then the Save Image As dialog is displayed.
3. Select the file type in Save as Type from either .bmp or .png.
4. Type desired file name.
5. Click Save.

Inverting Display Color

Inverting display color changes the background color of the screen from black to white. When you print out the saved image on a white paper, the background in white color saves a printer ink.

1. Click File in the tool bar.
2. Select Invert Color from menu, then the background color of the screen is changed from black and white.
3. Save the screen by following the procedure of Saving Image.
Advanced Waveform Analysis

- Overview
- Using Jitter Injection
- Using Emphasis
- Using De-embedding
- Using Equalization

Other topics about Enhanced Time Domain Analysis
In Advanced Waveform Analysis, emphasis and equalization helps to improve the quality of the waveform and subsequently the quality of the eye diagram.

There are two options to display the advance waveform window:

- Click on the **Adv Waveform** tab under the **Setup** tab and click one of these buttons: **Emphasis**, **De-embedding**, **Equalization**.

- Click on **Advanced Waveform** button under the **Eye/Mask** tab

View radio button allows you to select the observation point for TDR/TDT or Eye, according to your
selection. Changing View also affects the trigger mode:

- View before the DUT shows the Stimulus View. At this view, trigger mode at the toolbar changes to "STOP".
- View after the DUT shows the Response View. At this view, trigger mode changes to "RUN".
  - When Stimulus View is selected and the trigger mode "RUN/Single" is selected, the view changes to Response View.

**Other topics about Advanced Waveform Analysis**
Using Jitter Injection

To execute jitter injection, click on Jitter button and check Enable.

When Jitter Injection is turned ON, a check mark appears on the Jitter button. The Stimulus Type (under Eye/Mask tab) is automatically changed to Statistical.

See also Bit Pattern Type.

There are two types of jitter injection available to choose from:

- **Random Jitter** - follows the Gaussian distribution and is represented by the rms value of the Random Jitter distribution.
- **Periodic Jitter** - represented by peak-to-peak value.

In general, Deterministic Jitter and Random Jitter totals up to Total Jitter (TJ). Deterministic Jitter is bounded by a finite magnitude. It can be broken into jitter which is correlated to the data sequence and jitter that occurs independent of data. Periodic Jitter is data independent.
Jitter Unit: You can select from Unit Interval (UI) or Second. Second = Unit Interval / Data Rate

Display Limit: This is a probability density limit. Normally, it is not necessary to change this from default setting. This function allows you to execute a fine tuning of skirt of eye diagram. This applied to eye mask result. So that, you can have the same result on the mask test by adjusting this value. This is not applied to the eye result.
Using Emphasis

1. To execute emphasis, click on Emphasis button and check Enable.

2. Cursor Level:

   - Pre cursor is the ratio between $V_C$ and $V_{b2}$:
     \[
     \text{Pre Cursor} = 20 \log_{10} \left( \frac{V_C}{V_{b2}} \right)
     \]

   - Post 1 cursor is the ratio between $V_{b1}$ and $V_a$:
     \[
     \text{Post 1 Cursor} = 20 \log_{10} \left( \frac{V_{b1}}{V_a} \right)
     \]

   - Post 2 cursor is the ratio between $V_{b2}$ and $V_{b1}$:
     \[
     \text{Post 2 Cursor} = 20 \log_{10} \left( \frac{V_{b2}}{V_{b1}} \right)
     \]

When the emphasis is turned on, the DUT length should be larger than $(1/\text{Data Rate}) \times 3$. 

Other topics about Advanced Waveform Analysis
Using De-embedding

1. To execute de-embedding, click on **De-embedding** button and check **Enable**.
2. If differential topology is selected, load S4P or S2P file. For single ended topology, load S2P file.
3. You can enable each SnP file by checking **Enable** in **Select De-embedding** file.
4. The port orientation is shown and it defers depending on the DUT topology.

**Other topics about Advanced Waveform Analysis**
Using Equalization

To execute equalization, click on **Equalization** button and check **Enable**.

You can either select to use the equation by specifying the variables or equation file.

You must specify four variables in this equation in order to generate the filter. These variables are:

- DC Gain
- Zero Frequency
- Pole 1
- Pole 2

Example of Equalization File

!Keysight Technologies

!VNA-TDR Equalization File

```
# Hz dB
1000000000,-1.53E+01,-3.27E-01  
2000000000,-1.52E+01,-7.34E+01  
3000000000,-1.03E+01,-8.63E+01  
40000000000,-9.88E+00,-1.00E+02
```

-------------------------------------
! Comment
#
Hz MA or DB or RI
Freq1, Data1a, Data1b
Freq2, Data2a, Data2b
Freq3, Data3a, Data3b
.
.
Where

MA: Dataxa = Magnitude (Linear), Dataxb = Phase

DB: Dataxa = Magnitude (dB), Dataxb = Phase

RI: Dataxa = Real, Dataxb = Image

- If there is no Dataxb, Dataxb is taken as 0.
- If there is no MA/DB/RI, MA is selected.
- ! (comment) can be place at any lines (Not only top lines but also in the middle).
- If Freq n > Freq (n+1), then the data at freq (n+1) is ignored.
- If frequency span of setting on the VNA is wider than frequency range of data, the extrapolation is applied.

Error Messages

- File Name Error: is the same as VNA error. This message appears when an error exists in the file name and hence a command is not executed correctly.
- Failed To Read Error. This message appears when the file cannot be opened or if the file does not exist. It also occurs when the format of the file is incorrect or the number of data line is less than 2 or more than 10001.
Advanced Mode

- Overview
- Activating and Deactivating Advanced Mode

Other topics about Enhanced Time Domain Analysis
Overview

- Feature
- Advanced Mode Considerations

Other topics about Advanced Mode

Feature

The following table shows the comparison of Advanced and Basic Modes.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Feature</th>
<th>TDR Application (User Interface)</th>
<th>Hard Key on Front Panel</th>
<th>Soft Keys (right side of the screen)</th>
<th>Measurement Class Selection</th>
<th>Indicators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced</td>
<td>- Customized TDR/S-parameter measurements in TDR (Ch1)</td>
<td>Available</td>
<td>Available</td>
<td>Available but not recommended</td>
<td>TDR indicator: Yellow</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Such as:</td>
<td></td>
<td></td>
<td></td>
<td>In Instrument status bar</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- More Marker Functions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Limit Test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- TRL Calibration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Two channel measurements (Ch1 for TDR/Ch2 for</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6508
Basic | Easy to use | Available | Locked except for Trace Prev, Trace Next, Trace Max and keys in numeric key pad. | Hidden | Not Available | TDR indicator: Blue In Instrument status bar

1 Setting VNA-TDR to Advanced Mode enables VNA functions which are disabled in Basic Mode. It is possible to set up features that will result in unexpected measurement results. Therefore, only advanced users should use Advanced Mode.

The sample measurements are shown in the measurement examples.

**Advanced Mode Considerations**

In the Advanced Mode, you can access all VNA functions. The setting you changed may affect the measurement unexpectedly. The measurement may not be correct if you have such a case. Therefore, the measurement performance is not guaranteed in the Advanced Mode.

It is known that changing the following settings can cause incorrect measurement.
<table>
<thead>
<tr>
<th>Hard key</th>
<th>Do not use</th>
</tr>
</thead>
</table>
| Meas      | Single end 1 port: Parameters related with ports 2, 3 and 4  
|           | Single end 2 port/Differential 1 port: Parameters related with ports 3 and 4                                                                                                                            |
| Display   | Data Math, Equation Editor                                                                                                                                                                                |
| Sweep Setup | Start/Stop/Center/Span  
|           | Points  
|           | Sweep Type  
|           | Frequency Offset                                                                                                                                                                                            |
| Calibration | Fixture Simulator, AFR (If you want to use AFR, create a snp file in standard class, then use the embedding in adv waveform)                                                                              |
| Marker Function | Marker -> Start, Marker -> Stop                                                                                                                                                                          |

Here is another caution in the Advanced Mode.

<table>
<thead>
<tr>
<th>Function</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Meas</td>
<td>This may cause to tilt the wave form in the time domain. Use TDR GUI to select parameter.</td>
</tr>
<tr>
<td>Display</td>
<td>This may cause to tilt the wave form in the time domain.</td>
</tr>
<tr>
<td>Analysis</td>
<td>This may cause to tilt the wave form in the time domain.</td>
</tr>
<tr>
<td></td>
<td>The range of these settings is narrower than one of the Rise Time in the TDR GUI. Use TDR GUI to set the rise time.</td>
</tr>
<tr>
<td>Calibration</td>
<td>This may cause to exceed the DUT length limitation.</td>
</tr>
<tr>
<td>Scale</td>
<td>This may cause to exceed the DUT length limitation.</td>
</tr>
<tr>
<td>Ave</td>
<td>Even if you have narrow IF Bandwidth, you may not have noise reduction at lower frequency.</td>
</tr>
</tbody>
</table>
Activating and Deactivating Advanced Mode

- Activating Advanced Mode
- Deactivating Advanced Mode

Other topics about Advanced Mode

Activating Advanced Mode

1. Select the Setup tab.
2. Click Advanced Mode in the More Functions tab, then the Advanced Mode dialog is displayed.
3. When you want to perform a calibration such as TRL calibration in Advanced Mode, select the check box named Use Advanced Calibration Methods in dialog.
   - When you select the check box, the calibration you did in basic mode is cleared.
4. Click Yes to start the Advanced Mode.
5. The following settings are changed.
   - Softkeys are displayed on the right side of screen
   - All hard keys are unlocked.
   - SVC in the instrument status bar turns blue.

Deactivating Advanced Mode (Returning to Basic Mode)

1. Select the Setup tab.
2. Click Basic Mode button in the More Functions tab, then Advanced Mode dialog is displayed.
3. Click Yes to start the Basic Mode.
4. The application is re-started and all settings are reset.
Frequently Used Functions in Advanced Mode

- For Channel 1
- For Channel 2

Other topics about Advanced Mode

This section provides links to standard functions.

For Channel 1

As channel 1 is for VNA-TDR measurements, you can also use some of the standard VNA functions.

- **Cal** (Mechanical, TRL) (Only when Use Advanced Calibration Methods is checked)
- Marker Search
- Limit Test

For Channel 2

As the channel 2 can be used as one channel, you can use all of the standard functions.

- Measurement Parameter
  - Single-ended & Differential
- Format
- Scale
- Display Allocation
- IFBW
- Cal
- Stimulus
- Power
- NOP
- Sweep Type
- Segment Sweep

- Trigger
- Marker Search
- Limit Test
- Port Z conversion
- De-embedding
- Save/Recall
- Save data (csv, snp)
Measurement Examples

- TDR - PCB Impedance Measurement (Advanced Mode Example)
- 2 Channel Measurement Example (Advanced Mode Example)
This measurement example describes how to measure the characteristic impedance of a single-ended transmission line within a defined zone.

Ensure that the VNA-TDR application is in Advanced Mode in order to follow this example.

Before beginning the PCB impedance measurement, we need:

1. Set Measurement Conditions
2. Define Measurement Zone
3. Set Measurement Zone and Measure Characteristic Impedance

Set Measurement Conditions

1. Click Setup > Setup Wizard.
2. Select Deskew then click Next.
3. Select Single Ended 1-Port and click Next.
4. Follow the instructions then click Deskew. When finished, click Next.
5. Follow the instructions then click Measure. When finished, click Next.
Define Measurement Zone

Using VNA-TDR application GUI:

1. Disconnect the DUT.
2. Click Trace > 1 and double-click on the trace.
3. Click TDR/TDT > Parameters > Linear (under Format).
4. Click AutoScale > X & Y to adjust timebase for the entire response of the DUT is visible.
5. Click Marker > 1.

Using **Hardkey/SoftTab/Softkey** (on the right side of the screen):

6. Click Search > Target > Target Value.
7. Type 0.5 target value into the Target Search entry box then click Target Search. These steps locate the instant t1 on the TDR waveform where the open discontinuity occurs.
8. Connect the DUT.

Using VNA-TDR application GUI,

9. Click Marker > 2.

Using **Hardkey/SoftTab/Softkey** (on the right side of the screen),

10. Click Search > Target > Target Value.
11. Type 0.5 target value into the Target Search entry box then click Target Search. These steps locate the instant t2 on the TDR waveform where the open discontinuity occurs.

Calculate the measurement zone using the formulas below.

Compute the round trip propagation time of the transmission line:

\[ Trt = t2 - t1 = 4.96 \text{ ns} - 0 = 4.96 \text{ ns} \] (refer to the figure below)

Determine the initial instant, \( t_i \), of the measurement zone:
\[ ti = t_1 + 30\% Trt = 0 + (0.3) \times 4.96 \text{ ns} = 1.49 \text{ ns} \]

Determine the final instant, \( tf \), of the measurement zone:

\[ tf = t_1 + 70\% Trt = 0 + (0.7) \times 4.96 \text{ ns} = 3.47 \text{ ns} \]

Set Measurement Zone and Measure Characteristic Impedance

1. Click TDR/TDT > Parameters > Impedance (under Format).
2. Click TDR/TDT. Type in \( 25 \) into scale/div (left) entry box under Vertical.
3. Click TDR/TDT. Type in \(-75\) into position (right) entry box under Vertical.

Using Hardkey/SoftTab/Softkey (on the right side of the screen):

4. Click Search > Main > Domain then select User 1.
5. Click Search > Main > Domain Start. Type in 1.49 ns in the entry box.
6. Click Search > Main > Domain Stop. Type in 3.47 ns in the entry box.
7. Click Math > Analysis > Statistics.... The Trace Statistics dialog is displayed.
8. In the **Trace Statistics** dialog, select **User 1** from the drop down menu then click **OK**.

9. Click the left side (small button) of **Math > Analysis > Statistics...** to enable statistics and measure the characteristic impedance.

![Image of network analyzer with trace statistics dialog open]

The characteristic impedance is the mean value, 49.91 U.
2 Channel Measurement Example

- Overview
- Procedure

Other topics about Measurement Examples

Overview

This example shows a 2 channel measurement in the Advanced Mode.

2 channel measurements allow you to make the following measurements:

- TDR measurement on channel 1
- More customized S-parameter measurement on channel 2

**Note:** S-parameter measurements in Channel 1 are used for the time domain transformation. In order to ensure correct time domain results, the S-parameter settings for TDR are not accessible by the user. The optimum settings are calculated within the TDR algorithm. However, there are cases when the user may want specific settings for their S-parameter measurements. For these cases, another channel is used for customization (TDR works for Channel 1 only).

Here is the sequence of this example.

- Setup for channel 1
  1. Deskew
  2. Auto DUT length
  3. Set rise time
  4. Setup limit table for the trace 1

- Setup for channel 2
1. Setup start and stop frequency and IF Bandwidth.
2. Setup Sdd11 of Balance-Balance measurement
3. Setup limit table for the trace 1
4. Full 4-port calibration with ECal

- Measurement for channel 1
  1. Trigger
  2. Auto scale

- Measurement for channel 2
  1. Trigger

**Procedure**

**Preparation for 2 Channel Measurement**

1. Connect E-Cal on the USB port on the front panel.
2. Connect cables to all test ports.
3. Activate Advanced Mode (Do not select the **Use Advanced Calibration Methods** check box)
4. Click the **Stop Single** button to stop the trigger.

Using **Hardkey/SoftTab/Softkey** (on the right side of the screen):

1. Press **Trigger > Main > Trigger...** to access the Trigger dialog.
2. Under **Trigger Scope** select **Active Channel** then click **OK**.
3. Press **System > System Setup > Sound** then adjust the setting to 0 (zero) to turn off the beeper warning.

**Setup for Channel 1**

1. Click the **Setup** tab in the TDR GUI.
2. Click **Setup Wizard** (under **Basic**).

3. Set the measurement condition using the **Setup Wizard**:

   a. Select **Deskew** (under **Error Correction**) then click **Next >**.
   b. Click the **Differential 2-Port** button, then click **Next >**.
   c. Click the **Deskew** button, then click **Next >**.
   d. Click the **Measure** button, then click **Next >**.
   e. Set the **Rise Time** to **45 ps** and select **20-80%** from the **Definition** drop-down list.
   f. Click the **Finish** button.

**Limit Test Setup**

Using **Hardkey/SoftTab/Softkey** (on the right side of the screen):

1. Ensure that Trace 1 is selected.

2. Press **Math > Analysis > Limits...** to access the **Limit Test Setup** dialog.

3. Check **Limit Test ON** to turn on the limit test.

4. Check **Limit Line ON** to turn on the limit line then click **OK**.

5. Click **Math > Analysis > Limit Table** then select **Limit** to display the limit table.

6. Edit the table as shown below.

<table>
<thead>
<tr>
<th>Type</th>
<th>Begin Stimulus</th>
<th>End Stimulus</th>
<th>Begin Response</th>
<th>End Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Max</td>
<td>0 s</td>
<td>1 ns</td>
<td>105 U</td>
<td>105 u</td>
</tr>
<tr>
<td>2 Min</td>
<td>0 s</td>
<td>1 ns</td>
<td>75 U</td>
<td>75 U</td>
</tr>
</tbody>
</table>

**Setup for Channel 2**

1. Click **Trace > Trace Setup > Add Trace** then select **New Trace + Channel + Window**.

2. Click **Trigger > Main > Hold**.

3. Press the **Freq** hardkey. A table is displayed below the window.

   **Note:** When pressing the **Freq** hardkey, the table is shown in Channel 1 since the sweep is set to segment. For channel 2, the default would be linear sweep, so the necessary parameters will need to be set up using the softkeys.
4. Edit the table as shown below.

<table>
<thead>
<tr>
<th></th>
<th>State</th>
<th>Start</th>
<th>Stop</th>
<th>Points</th>
<th>IFBW</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>On</td>
<td>1 GHz</td>
<td>3 GHz</td>
<td>29</td>
<td>10 kHz</td>
<td>5 dBm</td>
</tr>
</tbody>
</table>

5. Press **Cal > Fixtures > Apply Fixtures ON** to turn on the fixture simulator.

6. Click **Setup > Layout > Measure...** to access the Measure dialog.

7. In the Measure dialog, select the Balanced tab.

8. Check the desired balanced measurement (for example, Sdd21), then click **Apply** to view the measurement result.

9. When finished, click **OK**.

**Full 4-port Calibration**

1. Connect 4 Port Ecal with the cables.

2. Press **Cal > Other Cals > Ecal...**.

3. Follow the wizard for a 4-Port ECal.

**Limit Test Setup**

Using **Hardkey/SoftTab/Softkey** (on the right side of the screen):

1. Press **Math > Analysis > Limits...** to access the Limit Test Setup dialog.

2. Check Limit Test ON to turn on the limit test.

3. Check Limit Line ON to turn on the limit line then click **OK**.

4. Click **Math > Analysis > Limit Table** then select Limit to display the limit table.

5. Edit the table as shown below.

<table>
<thead>
<tr>
<th></th>
<th>Begin Stimulus</th>
<th>End Stimulus</th>
<th>Begin Response</th>
<th>End Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Min</td>
<td>100 MHz</td>
<td>1.25 GHz</td>
<td>-1.5 dB</td>
</tr>
<tr>
<td>2</td>
<td>Min</td>
<td>1.25 GHz</td>
<td>2.5 GHz</td>
<td>-5 dB</td>
</tr>
<tr>
<td>3</td>
<td>Min</td>
<td>2.5 GHz</td>
<td>7.5 GHz</td>
<td>-7.5 dB</td>
</tr>
</tbody>
</table>
Measurement in Channel 1

1. Connect the DUT with the cables.

2. Click the **Stop Single** button in the TDR GUI to make one single measurement.

3. Select Trace 1.

4. Click **Auto Scale** in the TDR GUI, then select **All Traces**.

Measurement in Channel 2

1. Select the S-parameter trace of interest.

2. Click **Trigger > Main > Single** to make one single measurement.
Connecting with PathWave Vector Signal Analysis (89600 VSA)

Connecting to the 89600 VSA software allows for greater signal evaluation and troubleshooting.

Supported Models

- PXI VNA: M9800A to M9808A
- USB VNA: P5000A to P5008A, P5020A to P5028A
- PNA-B VNA

Requirements

- The 89600 VSA Version 2020 Update 0.1 or later is required.
- VNA
  - Firmware Version A.14.10.12 (A.14.60.xx for PNA-B VNA) and above
  - Option 190 Vector signal analysis hardware (PXI/USB VNA only)
  - S95090A/B (PXI VNA) or S97090A/B (USB VNA) or S93090xA/B (PNA-B VNA) Spectrum Analysis software

Limitations

- Negative Trigger Delay is not supported
  - Generally, VSA supports negative delay, in case of connecting with SA or Scope. But, VNA connection case, the negative delay is not supported because VNA trigger system hardware does not support negative delay. This is further follow-on feature.
- 100MHz Reference Input is not supported. (Only 10 MHz Reference Input is supported.)
  - The External Reference Frequency field is always grayed out in the reference source selection under Utilities > Frequency Reference.... 100MHz external input cannot be selected.
- User Cal for SA channel
SA channel is used for the VSA-VNA connection. When establish connection between VSA and VNA, VSA creates hidden SA channel on VNA. Then, VSA queries IQ data from the SA channel on VNA. During the connection, the user should not control the hidden VNA SA channel, otherwise, the VSA-VNA connection might be broken. VNA SA channel has the user cal function, but in the 89600 VSA-VNA connection, user cal on the hidden SA channel is not supported.

Connecting with the 89600 VSA Software

Before connection, make sure that the VSA Software and its license are installed.

VNA Application

1. Launch Network Analyzer application from Windows Start or Shortcut on the desk top.
2. Click \textit{System} > \textit{System Setup} > \textit{Remote Interface}.
3. Check the HiSlip.

IO Connection Expert

1. Execute IO Connection Expert from Windows Start.
2. Click \texttt{+Add}, then select \texttt{LAN instrument}.
3. Add LAN device dialog box is displayed.
4. Select \texttt{Enter Address} Tab.
5. Type your host name in Hostname or IP Address in Set LAN Address
6. Select \texttt{HiSlip} in Set Protocol
7. Type your remote name in Remote Name: (TCPIP0::<hostname>::hislip<address>)
8. Check \texttt{Allow *IDN Query}.
9. Click Test This VISA address, then confirm if your address is verified correctly.
10. Click OK.

**VSA Application**

1. Execute 89600 VSA application from Windows Start.

2. Select **Utilities > Hardware > Discovered Instruments...**

3. Open **SIM** option and make sure that the Vector Network Analyzer entry matching your hardware is displayed in the list. If not, check the VSA version.

4. Click Rediscover icon to find TCPIP0 connection, then the VNA connection will be displayed.

5. Select **Configurations** tab.

6. Click +, then select "Keysight Network Analyzer" and click ">" to register your VNA.
7. Select your VNA in “Current Analyzer Configuration” option. Then, VSA will connect with VNA to start measurements by VNA.
Low Frequency Extension (LFE)

- Overview
- Starting and Exiting LFE
- LFE Calibration
Overview

The 2-port and 4-port Low Frequency Extension (LFE) options add additional hardware to extend the start frequency of the VNA down to 900 Hz.

Note: Source and Receiver frequencies MUST BE EQUAL if both are in the LFE frequency band and LFE is turned on. If they are both in the LFE band and they are not equal, the following error message is displayed and the VNA will go into Hold mode:

In this topic:

- Model Compatibility
- LFE Block Diagram
- Supported Applications

Model Compatibility

The Low Frequency Extension options are available for the following models:

<table>
<thead>
<tr>
<th>PNA Model</th>
<th>2-Port</th>
<th>Model Option</th>
<th>4-Port</th>
<th>425</th>
<th>425+S93029A/B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>205</td>
<td>220</td>
<td>405</td>
<td>420</td>
<td>425</td>
</tr>
<tr>
<td>N5221B</td>
<td>v</td>
<td>v</td>
<td>v</td>
<td>v</td>
<td>v</td>
</tr>
<tr>
<td>N5222B</td>
<td>v</td>
<td>v</td>
<td>v</td>
<td>v</td>
<td>v</td>
</tr>
<tr>
<td>N5224B</td>
<td>v</td>
<td>v</td>
<td>v</td>
<td>v</td>
<td>v</td>
</tr>
<tr>
<td>N5225B</td>
<td>v</td>
<td>v</td>
<td>v</td>
<td>v</td>
<td>v</td>
</tr>
<tr>
<td>N5227B</td>
<td>v</td>
<td>v</td>
<td>v</td>
<td>v</td>
<td>v</td>
</tr>
<tr>
<td>N5241B</td>
<td>v</td>
<td></td>
<td>v</td>
<td>v</td>
<td>v</td>
</tr>
<tr>
<td>N5242B</td>
<td>v</td>
<td></td>
<td>v</td>
<td>v</td>
<td>v</td>
</tr>
<tr>
<td>N5244B</td>
<td>v</td>
<td></td>
<td>v</td>
<td>v</td>
<td>v</td>
</tr>
<tr>
<td>N5245B</td>
<td>v</td>
<td></td>
<td>v</td>
<td>v</td>
<td>v</td>
</tr>
<tr>
<td>N5247B</td>
<td>v</td>
<td></td>
<td>v</td>
<td>v</td>
<td>v</td>
</tr>
<tr>
<td>N5249B</td>
<td>v</td>
<td></td>
<td>v</td>
<td>v</td>
<td>v</td>
</tr>
</tbody>
</table>
LFE Block Diagram

The following LFE block diagram shows how the low-frequency hardware is configured for a single test port. The other ports are configured similarly.

![LFE Block Diagram](image)

Supported Applications

The following table shows which applications are supported/not supported by LFE.

<table>
<thead>
<tr>
<th>Application</th>
<th>Supported by LFE?</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gain Compression</td>
<td>Yes</td>
<td>S93086A/B</td>
</tr>
<tr>
<td>Gain Compression Converters</td>
<td>Yes</td>
<td>S93086A/B</td>
</tr>
<tr>
<td>Scalar Mixer/Converter</td>
<td>Yes</td>
<td>S93082A/B</td>
</tr>
<tr>
<td>Frequency Offset Mode</td>
<td>Yes</td>
<td>S93080A</td>
</tr>
<tr>
<td>Feature</td>
<td>Available</td>
<td>Model</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>Time Domain</td>
<td>Yes</td>
<td>S93010A/B</td>
</tr>
<tr>
<td>Dynamic Uncertainty Analysis</td>
<td>Yes</td>
<td>S93015A/B</td>
</tr>
<tr>
<td>Fast CW</td>
<td>Yes</td>
<td>S93118A/B</td>
</tr>
<tr>
<td>Auto Fixture Removal</td>
<td>Yes</td>
<td>S93007A/B</td>
</tr>
<tr>
<td>Scalar Mixer/Converter + Phase</td>
<td>No</td>
<td>S93083A/B</td>
</tr>
<tr>
<td>Vector Mixer/Converter</td>
<td>No</td>
<td>S93083A/B</td>
</tr>
<tr>
<td>Embedded LO</td>
<td>No</td>
<td>S93084A/B</td>
</tr>
<tr>
<td>Swept IMD</td>
<td>No</td>
<td>S93087A/B</td>
</tr>
<tr>
<td>Swept IMD Converters</td>
<td>No</td>
<td>S93087A/B</td>
</tr>
<tr>
<td>IM Spectrum</td>
<td>No</td>
<td>S93087A/B</td>
</tr>
<tr>
<td>IM Spectrum Converters</td>
<td>No</td>
<td>S93087A/B</td>
</tr>
<tr>
<td>True-Mode Stimulus Application (TMSA)</td>
<td>No</td>
<td>S93460A/B</td>
</tr>
<tr>
<td>Source Phase Control</td>
<td>No</td>
<td>S93088A/B</td>
</tr>
<tr>
<td>Multiport</td>
<td>No</td>
<td>S93551A/B</td>
</tr>
<tr>
<td>Differential IQ</td>
<td>No</td>
<td>S93089A/B</td>
</tr>
<tr>
<td>Spectrum Analyzer</td>
<td>No</td>
<td>S93090xA ,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>S9309xA</td>
</tr>
<tr>
<td>Pulsed RF</td>
<td>No</td>
<td>S93025A/B</td>
</tr>
<tr>
<td></td>
<td></td>
<td>with</td>
</tr>
<tr>
<td></td>
<td></td>
<td>021 and 022</td>
</tr>
<tr>
<td>Noise Figure</td>
<td>No</td>
<td>S93029A/B</td>
</tr>
<tr>
<td>Noise Figure Converters</td>
<td>No</td>
<td>S93029A/B</td>
</tr>
</tbody>
</table>
Starting and Exiting LFE

In this topic:

- Starting LFE
- Starting LFE with Auto-Enable
- Exiting LFE

Starting LFE

**How to start LFE**

**Using Hardkey/SoftTab/Softkey**

1. Press **Sweep > Source Control > LF Extension**.
2. Ensure that **ON** is selected in the **LF Extension** softkey.

**Using a mouse**

1. Click **Stimulus**
2. Select **Sweep**
3. Select **Source Control**
4. Select **LF Extension [on/OFF]**

Exiting LFE

1. On the VNA, press **Sweep > Source Control > LF Extension**.
2. Ensure that **OFF** is selected in the **LF Extension** softkey.
LFE Calibration

The procedure in this topic describes how to perform a calibration using the Cal All Calibration Wizard method, which allows multiple channel calibration in a single session.

**Note:** Beginning with the A.12.80 release, Cal All has been extended in order to deal with the new Low Frequency Extension option. If a user has a mixture of LFE and non-LFE channels and they would like to use Calibrate All to calibrate them at the same time, two calibration channels are created to account for the hardware differences between the two situations. When using the GUI or COM to set calibration and stimulus conditions, the settings are applied to both calibration channels. With SCPI, the user can query the primary guided calibration channel using `SYST:CAL:ALL:GUID:CHAN:VAL?`. This will return the primary calibration channel. When subsequent Guided Cal commands are used, settings will be transferred to the second calibration channel. If there is a desire to set these settings separately, the user should query for all Cal All Calibration channels with `SYST:CAL:ALL:GUID:CHAN:LIST?`. The user should set values for the primary calibration first, and then secondary calibrations. When initializing the calibration and acquiring steps, use the primary cal all channel number.

For information about other calibration methods that can be used, refer to the following links:

- Basic Cal
- Smart Cal
- Response Cal
- Source Power Cal

**Cal All Procedure**

1. Ensure that the measurement classes to calibrate are active.

2. Press **Sweep > Source Control > LF Extension ON.**

3. Ensure that the measurement frequency range is set properly.

4. Press **Cal > Main > Other Cals > Cal All...** to launch the following dialog showing the active measurement classes:
5. Check the measurement classes to calibrate.

6. Select the ports, then click on the **Apply to all channels** button to apply the port selections to all channels.

7. Click **Next**, then confirm or change the calibration properties in the **Measurement Class Cal Properties** dialog.

8. Click **Next** to access the **Calibration Attenuator Settings** dialog.

9. In the **Calibration Attenuator Settings** dialog, perform the following:

   a. Set the attenuator settings. Learn more.

   b. Click on the **Noise Reduction** button to improve measurement accuracy. Learn more.

   c. Click on the **Mechanical Devices** button to view all switches and attenuators in the VNA. Learn more.

10. Click **Next**, then select the DUT connectors and calibration kits in the **Select DUT Connectors and Cal Kits** dialog.

11. Click **Next** to access the **Power Cal Settings** dialog.
12. In the **Power Cal Settings** dialog, perform the following:

   a. Check **Use Multiple Sensors** if more than one power sensor is needed to cover the frequency range then select a sensor from the **Sensor** down menu.

   b. Learn more about **Accuracy Tolerance** and **Max Number of Readings**.

13. Click **Next** and follow the calibration process until completed.
Overview

The N5290A (900 Hz to 110 GHz) and N5291A (900 Hz to 120 GHz with over range to 125 GHz) are broadband network analyzer systems that utilize compact, broadband remote heads.

The standard system is comprised of the following components:

- N5222B, N5227B, N5242B, or N5247B VNA (see N5290A/N5291A configurations)
- N5292A Test Set
- N5293A (900 Hz to 110 GHz) or N5295A (900 Hz to 120 GHz) Frequency Extenders

In this topic:

- Features
- Capabilities
- N5293A/95A Frequency Extender LED States

See Also

Keysight Double-Ended Torque Wrench for 1.0 mm Connectors (internet connection required)

Features

- Power output of 0 dBm at 110 GHz
- Wide receiver and system dynamic range
- Excellent system stability
- Compact remote heads with 1 mm connectors
- Single module cable for easy setup
- Supports several of the existing VNA/PNA-X software applications at millimeter wave frequencies
- Available in 2-port and 4-port configurations
- All ports configured for use at millimeter wave frequencies
- Measurements down to 900 Hz
- Supports both single sweep Broadband and Banded waveguide measurements

Capabilities

- ONLY the applications listed in the Supported Applications table are supported
- Banded measurements require the N5292A test set
● No automatic power leveling. However, after performing an Installation Calibration, the source power at the output of the N5293A or N5295A Frequency Extender is adjustable.

● No DC power connector in N5292A test set to provide bias voltage to external frequency extenders for banded measurements

● Frequency extender modules cannot be HOT swapped while test set is powered on

● Applications that require the use of source and receiver attenuators DO NOT apply to a millimeter wave system

### N5293A/95A Frequency Extender LED States

The following is a list of possible LED states:

- **Blue**: Connected and addressed by N5292A Test Set
- **Green**: Standby
- **Orange**: Standby or warmup
- **Red**: Error condition
- **Off**: N5293A/95A is turned off

### See Also

- **N5290A/91A (N5292-90002) Millimeter Wave System Installation Guide** (requires internet connection) for information about front and rear panel interconnections.
- **N5290A/91A (N5292-90026) Millimeter System Service Guide** (requires internet connection)
- **Low Frequency Extension (LFE)**

### Other Related Topics
### Supported Applications

The following table shows which applications the N5290A/91A Millimeter system supports.

<table>
<thead>
<tr>
<th>Application</th>
<th>Supported N5290A/91A with LFE enabled</th>
<th>Supported N5290A/91A with LFE disabled</th>
<th>Supported in Banded Mode with Controller</th>
<th>Software Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gain Compression</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>S93086A/B</td>
</tr>
<tr>
<td>Gain Compression Converters</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>S93086A/B</td>
</tr>
<tr>
<td>Scalar Mixer/Converter (magnitude only)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>S93082A/B</td>
</tr>
<tr>
<td>Frequency Offset Mode</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>S93080A/B</td>
</tr>
<tr>
<td>Time Domain</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>S93010A/B</td>
</tr>
<tr>
<td>Dynamic Uncertainty Analysis</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>S93015A/B</td>
</tr>
<tr>
<td>Fast CW</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>S93118A/B</td>
</tr>
<tr>
<td>Auto Fixture Removal</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>S93007A/B</td>
</tr>
<tr>
<td>Source Phase Control</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>S93088A/B</td>
</tr>
<tr>
<td>Differential IQ</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>S93089A/B</td>
</tr>
<tr>
<td>Spectrum Analyzer</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>S93090xA/B, S9309xA/B</td>
</tr>
<tr>
<td>Pulsed RF</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>S93025A/B with 021 and 022</td>
</tr>
<tr>
<td>Noise Figure Converters</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>S93029A/B</td>
</tr>
<tr>
<td>Embedded LO</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>S93084A/B</td>
</tr>
<tr>
<td>Swept IMD</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>S93087A/B</td>
</tr>
<tr>
<td>Feature</td>
<td>Option A</td>
<td>Option B</td>
<td>Option C</td>
<td>Option Code</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>Swept IMD Converters</td>
<td>No</td>
<td>Yes²</td>
<td>No</td>
<td>S93087A/B</td>
</tr>
<tr>
<td>IM Spectrum</td>
<td>No</td>
<td>Yes²</td>
<td>Yes</td>
<td>S93087A/B</td>
</tr>
<tr>
<td>IM Spectrum Converters</td>
<td>No</td>
<td>Yes²</td>
<td>Yes</td>
<td>S93087A/B</td>
</tr>
<tr>
<td>True-Mode Stimulus Application (TMSA)</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>S93460A/B</td>
</tr>
<tr>
<td>Enhanced Time Domain Analysis</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>S93011A/B</td>
</tr>
<tr>
<td>Noise Figure</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>S93029A/B</td>
</tr>
<tr>
<td>Scalar Mixer/Converter + Phase</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>S93083A/B</td>
</tr>
<tr>
<td>Vector Mixer/Converter</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>S93083A/B</td>
</tr>
<tr>
<td>Multiport</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>S93551A/B</td>
</tr>
<tr>
<td>NVNA</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>S945xxA/B</td>
</tr>
</tbody>
</table>

1. Option S930909A/B 90 GHz spectrum analysis is supported on the N5290A only.
2. Swept IMD and IMDX operational to 24 GHz.
Supported Configurations

In addition to the standard N5290A and N5291A configurations, many combinations of hardware can be configured to create a Broadband single sweep millimeter wave system. The following table shows the supported hardware configurations using the N5292A Test Set Controller.

<table>
<thead>
<tr>
<th>VNA Model</th>
<th>Low Frequency Extension (Option205/425) Supported?</th>
<th>2-Port Test Set Controller (N5292A Option 200)</th>
<th>4-Port Test Set Controller (N5292A Option 400)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-Port N522xB/N524xB</td>
<td>N5222B Yes</td>
<td>Option 222 Interconnect kit for 2-port test set and 2-port VNA with 3.5 mm ports</td>
<td>Option 422 Interconnect kit for 4-port test set and 2-port VNA with 3.5 mm ports</td>
</tr>
<tr>
<td></td>
<td>N5242B Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N5224B No</td>
<td>Option 224 Interconnect kit for 2-port test set and 2-port VNA with 2.4 mm ports</td>
<td>Option 424 Interconnect kit for 4-port test set and 2-port VNA with 2.4 mm ports</td>
</tr>
<tr>
<td></td>
<td>N5225B No</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N5227B Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N5244B No</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N5245B Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N5247B Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4-Port N522xB/N524xB</td>
<td>N5222B Yes</td>
<td>Option 242 Interconnect kit for 2-port test set and 4-port VNA with 3.5 mm ports</td>
<td>Option 442 Interconnect kit for 4-port test set and 4-port VNA with 3.5 mm ports</td>
</tr>
<tr>
<td></td>
<td>N5242B Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N5224B No</td>
<td>Option 244</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N5241 No</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N5225B</td>
<td>N5227B</td>
<td>N5244B</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Interconnect kit for 2-port test set and 4-port VNA with 2.4 mm ports</td>
<td>Interconnect kit for 4-port test set and 4-port VNA with 2.4 mm ports</td>
<td></td>
</tr>
</tbody>
</table>
Starting and Exiting Millimeter Wave Mode

In this topic:

- Starting Millimeter Wave Mode
- Exiting Millimeter Wave Mode
- Starting and Exiting Low Frequency Extension (separate topic)

**Note:** The Low Frequency Extension (LFE) is disabled by default. LFE must be enabled to extend the low-end frequency range from 10 MHz down to 900 Hz.

### Starting Millimeter Wave Mode

#### How to start Millimeter Wave Mode

**Using Hardkey/SoftTab/Softkey**

1. Press **Setup > External Hardware > Millimeter Config.**
2. In the **Millimeter Configuration** dialog, select a banded or broadband configuration.
3. Configure the system then click **OK.**

**Using a mouse**

1. Click **Instrument**
2. Select **Setup**
3. Select **External Hardware**
4. Select **Millimeter Config...**
5. In the **Millimeter Configuration** dialog, select a banded or broadband configuration.
6. Configure the system then click **OK.**

### Exiting Millimeter Wave Mode
### How to exit Millimeter Wave Mode

#### Using Hardkey/SoftTab/Softkey

1. Press **Setup** > *External Hardware* > *Millimeter Config*.

2. In the *Millimeter Configuration* dialog, select **Standard PNA**.

3. Click **OK**.

#### Using a mouse

1. Click **Instrument**.

2. Select **Setup**.

3. Select **External Hardware**.

4. Select **Millimeter Config...**.

5. In the *Millimeter Configuration* dialog, select **Standard PNA**.

6. Click **OK**.
Millimeter Configuration

This topic describes the Millimeter Configuration dialog functions used to configure the N5290A/91A for broadband or banded measurements.

In this topic:

- Accessing Millimeter Configuration Dialog
- Broadband Millimeter Configuration Dialog
- Banded Millimeter Configuration Dialog
- Warning Messages

See Also

- N5290A/91A (N5292-90026) Millimeter System Service Guide (requires internet connection)
- Broadband System Measurement Setup
- Banded System Measurement Setup

Accessing Millimeter Configuration Dialog

- Press Setup > External Hardware > Millimeter Config.

Broadband Millimeter Configuration Dialog

<table>
<thead>
<tr>
<th>Programming Commands</th>
<th>Broadband Millimeter Configuration dialog box help</th>
</tr>
</thead>
</table>
Select Configuration  Lists the Standard VNA configuration, N529xA configuration, and other configurations that you have created.

- Click **New** to create a new Millimeter Module configuration. A name is automatically selected. Edit the Selected Configuration field to change the configuration name.

- Click **Remove** to delete the selected Millimeter Module Configuration.

- Select **N5290A Broadband** or **N5291A Broadband**

- For banded, define a frequency band.

- Select **Standard PNA** to exit the Millimeter mode to use the analyzer as a standard VNA.

**Properties**

**Name**  Shows the currently selected configuration. Edit this field to change the configuration name. Type a unique name using only alphanumeric and underscore characters.

**Test Set**  When **N5290A Broadband** or **N5291A Broadband** is the selected configuration, the N5292A test set is automatically selected. No other test set can be selected for the Broadband configuration.

**Enable Modules**  Enable or disable modules (frequency extenders) for the corresponding ports. The check boxes are disabled if a test set is attached and no module is detected on the corresponding port.
**Test Set IF... button**

![Test Set IF Switch settings for Channel N](image)

**Test Set IF Switch settings for Channel N**  Route the Reference or Test IF to the front panel (Front) or rear panel (Normal) of the test set. The Normal setting sets the IF bandwidth to 40 MHz. The Front setting sets the IF bandwidth to 2 GHz. Each port has a Reference IF and a Test IF switch. Each IF switch is independent per port and per channel. At preset, the default is Normal. IF switch settings are saved with the instrument state.

**Installation Cal...**  The installation calibration adjusts the source power at the output of the N5293A or N5295A Frequency Extender. It also adjusts for IF path loss variation. These can change each time a frequency extender is moved from one port to another, or from one system to another. This calibration should be done each time a change is made to the configuration. For a procedure, refer to Installation Cal.

**Frequencies**

**Port**  This is the port number of the VNA when in Standard PNA mode or the port number of the N5292A test set when in N529xA Broadband mode.

**Type**  The port type is set to Module when a frequency extender (N5293A or N5295A) is detected at the port. The Enable Modules checkbox will be checked for the port. When a frequency extender is not detected at the port, or the Enable Modules for the port is deselected, VNA is displayed to specify the standard VNA operation at the port.

**Start/Stop**  Displays the Start and Stop frequency of the frequency extenders (N5293A or N5295A) connected to each port and/or any VNA port in standard mode. Start/stop frequencies can be set by pressing Freq > Main > Start or Stop then entering the frequency.

**Cancel**  Closes dialog box without saving changes.

**OK**  Saves the configuration and the VNA is Preset before making the appropriate settings.

**About MM...**  Displays information about the current hardware configuration.
Banded Millimeter Configuration Dialog

Select Configuration  Lists the Standard VNA configuration and other configurations that you have created.

- Click **New** for first-time use.
- For the N529xA, select **N529xA Broadband**.
- For banded, define a frequency band.
Select **Standard PNA** to exit the Millimeter Configuration dialog.

**Name**  Shows the currently selected configuration. Edit this field to change the configuration name. Type a unique name using only alphanumeric characters and underscore.

**New**  Click to create a new Millimeter configuration. A name is automatically selected. Edit the Name field to change the configuration name.

**Remove**  Deletes a Millimeter Configuration.

**Properties**

**Test Set**  Select a test set to use in the current configuration. The firmware verifies that the selected test set is connected. If the selection does not match the connected test set, an error message is displayed. However, you are not prevented from continuing.

**Enable Modules**  Enable or disable modules (frequency extenders) for the corresponding ports. The check boxes are disabled if a test set is attached and no module is detected on the corresponding port.

**Module IF Gain**  When Auto is selected, the IF gain is automatically adjusted depending on the frequency of operation to avoid overdriving the receivers. In addition, you can manually select 0 dB, 2 dB, 4 dB, 6 dB, 8 dB, 10 dB, 11 dB, 13 dB, or 15 dB.

**Max Power Limit at Module RF IN**  The maximum mmWave module RF input is limited to this value when **Test Set RF ALC** is OFF. When you exit this dialog box using **OK**, set the power out of the VNA using the **Power and Attenuator dialog**.

**Offset**  Sets the loss of the cables. The mmWave module RF input is adjusted by this amount. Positive offset increases the power.

**Slope**  Helps compensate for cable and test fixture power losses at increased frequency. The mmWave module RF input power increases as the sweep frequency increases in dB/GHz. The slope is defined relative to the mmWave module RF input frequency. The slope starts at 0 Hz and a positive slope will increase the power level. Range is +/- 2 dB/GHz.

**Test Set IF...** button
**Test Set IF Switch settings for Channel N**  Route the Reference or Test IF to the front panel (Front) or rear panel (Normal) of the test set. The Normal setting sets the IF bandwidth to 40 MHz. The Front setting sets the IF bandwidth to 2 GHz. Each port has a Reference IF and a Test IF switch. Each IF switch is independent per port and per channel. At preset, the default is Normal. IF switch settings are saved with the instrument state.

Installation Cal... The installation calibration adjusts the source power at the output of the N5293A or N5295A Frequency Extender. It also adjusts for IF path loss variation. These can change each time a frequency extender is moved from one port to another, or from one system to another. This calibration should be done each time a change is made to the configuration. For a procedure, refer to Banded Installation Menu Calibration.

### Frequency Settings

**Multiplier RF IN**  RF Frequency Range (displayed in grey fields) multiplied by the multiplier value equals test port frequency range.

**Multiplier LO IN**  LO Frequency Range (displayed in grey field) multiplied by this value equals the test port frequency. The IF frequency is:

- PNA-X models = 7.438 MHz

**Test Port Frequency**  Set the Start and Stop frequencies of the selected configuration at the test ports. This becomes the displayed Start and Stop frequency of the VNA.

### Important Notes

- To set Test Port Frequency, first set the appropriate **Multiplier** values that are specified in your mmWave module documentation.
- Ensure that the RF and LO Frequencies (highlighted below) are within the frequency range of the sources. The VNA offers no warning if they are NOT.

<table>
<thead>
<tr>
<th>Multiplier RF IN</th>
<th>Multiplier LO IN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Frequency</td>
<td>Stop Frequency</td>
</tr>
<tr>
<td>12.5000000000 GHz</td>
<td>18.3333333333 GHz</td>
</tr>
<tr>
<td>9.3750000000 GHz</td>
<td>13.7500000000 GHz</td>
</tr>
</tbody>
</table>

### Source

Click a button to launch the **External Devices dialog** where you can select an internal or external source to be used for the VNA LO source or VNA RF source.

**Cancel**  Closes dialog box without saving changes.

**OK**  Saves the configuration and the **VNA is Preset** before making the appropriate settings.
About MM... Displays information about the current hardware configuration.

Warning Messages

The following dialog indicates that an N529xA Broadband configuration is active but some hardware is missing. It will only list the hardware that was not detected. In this case the test set and two selected heads were not found. The **Do not show this dialog again** check will suppress this dialog until the firmware is restarted.

This following dialog indicates that an N529xA Broadband configuration is active but some hardware is uncalibrated. It will only list the hardware that is uncalibrated but a full calibration should be executed if anything has changed. The **Do not show this dialog again** check will suppress this dialog until the firmware is restarted. For a calibration procedure, refer to Installation Cal.
The following millimeter hardware has changed:
- N5292A testset
- Port 1 millimeter head
- Port 2 millimeter head

Please press the button below to perform a millimeter system Installation Cal.

Do not show this dialog again.

Installation Cal... Cancel Help
Calibration

The procedure in this topic describes how to perform a calibration using the **Cal All Calibration Wizard** method, which allows multiple channel calibration in a single session.

Also described here is the **Installation Cal**, which should be done each time a change is made to the configuration, and the **Banded Installation Menu**, which is displayed only when a calibration is out of date or has not been performed.

Calibration procedures accessed in the user interface vary depending on which application (*Measurement Class*) is active. For information about other calibration methods and their purpose, refer to the following links:

- Basic Cal (separate topic)
- Smart Cal (separate topic)
- ECal (separate topic)
- Response Cal (separate topic)
- Source Power Cal (separate topic)
- Mixer Char Wizard (separate topic)
- Phase Reference Wizard (separate topic)
- Cal All Procedure

- Installation Cal
- Creating a Power Table
- Installation Calibration on a Banded Configuration

**Cal All Procedure**

A power table can be used instead of multiple power sensors to cover a wide frequency range. In addition, a power table can be used when the measurement frequency exceeds the frequency range of the power sensor (typically > 110 GHz). Refer to **Creating a Power Table** for information on how to create a power table.

1. Ensure that the measurement classes to calibrate are active.

2. Press **Cal > Main > Other Cals > Cal All...** to launch the following dialog showing the active measurement classes:
5. Check the measurement classes to calibrate.

6. Select the ports, then click on the **Apply to all channels** button to apply the port selections to all channels.

7. Click **Next**, then confirm or change the calibration properties in the **Measurement Class Cal Properties** dialog.

8. Click **Next** to access the **Calibration Attenuator Settings** dialog.

9. In the **Calibration Attenuator Settings** dialog, perform the following:

   a. Set the attenuator settings. **Learn more.**

   b. Click on the **Noise Reduction** button to improve measurement accuracy. **Learn more.**

   c. Click on the **Mechanical Devices** button to view all switches and attenuators in the VNA. **Learn more.**

10. Click **Next**, then select the DUT connectors and calibration kits in the **Select DUT Connectors and Cal Kits** dialog.

11. Click **Next** to access the **Power Cal Settings** dialog.
12. In the **Power Cal Settings** dialog, perform the following:

   a. Check **Use Multiple Sensors** if more than one power sensor is needed to cover the frequency range then select a sensor from the **Sensor** drop down menu.

   b. Otherwise, check **Use Multiple Sensors** then select the power table from the **Sensor** drop down menu. Learn how to create a power table.

   c. Learn more about **Accuracy Tolerance** and **Max Number of Readings**.

13. Click **Next** and follow the calibration process until completed.

**Installation Cal**

This calibration should be done each time a change is made to the configuration.

1. Press **Setup > External Hardware > Millimeter Config**.

2. Click **New** to create a new configuration or select one from the list under **Select Configuration**.
3. If desired, change the name shown in the **Name** field.

4. Click on the **Installation Cal...** button to launch the calibration. The following dialog is displayed.

![Millimeter Installation Cal](image1)

5. Connect a load to the port then click the **Measure** button.

![Millimeter Installation Cal](image2)
6. When finished, the following dialog is displayed. Remove the load then click the **Finish** button.

![Millimeter Installation Cal](image1)

7. The following dialog is displayed until the calibration is finished.

![Millimeter Installation Cal](image2)

**Creating a Power Table**

A power table is a text file with data that describes the output power of the module as a function of frequency. This is valid when the frequency extender module is driven at high levels (+11 dBm). This file may have been created for you by a third party or shipped with your frequency extender Module. If not, you can create this *.prn file from the manufacturer's specification for the frequency extender module.

This file can be created manually, using a text file program such as Notepad.

1. Copy the header information, and create the file with two columns, one for frequency and one for output power.
1.

Example .prn file

2. Name the power table file **pawertable1.prn**, where "1" corresponds to the port number. This file must be stored in the following directory on the VNA:

   C:\ProgramData\Keysight\Network Analyzer\Configurations\<configuration name>\n
   where *configuration name* is a directory name that is also the name of the currently selected configuration in the **Millimeter Configuration** dialog. For example, "N5291A Broadband".

3. Once the power table has been created and saved to the directory shown above, it will be listed in the **Sensor** drop down menu in the **Power Cal Settings** dialog where it may be selected instead of a power sensor.

---

**Installation Calibration on a Banded Configuration**

The following describes the process for the installation and calibration when using banded 3rd party frequency extenders with a PNA/PNA-X that is connected via a N5292A test controller.

This is to ensure that when the N5290/91A broadband is used with the N5290A304 1.8m cable adapter the system is optimized for the best power levels being applied to the RF and LO of the external modules. In addition, it will ensure that we have a calibrated IF channel for accurate raw receiver measurements that require this level of accuracy, like the Spectrum Analysis application.

Note that the above calibration and installation process optimized for use with supported VDI and OML frequency extenders. While other Frequency extenders maybe used, they should at minimum
comply to the requirements as follows:

1. Nominal RF saturated power requirement of 10 dBm +/- 4 dBm. (9.2GHz-21.3GHz).
2. Nominal LO saturated power requirement of 10 dBm +/- 3 dBm. (9.2GHz-21.3GHz).
3. The IF level should be optimized for an operation of 7.606 MHz and not exceed the -20 dBm.

**System Hardware**

In the description we will use the following hardware:

1. N5247B
2. N5292A Test Set Controller
3. WR-10 3rd party frequency extenders

**Required Calibration Hardware**

The process would require at least a 26.5 GHz power sensor and adapter to complete.

**Overall Process**

1. Connect the PNA-X, controller and attach the N5290A304 cable adapters
2. Do not connect the frequency extenders to the N5290304A cable adapters
3. Connect the power sensor to the PNA-X
4. Configure the power sensor for measurements
5. Use the millimeter Wave Configuration Dialog to configure the modules.
6. Select the installation menu to start the calibration of the system
7. Complete the calibration steps, for optimal performance all steps should be completed.

**Procedure**

The following set of descriptions are intended to walk through the individual steps needed to complete the calibration of the banded setup.

**Millimeter Wave Dialog**
1. Assign a name to the configuration. In this example we use “WR10 Module”.
2. Assign the ports on the N5292A to which the Frequency extenders will be connected.
3. Set the RF and LO Multipliers, 6 and 8 in this example.
4. Assign Test Set to be a N5292A and the ports for the modules to be connected.
5. Click the **Installation Cal...** button. The following dialog will appear.
The 3 Cals should run in the order shown: LO first, RF second, IF third.

LO Calibration

Before starting the LO calibration, disconnect the (Test Set to PNA) rear panel LO cable from the PNA. The Banded LO Cal requires a 3.5 mm Power Sensor to be connected to the PNA rear panel LO connector.

1. In the Banded Installation CAL window, click **Begin LO**. The following dialog box will appear.
2. Leave the system default settings and complete the LO calibration.
4. Connect the Power Sensor to the PNA Rear Panel LO as shown below.

5. Click **Begin** to run the LO Calibration. It will complete in less than a minute. When done, disconnect the
Power Sensor from the PNA rear panel and reconnect the Test Set LO Cable.

6. Optionally, you can run the LO CAL Verify by clicking on the **Verify** button.
   - The first (bottom) trace shows LO Power without the Cal applied.
   - The second (top) trace shows LO power after the Cal is applied.
Overlap is normal for frequencies less than 8 GHz and greater than 22 GHz.

RF Calibration

1. In the Banded Installation CAL window, click **Begin RF**. The following dialog box will appear.
2. Select the Power meter and complete the calibration by connecting the power sensor to the end of the N5290A304 adapter cable RF cable as shown below.

3. Click **Exit** when done with the RF Cal.
IF Calibration

1. Turn off the N5292A controller and connect up all the banded modules.
2. Turn on the controller and leave all ports open.
3. Connect all the LO and RF cables to the modules.
4. In the Banded Installation CAL window, click **Begin IF**. The following dialog box will appear.

5. Once complete the system will be optimized for the best settings to be used by the banded system configured.
Broadband System Measurement Setup

Broadband refers to mmWave configurations with a frequency range of 900 Hz to 110 GHz (N5290A) or 900 Hz to 120 GHz (N5291A). This configuration spans the entire frequency range in a single sweep.

After minimal setup shown below, the N5290A/91A functions the same as a standard VNA. All ports can be used at mmWave frequencies or a combination as required.

In this topic:

- Limitations
- Setting Up a Measurement

Limitations

- ONLY the applications listed in Supported Applications are supported
- No automatic power leveling. However, after performing an Installation Calibration, the source power at the output of the N5293A or N5295A Frequency Extender is adjusted.
- Simultaneous broadband and banded measurements NOT supported
- Applications that require the use of source and receiver attenuators DO NOT apply to a millimeter wave system

See Also

- Supported Applications
- Supported Configurations
- Low Frequency Extension (LFE)

Setting Up a Measurement

1. Press Setup > External Hardware > Millimeter Config.
2. Select N5291A Broadband (or N5290A Broadband).
3. Note the ports selected for **Enable Modules**. Frequency Extenders connected to the test set are automatically detected. Configure as necessary.

4. Press **OK**. This closes the dialog.

5. To change the start frequency below 10 MHz (minimum of 900 Hz), press **Freq > Main > Start** then enter the start frequency below 10 MHz. The following dialog is displayed.

6. Click **OK**.

7. To change the stop frequency, press **Freq > Main > Stop** then enter the stop frequency.
Banded System Measurement Setup

Banded refers to any configuration that is not a broadband configuration and is waveguide based. Frequency extenders cover frequency ranges from 50 GHz to 1.1 THz. The supported solutions can be configured for different frequency bands with the N5292A Test Set controller depending on the measurements required and the frequency extenders being used.

After minimal setup shown below, the N5290A/91A functions the same as a standard VNA. All ports can be used at mmWave frequencies or a combination as required.

The following table shows the waveguide designation equivalents.

<table>
<thead>
<tr>
<th>MIL-DTL-85/3C</th>
<th>IEEE Std 1785.1</th>
<th>Frequency Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>WR-15</td>
<td>WM-3759</td>
<td>50 GHz to 75 GHz</td>
</tr>
<tr>
<td>WR-12</td>
<td>WM-3099</td>
<td>60 GHz to 90 GHz</td>
</tr>
<tr>
<td>WR-10</td>
<td>WM-2540</td>
<td>75 GHz to 110 GHz</td>
</tr>
<tr>
<td>WR-08</td>
<td>WM-2032</td>
<td>90 GHz to 140 GHz</td>
</tr>
<tr>
<td>WR-06</td>
<td>WM-1651</td>
<td>110 GHz to 170 GHz</td>
</tr>
<tr>
<td>WR-05</td>
<td>WM-1295</td>
<td>140 GHz to 220 GHz</td>
</tr>
<tr>
<td>WR-04</td>
<td>WM-1092</td>
<td>170 GHz to 260 GHz</td>
</tr>
<tr>
<td>WR-03</td>
<td>WM-864</td>
<td>220 GHz to 330 GHz</td>
</tr>
<tr>
<td>WR-02</td>
<td>WM-570</td>
<td>330 GHz to 500 GHz</td>
</tr>
<tr>
<td>WR-1.5</td>
<td>WM-380</td>
<td>500 GHz to 750 GHz</td>
</tr>
<tr>
<td>WR-1.0</td>
<td>WM-250</td>
<td>750 GHz to 1.1 THz</td>
</tr>
</tbody>
</table>


Requirements

- N5292A Test Set
- DC Source to provide bias voltage to frequency extenders
- Option S93080A/B and 020
- Special adapter cable to connect between frequency extender and N5292A Test Set
- Source power calibration is required for ALL power related measurements with banded millimeter modules

Limitations
ONLY the applications listed in Supported Applications are supported

Banded measurements must use the N5292A Test Set

No DC power connector in N5292A Test Set to provide bias voltage to frequency extenders for banded measurements

No automatic power leveling. However, after performing an Installation Calibration, the source power at the output of the N5293A or N5295A Frequency Extender is adjustable.

Simultaneous broadband and banded measurements NOT supported

See Also

- Banded Millimeter Configuration Dialog
- Supported Applications
- Supported Configurations
- Low Frequency Extension (LFE)

Setting Up a Measurement

1. Press Setup > External Hardware > Millimeter Config.
2. Select a banded configuration.
3. Click **PNA RF Source** or **PNA LO Source** to launch the External Devices dialog where you can select an internal or external source to be used for the VNA RF source or VNA LO source.

4. Set the appropriate **Multiplier** values that are specified in your mmWave module documentation.

5. In the **Test Port Frequency** fields, set the **Start** and **Stop** frequencies.

6. Ensure that the RF and LO Frequencies are within the frequency range of the sources. The VNA offers no warning if they are NOT.

7. Press **OK**. This closes the dialog.
Mixer Measurement Configuration

Mixer measurements can be made at mmWave frequencies using SMC. (VMC measurements are NOT supported. See Supported Applications.)

In this topic:

- Requirements
- Limitations
- Hardware Connections
- Procedure
- Measuring Harmonic Mixers
- SMC Calibration

See Also

- SMC Measurements
- SMC mixer setup dialog
- Supported Applications

Requirements

- For 2-port VNAs, connect External Source to the VNA GPIB Controller port. Learn how to Configure an External Source.
- Connect the 10 MHz reference signal of an external source to the VNA

Limitations

- 2-port systems require external source for LO
- VMC measurements are NOT supported

Hardware Connections

Connections with a 4-port mmWave system

Upconverters

- DUT Input - Connect to Standard VNA port 2 or port 4.
- DUT LO - Connect mmWave module to test set Port 3.
- DUT Output - Connect mmWave module to test set Port 1.

**Downconverters**

- DUT Input - Connect mmWave module to test set Port 1.
- DUT LO - Connect mmWave module to test set Port 3.
- DUT Output - Connect to Standard VNA port 2 or port 4.

**Connections with a 2-port mmWave system**

**Upconverters** - requires a mmWave module as a **source** at the DUT LO and a mmWave module as a **receiver** at the DUT Output:

- DUT Input - Connect to Standard VNA port 2.
- DUT LO - Connect the RF cable of the mmWave module to an external source or the VNA (SRC2) second source.
- DUT Output - Connect the mmWave module to the test set port 1.

**Downconverters** - requires two mmWave modules as **sources**

- DUT Input - Connect the mmWave module to the test set port 1.
- DUT LO - Connect the RF cable of the mmWave module to an external source or the VNA (SRC2) second source.
- DUT Output - Connect to Standard VNA port 2.

**Procedure**

1. Connect your DUT to the mmWave system as described in Hardware Connections.
2. Configure this dialog (Millimeter Module Configuration).
3. For banded mixer measurements only, check **Mixer Mode**.
4. Press **OK**. This presets the VNA.
5. Create an SMC measurement:
a. Press **Setup > Main > Meas Class....**

b. Select **SMC**, then either:

   - **OK** delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.

c. An SC21 measurement is displayed.

4. **Make mixer settings** in the same way as with standard SMC measurements. Only two DUT ports can be swept in frequency. The remaining DUT port must be a fixed frequency. [See configuration used for harmonic mixers.](#)

5. **Increase power** for mmWave modules that are connected directly to a VNA port or external source.

6. Calibrate using the **SMC Calibration**.

---

**Measuring Harmonic Mixers**

Harmonic mixers have a multiplier circuit in the LO port of the DUT. Enter the multiplier value in the numerator of the X LO port in the **SMC mixer setup dialog**. This will provide the correct LO frequencies out of the appropriate source.

**SMC Calibration**

With a configured SMC measurement active, perform the **Cal All Procedure**.
Troubleshooting the VNA

By running a few checks, you can identify if the analyzer is at fault. Before calling Keysight Technologies or returning the instrument for service, please make the following checks.

- Check the Basics
- VNA Application Terminates Unexpectedly
- Check Error Terms
- Check the Service Guide
- Error Log

Other Support Topics

Check the Basics

A problem can often be solved by repeating the procedure you were following when the problem occurred. Before calling Keysight Technologies or returning the instrument for service, please make the following checks:

**Note:** Problems with the VNA application (slow or terminates unexpectedly) can be caused by a faulty Hard Disk Drive (HDD). For more information, see Preventing VNA SSD Problems and The VNA SSD Recovery Process.

1. Is there power at the power socket? Is the instrument plugged in?

2. Is the instrument turned on? Check to see if the front panel line switch and at least one of the LED rings around the test ports glows green. This indicates the power supply is on.

3. If you are experiencing difficulty with the front-panel keypad or peripherals, the USB bus may be overloaded. Remove the USB devices, restart the VNA, and reconnect the USB devices. See Power-up.

4. If other equipment, cables, and connectors are being used with the instrument, make sure they are connected properly and operating correctly.

5. Review the procedure for the measurement being performed when the problem appeared. Are all the settings correct?

6. If the instrument is not functioning as expected, return the unit to a known state by pressing the Preset key.
7. Is the measurement being performed, and the results that are expected, within the specifications and capabilities of the instrument?

8. If the problem is thought to be due to firmware, check to see if the instrument has the latest firmware before starting the troubleshooting procedure.

9. Check that the measurement calibration is valid. See Accurate Measurement Calibrations for more information.


11. **Phase lock lost message** - This usually occurs when there is not enough source power to phase lock the VNA. It can occur during an errant FCA setup or Source Power Calibration. It can also occur if one of the front panel reference channel loops is not connected. Otherwise, this indicates a hardware problem.

---

### VNA Application Terminates Unexpectedly

If an unexpected and irrecoverable error occurs, Keysight would like to know about it. The VNA attempts to save pertinent information about the state of the system. **The VNA does NOT send this information to Keysight.**

We respect the privacy of our customers. However, access to information that helps us improve the VNA is a benefit to both Keysight and you. Please take the time to contact us or email the saved information to www.keysight.com/find/contactus.

The following procedure shows how to do this:

1. A message box immediately appears on the screen containing the location of a directory. Please record this message. If you miss the message, you can find the directory location using the Windows Event Log: On the VNA, click Start, Settings, Control Panel, Administrative Tools, Event Viewer. Double-click the top line (most recent event). The location of the directory is seen in the Description.

2. A dialog box may appear on the screen allowing you to add comments to help us replicate the crash.

3. Find the directory (described in Step 1) which contains the following files:

   - **835x.dmp** which is the 835x.exe capturing the context in which the program crashed.
   - **835x.xml** which reports some very basic information (exception code, OS version, and the list of modules loaded at the time of the crash and their respective version numbers).
   - **835xCrashLog.txt**: The text file with your comments (described in Step 2), if submitted.

4. If your VNA is not connected to LAN or is not configured to send email, copy the files to a PC. Then, please...
Check Error Terms

If you print the error terms at set intervals (weekly, monthly, and so forth), you can compare current error terms to these records. A stable, repeatable system should generate repeatable error terms over long time intervals, for example, six months. If a subtle failure or mild performance problem is suspected, the magnitude of the error terms should be compared against values generated previously with the same instrument and calibration kit. See the procedure for monitoring error terms.

- A long-term trend often reflects drift, connector and cable wear, or gradual degradation, indicating the need for further investigation and preventative maintenance. Yet, the system may still conform to specifications. The cure is often as simple as cleaning and gaging connectors or inspecting cables.
- A sudden shift in error terms reflects a sudden shift in systematic errors, and may indicate the need for further troubleshooting.

Consider the following while troubleshooting:

- All parts of the system, including cables and calibration devices, can contribute to systematic errors and impact the error terms.
- Connectors must be clean and gauged, and within specification for error term analysis to be meaningful. See the Chapter 2 in the VNA Service Guide for information on cleaning and gaging connectors.
  - Avoid unnecessary bending and flexing of the cables following measurement calibration, thus minimizing cable instability errors.
  - Use good connection techniques during the measurement calibration. The connector interface must be repeatable. See the VNA Service Guide for information on connection techniques.
- It is often worthwhile to perform the procedure twice (using two distinct measurement calibrations) to establish the degree of repeatability. If the results do not seem repeatable, check all connectors and cables.
- Use error-term analysis to troubleshoot minor, subtle performance problems. See Chapter 3, "Troubleshooting," in the VNA Service Guide if a blatant failure or gross measurement error is evident.

Check the Service Guide

Check the VNA Service Guide for specific troubleshooting procedures to help identify problems. You can download a copy of the Service Guide from our Web site: www.keysight.com/manuals/pna

Error Log
Some VNAs create automatic log of data for troubleshooting purpose. The log file stores data related to the total power ON time, number of times of power ON, results of power ON test and so on. For security reasons, if this data needs to be deleted, then `SERVice:LOGGing:CLEar` command can be used to clear the log recorded by the instrument.
Analyzer Error Messages

- 500 - 750 Calibrate
- 770 - 1000 Hardware
- 1000 - 1200 Measure
- 1281 - 1535 Parser
- 1536 - 1650 Display
- 1700 - 2000 Channel
- 2001 - 3021 General
- Standard SCPI Errors

Note: The EventID's listed below are provided for COM programming. For more information, see Working with analyzer Events

See Also: About Error Messages

Memory Overflow Error

Memory overflow. Trigger state set to Hold. Lower the IF bandwidth, or increase dwell or sweep time.

Severity: Informational

Further explanation: The measurement that you are currently making requires that data be stored faster than it can be processed. Very few customers will experience this situation.

Suggestions: To limit the amount of data to be stored, try lowering the IF Bandwidth, slow the sweep time, increase the dwell time, or limit the number of data points. There are many other settings that can be adjusted to solve this problem.

EventID:

Cal Errors

Message: 512

"A secondary parameter (power, IFBW, sweep time, step mode)of the calibrated state has changed."
Severity: Informational

Further explanation: The calibration is questionable when any of these secondary parameters change after the calibration is performed.

Suggestions: If you require an accurate measurement with the new settings, repeat the calibration.

EventID: 68020200 (hex)

Message: 513
"Calibration cannot be completed until you have measured all the necessary standards for your selected Cal Type."

Severity: Informational

Further explanation: You probably received this message because you attempted to turn correction on without first measuring all of the calibration standards

Suggestions: Finish measuring the cal standards

EventID: 68020201 (hex)

Message: 514
"Calibration set has been recalled using a file previously saved on an analyzer that had a different hardware configuration."

Severity: Informational

Further explanation:

Suggestions:

EventID: 68020202 (hex)

Message: 515
"Calibration is required before correction can be turned on. Channel number is <x>, Measurement is <x>.

Severity: Informational

Further explanation: There are no error correction terms to apply for the specified channel and measurement.

Suggestions: Perform or recall a calibration
EventID: 68020203 (hex)

Message: 516

"Critical parameters in your current instrument state do not match the parameters for the calibration set, therefore correction has been turned off. The critical instrument state parameters are sweep type, start frequency, frequency span, and number of points."

Severity: Informational

Further explanation: None

Suggestions: You can either recalibrate using the new settings or change back to the original setting that was used when the calibration was performed.

EventID: 68020204 (hex)

Message: 517

"Interpolation is turned off and you have changed the stimulus settings of the original calibration, so correction has been turned off."

Severity: Informational

Further explanation: The most accurate calibration is maintained only when the original stimulus settings are used.

Suggestions: If reduced accuracy is OK, set interpolation ON to allow stimulus setting changes.

EventID: 68020205 (hex)

Message: 518

"Interpolation is turned off and you have selected correction ON. Correction has been restored with the previous stimulus settings."

Severity: Informational

Further explanation: None

Suggestions: None

EventID: 68020206 (hex)

Message: 519

"Stimulus settings for your current instrument state exceeded the parameters of the original calibration,"
so correction has been turned off."

Severity: Informational

Further explanation: Correction data outside the stimulus settings does not exist.

Suggestions: Perform a broadband calibration, with increased numbers of points with interpolation ON, to maintain calibration over the widest possible stimulus frequency settings.

EventID: 68020207 (hex)

Message: 520

"Cal Type is set to NONE for Channel <x>, Measurement <x>; please select Calibration menu or press Cal hard key."

Severity: Informational

Further explanation: A cal operation can not proceed until a calibration exists or the cal type is selected. This error can occur if the calibration can not be found. Also this error can happen if a calibration type is not specified before attempting to programmatically execute cal acquisitions.

Suggestions To find a calibration, select a Cal Set that contains the calibration needed for the current measurements. OR specify the cal type before beginning a calibration procedure.

EventID: 68020208 (hex)

Message: 521

"The measurement you set up does not have a corresponding calibration type, so correction has been turned off or is not permitted."

Severity: Informational

Further explanation: The calibration for the channel may apply only to certain S-Parameters. For example, a 1-Port calibration for S11 can not be applied to a 1-Port calibration applied to S22.

Suggestions: Select a calibration type, such as full 2-Port cal, that can be applied to all the measurements to be selected.

EventID: 68020209 (hex)

Message: 522

"The calibration type you selected cannot be set up."

Severity: Informational
Further explanation: "Please use the SCPI command ROUTe:PATH:DEFine:PORT <num>,<num> for full 2 port type port assignment."

Suggestions:

EventID: 6802020A (hex)

Message: 523

"The calibration path you selected cannot be set up because it is not valid for the current measurement."

Severity: Informational

Further explanation: "Please use the SCPI command ROUTe:PATH:DEFine:PORT <num>,<num> for full 2 port type port assignment related to your current measurement."

Suggestions:

EventID: 6802020B (hex)

Message: 524

"The source power calibration is complete."

Severity: Informational

Further explanation:

Suggestions:

EventID: 6802020C (hex)

Message: 525

"You have specified more than 7 standards for one or more calibration classes."

Severity: Informational

Further explanation: These have been truncated to 7 selections.

EventID: 6802020D (hex)

Message: 526

"No user calibration found for this channel."

Severity: Informational
**Further explanation:** A cal operation can not proceed until a calibration exists.

**Suggestions:** To find a calibration, you can select a Cal Set that contains the calibration needed for the current measurement.

**EventID:** 6802020E (hex)

**Message:** 527

"You do not need to acquire this standard for this calibration type."

**Severity:** Informational

Further explanation: This error can happen as a result of PROGRAMMATICALLY requesting the measurement of an un-needed calibration standard during a calibration procedure.

**Suggestions:** Check the specified cal type or eliminate the request for the measurement of the standard.

**EventID:** 6802020F (hex)

**Message:** 528

"Could not configure the Electronic Calibration system. Check to see if the module is plugged into the proper connector."

**Severity:** Informational

Further explanation: During an ECal operation, communication could not be established with the ECal module. The calibration will not be initiated until the presence of the ECal module is verified.

**Suggestions:** Verify the USB cable is connected properly. Disconnect and re-connect the cable to ensure the analyzer recognizes the module.

**EventID:** 68020210 (hex)

**Message:** 529

"DATA OUT OF RANGE: Design Limits Exceeded"

**Severity:** Error

**Further explanation:**

**Suggestions:**

**EventID:** E8020211(hex)

**Message:** 530
"EXECUTION ERROR: Could not open ECal module memory backup file"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020212 (hex)

Message: 531

"EXECUTION ERROR: Access to ECal module memory backup file was denied"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020213 (hex)

Message: 532

"EXECUTION ERROR: Failure in writing to ECal module memory backup file"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020214 (hex)

Message: 533

"EXECUTION ERROR: Failure in reading from ECal module memory backup file"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020215 (hex)

Message: 534
"EXECUTION ERROR: Array index out of range"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020216 (hex)

Message: 535

"EXECUTION ERROR: Arrays wrong rank"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020217 (hex)

Message: 536

"EXECUTION ERROR: CPU"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020218 (hex)

Message: 537

"EXECUTION ERROR: Cannot ERASE module"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020219 (hex)

Message: 538

"EXECUTION ERROR: Cannot WRITE module"
Severity: Error

Further explanation:

Suggestions:

EventID: E802021A (hex)

Message: 539

"EXECUTION ERROR: Entry Not Found"

Severity: Error

Further explanation:

Suggestions:

EventID: E802021B (hex)

Message: 540

"EXECUTION ERROR: Invalid command while system is busy"

Severity: Error

Further explanation:

Suggestions:

EventID: E802021C (hex)

Message: 541

"Electronic Cal: Unable to orient ECal module. Please ensure the module is connected to the necessary measurement ports."

Severity: Error

Further explanation: There is no RF connection to the ECal module during a calibration step. An ECal orientation measurement has been attempted but the signal was not found.

Suggestions: Connect the ECal module RF connections to ports specified for the calibration step. The ECal module typically requires at least -18dBm for measurements. If your measurement requires the power level to be less than that, clear the **Do orientation** checkbox to bypass the automatic detection step.
EventID: E802021D (hex)

Message: 542
"EXECUTION ERROR: NO SPACE for NEW CAL, DELETE A CAL"

Severity: Error

Further explanation:

Suggestions:

EventID: E802021E (hex)

Message: 543
"EXECUTION ERROR: No More Room"

Severity: Error

Further explanation:

Suggestions:

EventID: E802021F (hex)

Message: 544
"EXECUTION ERROR: Other array error"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020220 (hex)

Message: 545
"EXECUTION ERROR: Ranks not equal"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020221 (hex)
Message: 546
"EXECUTION ERROR: Too few CONSTANT ranks"
Severity: Error
EventID: E8020222 (hex)

Message: 547
"EXECUTION ERROR: Too few VARYing ranks"
Severity: Error
EventID: E8020223 (hex)

Message: 548
"EXECUTION ERROR: Unknown error"
Severity: Error
EventID: E8020224 (hex)

Message: 549
"EXECUTION ERROR: ecaldrvr.dll bug or invalid module #"
Severity: Error
EventID: E8020225 (hex)

Message: 550
"EXECUTION ERROR: unexpected error code from ecal driver"
Severity: Error
EventID: E8020226 (hex)

Message: 551
"EXECUTION ERROR: unexpected internal driver error"
Severity: Error
EventID: E8020227 (hex)
<table>
<thead>
<tr>
<th>Message: 552</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;HARDWARE ERROR: Can't access ECal Interface Module&quot;</td>
</tr>
<tr>
<td>Severity: Error</td>
</tr>
<tr>
<td>EventID: E8020228 (hex)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message: 553</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;HARDWARE ERROR: Can't release LPT port, reboot&quot;</td>
</tr>
<tr>
<td>Severity: Error</td>
</tr>
<tr>
<td>EventID: E8020229 (hex)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message: 554</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;HARDWARE ERROR: VNA Error&quot;</td>
</tr>
<tr>
<td>Severity: Error</td>
</tr>
<tr>
<td>EventID: E802022A (hex)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message: 555</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;HARDWARE ERROR: not enough data read from ECal module&quot;</td>
</tr>
<tr>
<td>Severity: Error</td>
</tr>
<tr>
<td>EventID: E802022B (hex)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message: 556</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;OPERATION ABORTED BY HOST COMPUTER&quot;</td>
</tr>
<tr>
<td>Severity: Error</td>
</tr>
<tr>
<td>EventID: E802022C (hex)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message: 557</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;OPERATION ABORTED BY USER&quot;</td>
</tr>
<tr>
<td>Severity: Error</td>
</tr>
<tr>
<td>EventID: E802022D (hex)</td>
</tr>
</tbody>
</table>
"OUT OF MEMORY"

Severity: Error

**EventID:** E802022E (hex)

**Message:** 559

"QUERY INTERRUPTED: Message(s Abandoned"

Severity: Error

**EventID:** E802022F (hex)

**Message:** 560

"QUERY UNTERMINATED: INCOMPLETE PROGRAM Message"

Severity: Error

**Further explanation:**

**Suggestions:**

**EventID:** E8020230 (hex)

**Message:** 561

"QUERY UNTERMINATED: NOTHING TO SAY"

Severity: Error

**Further explanation:**

**Suggestions:**

**EventID:** E8020231 (hex)

**Message:** 562

"QUEUE OVERFLOW"

Severity: Error

**EventID:** E8020232 (hex)

**Message:** 563
"SETTINGS CONFLICT: ADDITIONAL STANDARDS ARE NEEDED"

Severity: Error

EventID: E8020233 (hex)

Message: 564

"SETTINGS CONFLICT: Adapter Cal is NOT possible"

Severity: Error

EventID: E8020234 (hex)

Message: 565

"SETTINGS CONFLICT: COMMAND OUT OF SEQUENCE"

Severity: Error

EventID: E8020235 (hex)

Message: 566

"SETTINGS CONFLICT: Cal STOPPED - VNA SETUP CHANGED"

Severity: Error

EventID: E8020236 (hex)

Message: 567

"SETTINGS CONFLICT: Calibration is NOT in progress"

Severity: Error

EventID: E8020237 (hex)

Message: 568

"SETTINGS CONFLICT: Can't find specified GPIB board"

Severity: Error

EventID: E8020238 (hex)

Message: 569

"SETTINGS CONFLICT: Can't find/load gpib32.dll"
Severity: Error
EventID: E8020239 (hex)

Message: 570
"SETTINGS CONFLICT: Can't find/load sicl32.dll"

Severity: Error
EventID: E802023A (hex)

Message: 571
"SETTINGS CONFLICT: Can't initialize VNA (bad address?)"

Severity: Error
EventID: E802023B (hex)

Message: 572
"SETTINGS CONFLICT: Can't load LPT port driver or USB driver DLL"

Severity: Error
EventID: E802023C (hex)

Message: 573
"SETTINGS CONFLICT: Invalid Calibration Sweep Mode."

Severity: Error
EventID: E802023D (hex)

Message: 574
"SETTINGS CONFLICT: Invalid Calibration Type"

Severity: Error
EventID: E802023E (hex)

Message: 575
"SETTINGS CONFLICT: Invalid Calibration"
Severity: Error

EventID: E802023F (hex)

Message: 576
"SETTINGS CONFLICT: Invalid GPIB board number specified"

Severity: Error

EventID: E8020240 (hex)

Message: 577
"SETTINGS CONFLICT: Invalid GPIB board type specified"

Severity: Error

EventID: E8020241 (hex)

Message: 578
"SETTINGS CONFLICT: Invalid Module Status"

Severity: Error

EventID: E8020242 (hex)

Message: 579
"SETTINGS CONFLICT: Invalid States"

Severity: Error

EventID: E8020243 (hex)

Message: 580
"SETTINGS CONFLICT: LPT port must be between 1 and 4"

Severity: Error

EventID: E8020244 (hex)

Message: 581
"Could not configure the Electronic Calibration system. Check to see if the module is properly connected."
Severity: Error
EventID: E8020245 (hex)

Message: 582
"SETTINGS CONFLICT: Specified LPT port does not exist"

Severity: Error
EventID: E8020246 (hex)

Message: 583
"SETTINGS CONFLICT: Use frequency domain for cal"

Severity: Error
EventID: E8020247 (hex)

Message: 584
"SETTINGS CONFLICT: Use step sweep type for cal."

Severity: Error
EventID: E8020248 (hex)

Message: 585
"SETTINGS CONFLICT: VNA address must be between 0 and 30"

Severity: Error
EventID: E8020249 (hex)

Message: 586
"SETTINGS CONFLICT: Wrong LPT port driver or USB driver DLL"

Severity: Error
EventID: E802024A (hex)

Message: 587
"SYNTAX ERROR: ECAL:DELAY command must have 2 numbers"

Severity: Error
EventID: E802024B (hex)

Message: 588
"SYNTAX ERROR: INCORRECT SYNTAX"
Severity: Error

EventID: E802024C (hex)

Message: 589
"SYNTAX ERROR: UNKNOWN COMMAND"
Severity: Error

EventID: E802024D (hex)

Message: 590
"Wrong port of module in RF path"
Severity: Error

EventID: E802024E (hex)

Message: 591
"User characterization not found in module"
Severity: Error

EventID: E802024F (hex)

Message: 592
Severity: Informational
"No source power calibration found for the channel and source port of the current measurement."

Further explanation: You tried to turn on source power cal but there is no source power cal data.

Suggestions: Perform a source power calibration

EventID: 68020250 (hex)

Message: 593
Severity: Informational

"A source power calibration sweep was not performed, so there is no correction for the channel and source port of the current measurement."

Further explanation: You tried to turn on source power cal but there is incomplete source cal data.

Suggestions: Perform a complete source power calibration

EventID: 68020251 (hex)

Message: 594

Severity: Informational

"A new trace could not be added to the active window for viewing the source power cal sweep, because it would have exceeded the limit on number of traces/window. Please remove a trace from the window before proceeding with source power cal."

Further explanation: The source power cal attempts to add a data trace to the active window. The active window already contains four traces.

Suggestions: Make the active window contain less than trace limit.

EventID: 68020252 (hex)

Message: 595

Severity: Informational

"A new measurement could not be added for performing the source power cal sweep, because the limit on number of measurements has been reached. Please remove a measurement before proceeding with source power cal."

Further explanation: The source power cal attempts to add a measurement. The analyzer already has the maximum number of measurements.

Suggestions: Delete a measurement.

EventID: 68020253 (hex)

Message: 596

Severity: Informational

"The calibration power value associated with the source power calibration of Port %1 on Channel %2 was changed with the calibration on. The calibration was not turned off, but the power value might no
longer represent the calibration."

**Further explanation:** The source power cal accuracy is questionable.

**Suggestions:** If high accuracy is required, perform another source power calibration.

**EventID:** 68020254 (hex)

**Message:** 597

**Severity:** Informational

- Message that is passed from the power meter driver for a source power calibration. -

**Further explanation:** This error is generated by the power meter driver and passed through the analyzer.

**EventID:** 68020255 (hex)

**Message:** 598

"During the acquisition of the sliding load standard, the slide was not properly moved to perform a circle fit. The standard's raw impedance was used to determine the directivity for one or more points."

**Severity:** Informational

**Further Explanation:** To accurately characterize the standard, the sliding load must be move sufficiently to ensure enough samples around the complex circle or Smith Chart. Under-sampling will cause an inaccurate result.

**Suggestions:** For best results when using a sliding load, be sure to use multiple slide positions that cover the full range of movement from front to back of the slot.

**EventID:** 68020256 (hex)

**Message:** 599

"This feature requires an unused channel, but could not find one. Please free up a channel and try again."

**Severity:** Informational

**Further Explanation:** You attempted to view an item within a calset. However, the calset viewer requires that the result be displayed in a channel that is not currently in use. All the channels are currently used. The view can not display the requested item.

**Suggestions:** You must delete at least one channel that is currently in use.
EventID: 68020257 (hex)

Message: 600

"Interpolation of the original calibration is not allowed since it was performed using Segment Sweep. Correction has been turned off."

Severity: Informational

EventID: 68020258 (hex)

Message: 601

"Cal preferences saved. Cal preference settings can be changed from the 'Cal Preferences' drop down Cal menu."

Severity: Informational

EventID: 68020259 (hex)

Message: 608

"CalType not set."

Severity: Error

Further explanation: A cal operation can not proceed until a calibration exists or the proper cal type is selected.

Suggestions: This error can happen if the calibration can't be found. To find a calibration, you can select a Cal Set that contains the calibration needed for the current measurements. This error can also happen if a calibration type is not specified before attempting to programmatically execute cal acquisitions. Specify the cal type before beginning a calibration procedure.

EventID: E8020260 (hex)

Message: 609

"The Calibration feature requested is not implemented."

Further explanation: The specified cal type can be one of many choices. For example, response calibrations require single standards, 1-Port calibrations require 3 standards, and 2-Port calibrations require up to 12 standards.

Suggestions: Be sure to measure only the standards needed for the specified cal type.

EventID: E8020261 (hex)
**Message: 610**

"The Calibration Class Acquisition requested is not valid for the selected Calibration Type. Please select a different acquisition or a different Calibration Type."

**EventID:** E8020262 (hex)

**Message: 611**

"The Calibration Standard data required for the selected caltype was not found."

**Severity:** Error

**Further explanation:** An unsuccessful attempt was made to retrieve a specified standard from the raw measurement buffer. The buffer should contain the raw measurements of cal standards stored during a calibration procedure.

**Suggestions:** Be sure the requested standard is required for the current cal type. Not all standards are needed for all cal types.

**EventID:** E8020263 (hex)

**Message: 612**

" The Error Term data required for the selected caltype was not found."

**Severity:** Error

**Further explanation:** An unsuccessful attempt was made to retrieve a specified error term from the error correction buffer. The buffer should contain the error correction arrays for the current calibration.

**Suggestions:** Be sure the requested error term is required for the current cal type. Not all error terms are needed for all cal types.

**EventID:** E8020264 (hex)

**Message: 613**

The Calibration data set was not found.

**Severity:** Error

**Further explanation:** An unsuccessful attempt to access a cal set has been made. This may indicate a calset has been deleted or has been corrupted.

**Suggestions:** Try again or select another cal set. If the cal set appears in the cal set list, it may need to be deleted.
EventID: E8020265 (hex)

**Message: 614**

"The specified measurement does not have a calibration valid for Confidence Check. Please select a different measurement, or recall or perform a different Calibration Type."

**Severity:** Error

**Further explanation:** The measurement choice is prevented so that calibration will not be turned off. Not all cal types support all measurements. For example, an 1-Port cal on S11 can not be used to calibrate an S12 measurement. When a measurement is selected that does not have a calibration which can be applied, an informational message is displayed and calibration is turned off.

**Suggestions:** Use a full 2-Port calibration to be compatible with any S-Parameter.

EventID: E8020266 (hex)

**Message: 615**

" New calset created."

**Severity:** Informational message.

**Further explanation:** The newly created cal set will be automatically named and time stamped. If this is the beginning of a calibration procedure, the cal set will not be stored to memory until the calibration has completed successfully. The new cal set will be deleted if the calibration is canceled or does not otherwise complete successfully.

**Suggestions:** Informational

EventID: 68020267

**Message: 617**

The calset file: <x> appears to be corrupted and cannot be removed. Exit the application, remove the file, and restart.

**Severity:** Error

**Suggestions:** The cal set file is stored in the application home directory D:\analyzerCalSets.dat. Remove this file, then restart the application.
"The calset file: <x> load failed."

**Severity:** Error

**Further explanation:** The calset file contains a collection of calsets. The file resides on the hard drive.

**Suggestions:** Try restarting the application. If the failure persists, you may have to delete the cal set data file and restart the application. The cal set file is stored in the application home directory. D:\analyzerCalSets.dat. Remove this file, then restart the application.

**EventID:** E802027A (hex)

**Message:** 635

"The calset file: <x> save failed."

**Severity:** Error

**Further explanation:** The file operation detected an error. The save operation was aborted.

**Suggestions:** Retry.

**EventID:** E802027B (hex)

**Message:** 636

"A calset was deleted."

**Severity:** Informational

**Further explanation:** One of the calsets has been successfully deleted from the collection of calsets available. This can happen as the result of a user request or intentional operation.

**Suggestions:** None

**EventID:** 6802027C (hex)

**Message:** 637

"The version of the calset file: <x> is not compatible with the current instrument."

**Severity:** Error

**Further explanation:** A versioning error can prevent a calset from being used. This can happen as a result of instrument firmware upgrades.

**Suggestions:** If the versioning error is the result of firmware upgrade, you will have to re-install the old version of firmware to re-use the calset file. Or you can re-create the calsets with the current
version of firmware.

The cal set file is stored in the application home directory D:\analyzerCalSets.dat. Remove this file, then restart the application.

**EventID:** E802027D (hex)

**Message:** 638

"Incompatible CalSets found: \(<x>\) of \(<y>\) stored calsets have been loaded."

**Severity:** Error

**Further explanation:** Errors were found on some of the calsets stored in the calset file. The errors may have been caused by versioning issues that may have corrupted the various calset keys.

**Suggestions:** Use the calset viewer to look at the contents of calset files. Delete the files that are corrupted.

**EventID:** 6802027E (hex)

**Message:** 639

"The Calset file: \(<x>\) was not found. A new file has been created."

**Severity:** Informational

**Further explanation:** The calset file should be stored on the hard drive. When the application is started, a search is done and the file is loaded if it can be found. If the file is not found, the analyzer will create a new file and display this message.

**Suggestions:** None

**EventID:** 6802027F (hex)

**Message:** 640

"The Calset specified is currently in use."

**Severity:** Error

**Further explanation:** This may indicate a conflict between multiple calset users attempting calibration tasks.

**Suggestions:** Save the instrument state. Preset the analyzer and recall the instrument state. This may abort any processes that may be in progress.
EventID: E8020280 (hex)

Message: 641
"The calset specified has not been opened."

Severity: Error

Further explanation: Multiple users may be attempting to access the calset.

Suggestions: Close multiple calset users so that only one user will access the calset.

EventID: E8020281 (hex)

Message: 642
"The maximum number of cal sets has been reached. Delete old or unused cal sets before attempting to create new ones."

Severity: Error

Suggestions: You may also delete the calsets data file.

The cal set file is stored in the application home directory. D:\analyzerCalSets.dat. Remove this file, then restart the application.

EventID: E8020282 (hex)

Message: 643
The requested power loss table segment was not found.

Severity: Error

EventID: E8020283 (hex)

Message: 644
"A valid calibration is required before correction can be turned on."

Severity: Error

Further explanation: This usually indicates a calibration procedure has not run to completion or that the selected measurement does not have a valid calibration available from within the currently selected cal set.

Suggestions: To find a calibration, you can select a Cal Set that contains the calibration needed for the current measurements. This error can happen if a calibration type is not specified before attempting to
programmatically execute cal acquisitions. Specify the cal type before beginning a calibration procedure.

**EventID:** E8020284 (hex)

**Message: 645**

The cal data for <x> is incompatible and was not restored. Please recalibrate.

**Severity:** Warning

**Further explanation:** None

**Suggestions:** None

**EventID:** A8020285 (hex)

**Message: 646**

"CalSet not loaded, version is too new."

**Severity:** Error

**Further explanation:** An old version of firmware is attempting to run with a new calset version. The version is incompatible.

**Suggestions:** The calset can be removed. You may also delete the calsets data file if you are migrating between various firmware revisions often and you would like to avoid this error. The cal set file is stored in the application home directory. D:\analyzerCalSets.dat. Remove this file, then restart the application.

**EventID:** E8020286 (hex)

**Message: 647**

"Custom cal type not found."

**Severity:** Error

**Further explanation:**

**Suggestions:**

**EventID:** E8020287 (hex)

**Message: 648**

"Custom correction algorithm defers to the client for interpolation."
Severity: Informational

EventID: 68020288 (hex)

Message: 649
"Custom cal dll threw an exception."

Severity: Error

EventID: E8020289 (hex)

Message: 650
"Could not load the ecal.dll library"

Severity: Error

EventID: E802028A (hex)

Message: 656
"The argument specified is not a valid cal type."

Severity: Error

EventID: E8020290 (hex)

Message: 657
"The function found existing interpolated data"

Severity: Informational

EventID: 68020291 (hex)

Message: 658
"The function computed new interpolation values."

Severity: Informational

EventID: 68020292 (hex)

Message: 659
"The source power measurement failed."
Severity: Error

Suggestions: Please check GPIB, power meter settings and sensor connections.

EventID: E8020293 (hex)

Message: 660

"Duplicate session found. Close session and retry."

Severity: Error

EventID: E8020294 (hex)

Message: 661

"The session does not exist. Open the session and try again."

Severity: Error

Further explanation:

EventID: E8020295 (hex)

Message: 662

"Attempt to launch a custom calibration failed."

Severity: Error

Further explanation:

EventID: E8020296 (hex)

Message: 663

"Request to measure a cal standard failed."

Severity: Error

Further explanation: Please ensure you are requesting to measure standards which are defined for this calibration.

EventID: E8020297 (hex)

Message: 664

"Since Electronic Calibration Kit is selected, Mechanical Cal Kit parameter cannot be changed."
Severity: Error

Further explanation:

EventID: E8020298 (hex)

Message: 665

"Frequencies of the active channel are below minimum or above maximum frequencies of the ECal module factory characterization."

Suggestions: Change the channel frequencies, or select another ECal module.

Severity: Error

EventID: E8020299 (hex)

Message: 666

"Calset chosen for characterizing the ECal Module Ports %1 does not contain a calibration for analyzer Ports %2."

Severity: Error

Suggestions: Go back to select another calset or to perform another cal.

EventID: E802029A (hex)

Message: 667

"ECal module only has sufficient memory remaining to store a maximum of %1 points in User Characterization %2."

Severity: Error

Suggestions: Decrease your number of points, or choose to overwrite another user characterization.

EventID: E802029B (hex)

Message: 668

Input values are non-monotonic. Cannot interpolate.

Severity: Error

EventID: E802029C (hex)

Message: 669
Interpolation target is out of range. Cannot interpolate.

**Severity:** Error

**EventID:** E802029D (hex)

**Message:** 670

Guided Calibration Error: <>

**Severity:** Error

**EventID:** E802029E (hex)

**Message:** 671

The first call to the guided calibration interface must be Initialize.

**Severity:** Error

**EventID:** E802029F (hex)

**Message:** 672

The selected thru cal method was not recognized.

**Severity:** Error

**EventID:** E80202A0 (hex)

**Message:** 673

Could not generate the error terms.

**Severity:** Error

**EventID:** E80202A1 (hex)

**Message:** 674

Guided calibration must be performed on the active channel

**Severity:** Error

**EventID:** E80202A2 (hex)

**Message:** 675
You can not start using calibration steps until you have successfully called `generate steps`.

**Severity:** Error

**EventID:** E80202A3 (hex)

**Message:** 676

The step number given is out of range. Step numbers should be between 1 and the number of steps. 0 is not a valid step number.

**Severity:** Error

**EventID:** E80202A4 (hex)

**Message:** 677

A calset was selected for channel: `<n>` without restoring stimulus.

**Severity:** Informational

**EventID:** 680202A5 (hex)

**Message:** 678

A calset was selected for channel: `<n>` restoring stimulus.

**Severity:** Informational

**EventID:** 680202A6 (hex)

**Message:** 679

The selected calset stimulus could not be applied to the channel.

**Severity:** Informational

**EventID:** 680202A7 (hex)

**Message:** 680

You attempted to measure power at a frequency outside the frequency range defined for the specified power sensor. Select another sensor or adjust the range for this sensor.

**Severity:** Error

**EventID:** E80202A8 (hex)
Message: 681
Specified frequency is outside the frequency ranges currently defined for the power meter's sensors.
Severity: Error
EventID: E80202A9 (hex)

Message: 682
Additional Calibration Standards need to be acquired in order to calibrate over the entire frequency range currently being measured.
Severity: Informational
EventID: 680202AA (hex)

Message: 683
The analyzer failed to convert cal kits for use by unguided calibrations. The recommended action is to restore Cal Kit defaults.
Severity: Error
EventID: E80202AB (hex)

Message: 684
The analyzer failed to convert cal kits for use by unguided calibrations. CalKit defaults have been restored.
Severity: Error
EventID: E80202AC (hex)

Message: 685
Power meter is reserved by a source power cal acquisition already in progress.
Severity: Error
EventID: E80202AD (hex)

Message: 686
Source power calibration has not been performed or uploaded for the specified channel and source port.
Severity: Error
EventID: E80202AE (hex)

Message: 687
Source power calibration data array size for the specified channel and source port does not match its associated stimulus number of points.

Severity: Error

EventID: E80202AF (hex)

Message: 688
Source power calibration of Port <n> on Channel <n> was turned off because the correction array no longer exists.

Severity: Error

EventID: E80202B0 (hex)

Message: 689
This command can only be used on a measurement created with a specified calibration loadport.

Severity: Error

EventID: E80202B1 (hex)

Message: 690
Interpolation is turned off and you have changed the stimulus settings of the original calibration, so correction has been turned off.

Severity: Error

EventID: E80202B2 (hex)

Message: 691
Stimulus settings for your current instrument state exceeded the parameters of the original calibration, so correction has been turned off.

Severity: Error

EventID: E80202B3 (hex)

Message: 692
Fixturing: the requested S2P file cannot be read. Possible formatting problem.

**Severity:** Error

**EventID:** E80202B4 (hex)

**Message:** 693

Fixturing: the requested S2P file cannot be opened.

**Severity:** Error

**EventID:** E80202B5 (hex)

**Message:** 694

Fixturing: the requested S2P file cannot be interpolated. This is usually because the frequency range in the file is a subset of the current channel frequency range.

**Severity:** Error

**EventID:** E80202B6 (hex)

**Message:** 695

Cal Registers can only be used by one channel: the channel conveyed in the name of the cal register. The name cannot be changed.

**Severity:** Error

Further explanation: See [Cal Registers](#)

**EventID:** E80202B7 (hex)

**Message:** 696

Fixturing: cannot be enabled with Response Calibrations and has been turned off.

**Severity:** Error

**EventID:** E80202B8 (hex)

**Message:** 697

The selected calibration cannot be performed for this measurement.

**Severity:** Error
**EventID:** E8020B9 (hex)

**Message:** 698

Fitting: RemoveAllConnectors() should be called prior to calling AddConnector after a fit has been attempted.

**Severity:** Error

**EventID:** E8020BA (hex)

**Message:** 699

An attempt was made to acquire calibration data before the system was properly initialized.

**Severity:** Error

**EventID:** E8020BB (hex)

**Message:** 700

Use IGuidedCalibration for multiport calibration types.

**Severity:** Error

**EventID:** E8020BC (hex)

**Message:** 701

Guided calibration requires number of thru measurement paths be at least equal to the number of calibration ports minus 1.

**Severity:** Error

**EventID:** E8020BD (hex)

**Message:** 702

A thru path was specified that includes a port which the calibration was not specified to include.

**Severity:** Error

**EventID:** E8020BE (hex)

**Message:** 703

One or more of the ports to be calibrated was not found in the set of specified thru paths.
Severity: Error

EventID: E80202BF (hex)

---

**Hardware Errors**

Message: 770
Input power too high. Source power is off.

Severity: Warning

EventID: A8030302 (hex)

Message: 771
Source power restored.

Severity: Informational

EventID: 68030303 (hex)

Message: 772
"The spampnp.sys driver is not working. Check system hardware. ! Data will be simulated. !"

Severity: Error

**Further explanation:** The Network Analyzer application cannot locate the DSP board. Hardware or a driver may be malfunctioning. This is also common when attempting to run the Network Analyzer on a workstation.

EventID: E8030304 (hex)

Message: 773
"Instrument Serial Bus Not Working."

Severity: Error

**Further explanation:** The instrument EEPROM appears to contain either all ones or all zeros. A serial bus hardware failure prevents reading the EEPROM.

EventID: E8030305 (hex)

Message: 784
Unleveled, source 1, out 1.

**Severity:** Error

**Further explanation:** The analyzer was unable to set the power on port <n> to the desired level

**Message:** 785

Unleveled, source 1, out 2.

**Severity:** Error

**Further explanation:** The analyzer was unable to set the power on port <n> to the desired level

**Message:** 786

Unleveled, source 1 synthesizer.

**Severity:** Error

**Message:** 787

Unleveled, source 2, out 1.

**Severity:** Error

**Message:** 788

Unleveled, source 2, out 2.

**Severity:** Error

**Message:** 789

Unleveled, source 2 synthesizer.

**Severity:** Error

**Message:** 790

Unleveled, LO drive.

**Severity:** Error

**Message:** 791

Unleveled, LO synthesizer.

**Severity:** Error
Message: 792
Unlocked, source 1 synthesizer, integrator low.

Severity: Error

Message: 793
Unlocked, source 1 synthesizer, integrator high.

Severity: Error

Message: 795
Unlocked, source 2 synthesizer, integrator low.

Severity: Error

Message: 796
Unlocked, source 2 synthesizer, integrator high.

Severity: Error

Message: 798
Unlocked, LO synthesizer, integrator low.

Severity: Error

Message: 799
Unlocked, LO synthesizer, integrator high.

Severity: Error

Message: 801
Unleveled, doubler 1, prelevel.

Severity: Error

Message: 802
Unleveled, doubler 2, prelevel.

Severity: Error
Message: 803
Unleveled, doubler 3, prelevel.
Severity: Error

Message: 804
Unleveled, doubler 4, prelevel.
Severity: Error

Message: 805
Unleveled, source 1, P4.
Severity: Error

Message: 806
Unleveled, source 2, P4.
Severity: Error

Message: 848
"Phase lock lost"
Severity: Error

Further explanation: The instrument source was not able to lock properly. This can be the result of broken hardware, poor calibration, or bad EEPROM values.

Suggestions: Perform source calibration. Click System / Service / Adjustments / Source Calibration

EventID: E8030350 (hex)

Message: 849
Phaselock restored.
Severity: Success

EventID: 0x28030351 (hex)

Message: 850
"Unknown hardware error."
Severity: Error

Further explanation: Hardware malfunctioned prevents communication with the DSP.

EventID: E8030352 (hex)

Message: 851

DSP communication lost.

Severity: Error

EventID: E8030353 (hex)

Message: 852

RF power off.

Severity: Error

EventID: E8030354 (hex)

Message: 853

RF power on.

Severity: Success

EventID: 28030355 (hex)

Message: 854

Hardware OK.

Severity: Success

EventID: 28030356 (hex)

Message: 855

"Source unleveled."

Severity: Error

Further explanation: The source was unable to properly level at the requested power. The indicated power may not be accurate.

Suggestions: Try a different power level. Recalibrate source, if problem persists.
EventID: E8030357 (hex)

Message: 856
Source leveled.
Severity: Success

EventID: 28030358 (hex)

Message: 857
Input overloaded.
Severity: Error

EventID: E8030359 (hex)

Message: 858
Input no longer overloaded.
Severity: Success

EventID: 2803035A (hex)

Message: 859
"Yig calibration failed."
Severity: Error
Further explanation: Internal self-calibration of YIG oscillator tuning failed.

EventID: E803035B (hex)

Message: 860
Yig calibrated.
Severity: Success

EventID: 2803035C (hex)

Message: 861
"Analog ramp calibration failed."
Severity: Error

Further explanation: Internal analog sweep ramp calibration has failed.

EventID: E803035D (hex)

Message: 862

Analog ramp calibrated.

Severity: Success

EventID: 2803035E (hex)

Message: 864

Source temperature OK.

Severity: Success

EventID: 28030360 (hex)

Message: 865

"EEPROM write failed."

Severity: Error

Further explanation: Attempt to store calibration data to EEPROM has failed. There is a possible hardware failure.

EventID: E8030361 (hex)

Message: 866

EEPROM write succeeded.

Severity: Success

EventID: 28030362 (hex)

Message: 867

Attempted I/O write while port set to read only.

Severity: Error

Further explanation: Attempt to write to an I/O data port while the port set to input/read only.
Suggestions: Set data port to write/output before attempting to write to port.

EventID: E8030363 (hex)

Message: 868

" Attempted I/O read from write only port.

Severity: Error

Further explanation: Attempt to read from an I/O data port while the port set to output/write only.

Suggestions: Set data port to read/input before attempting to read from port.

EventID: E8030364 (hex)

Message: 869

Invalid hardware element identifier.

Severity: Error

EventID: E8030365 (hex)

Message: 870

Invalid gain level setting.

Severity: Error

EventID: E8030366 (hex)

Message: 871

Device driver was unable to allocate enough memory. Please try rebooting.

Severity: Error

EventID: E8030367 (hex)

Message: 872

DSP Error. Please Contact Keysight Support. Technical Information: DSP Type 1

Severity: Error

EventID: E8030368 (hex)
<table>
<thead>
<tr>
<th>Message: 873</th>
<th>DSP Error. Please Contact Keysight Support. Technical Information: DSP Type 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Severity: Error</td>
<td></td>
</tr>
<tr>
<td>EventID: E8030369 (hex)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message: 874</th>
<th>DSP Error. Please Contact Keysight Support. Technical Information: DSP Type 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Severity: Error</td>
<td></td>
</tr>
<tr>
<td>EventID: E803036A (hex)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message: 875</th>
<th>DSP Error. Please Contact Keysight Support. Technical Information: DSP Type 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Severity: Error</td>
<td></td>
</tr>
<tr>
<td>EventID: E803036B (hex)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message: 876</th>
<th>DSP Error. Please Contact Keysight Support. Technical Information: DSP Type 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Severity: Error</td>
<td></td>
</tr>
<tr>
<td>EventID: E803036C (hex)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message: 910</th>
<th>The trigger connection argument was not recognized as valid by the firmware.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Severity: Error</td>
<td></td>
</tr>
<tr>
<td>EventID: 0xE803038E (hex)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message: 911</th>
<th>The trigger connection specified does not support this trigger behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>Severity: Error</td>
<td></td>
</tr>
<tr>
<td>EventID: E803038F (hex)</td>
<td></td>
</tr>
</tbody>
</table>
The trigger behavior specified was not recognized as valid by the firmware.

**Severity:** Error

**EventID:** E8030390 (hex)

**Message:** 913

The trigger connection specified does not physically exist on this network analyzer

**Severity:** Error

**EventID:** E8030391 (hex)

**Message:** 914

Cannot set "Accept Trigger Before Armed", since this hardware configuration does not support edge triggering.

**Severity:** Error

**EventID:** E8030392 (hex)

**Message:** 915

Cannot set "Trigger Output Enabled", since this hardware configuration does not support BNC2.

**Severity:** Error

**EventID:** E8030393 (hex)

**Message:** 916

Exceeded maximum trigger delay.

**Severity:** Error

**EventID:** E8030394 (hex)

**Message:** 917

Exceeded minimum trigger delay.

**Severity:** Error

**EventID:** E8030395 (hex)
Measure Errors

Message: 1024
If you are going to display or otherwise use a memory trace, you must first store a data trace to memory.

Severity: Warning

EventID: A8040400 (hex)

Message: 1025
"The measurement failed to shut down properly. The application is in a corrupt state and should be shut down and restarted."

Severity: Error

Further explanation: This message is displayed if the analyzer application becomes corrupt. If you continue to get this error, please call customer service.

EventID: E8040401 (hex)

Message: 1026
The measurement failed to shut down properly. The update thread failed to exit properly.

Severity: Warning

EventID: A8040402 (hex)

Message: 1027
"Group Delay format with CW Time or Power sweeps produces invalid data."

Severity: Warning

Further explanation: Group Delay format is incompatible with single-frequency sweeps. Invalid data is produced.

Suggestions: Ignore the data or choose a different format or sweep type.

EventID: A8040403 (hex)

Message: 1028
Severity: Informational
"MSG_LIMIT_FAILED"

**Further explanation:** Limit line test failed.

**EventID:** 68040404 (hex)

**Message:** 1029

**Severity:** Informational

"MSG_LIMIT_PASSED"

**Further explanation:** Limit line test passed.

**EventID:** 68040405 (hex)

**Message:** 1030

"Exceeded the maximum number of measurements allowed."

**Severity:** Warning

**Further explanation:** See *Traces, Channels, and Windows on the analyzer* for learn about maximum measurements.

**EventID:** A8040406 (hex)

**Message:** 1031

"Network Analyzer Internal Error. Unexpected error in AddNewMeasurement."

**Severity:** Warning

**Further explanation:** If you continue to get this message, contact product support.

**EventID:** A8040407 (hex)

**Message:** 1032

"No measurement was found to perform the selected operation. Operation not completed."

**Severity:** Warning

**Further explanation:** None

**Suggestions:** Create a measurement before performing this operation.

**EventID:** A8040408 (hex)
Message: 1033
The Markers All Off command failed.

Severity: Warning

EventID: A8040409 (hex)

Message: 1034
"A memory trace has not been saved for the selected trace. Save a memory trace before attempting trace math operations."

Severity: Warning

Further explanation: Must have a memory trace when trying to do Trace Math,

EventID: A804040A (hex)

Message: 1035
"MSG_SET_AVERAGE_COMPLETE"

Severity: Informational

Further explanation: Informational for COM programming. Averaging factor has been reached.

EventID: 6804040B (hex)

Message: 1036
"MSG_CLEAR_AVERAGE_COMPLETE"

Further explanation: Informational for COM programming. Averaging factor has NOT been reached.

EventID: 6804040C (hex)

Message: 1037
"Time Domain transform requires at least 3 input points. The transform has been deactivated."

Severity: Informational

Further explanation: None

Suggestions: Increase the number of points.

EventID: 6804040D (hex)
Message: 1038
Smoothing requires a scalar format, and has been deactivated.
Severity: Informational
EventID: 6804040E (hex)

Message: 1039
A receiver power calibration in this instrument state file cannot be recalled into this firmware version.
Severity: Warning
EventID: A804040F (hex)

Message: 1047
Could not achieve target power.
Severity: Error

Further explanation: This indicates that the analyzer was unable to find a source power during the THRU step of the cal sufficiently high to boost the measured noise power on port 2 to 6 dB above the noise floor.

Message: 1056
ERROR: The given LO number is out of range. For a one stage mixer, this number must be 1. For a two stage mixer, this number can be 1 or 2.
Severity: Error

Further explanation: None

Message: 1063
"The trigger connection argument was not recognized as valid by the firmware."
Severity: Error

Further explanation: This indicates that the analyzer was unable to find a source power during the THRU step of the cal sufficiently high to boost the measured noise power on port 2 to 6 dB above the noise floor.

Message: 1073
"Unexpected error"
Severity: Error

Further explanation: None

Message: 1084
"User Preset was issued, but no user preset state had been set."

Severity: Error

Further explanation: None

Message: 1100
"Exceeded limit on number of measurements."

Severity: Error

Further explanation: See Traces, Channels, and Windows on the analyzer for measurement limits.

EventID: E8040450 (hex)

Message: 1104
"Exceeded limit on number of measurements."

Severity: Error

Further explanation: See Traces, Channels, and Windows on the analyzer for measurement limits.

EventID: E8040450 (hex)

Message: 1105
"Parameter not valid."

Severity: Error

Further explanation: A measurement parameter that was entered programmatically is not valid.

EventID: E8040451 (hex)

Message: 1106
"Measurement not found."

Severity: Error
Further explanation: Any of these could be the cause:

Trying to calibrate but already have maximum measurements.

Trying to do a confidence check but there is not a measurement.

Trying to create, activate, or alter a measurement through COM that has been deleted through the front panel.

Trying to use a trace name through programming that is not unique.

EventID: E8040452 (hex)

Message: 1107

"No valid memory trace."

Severity: Error

Further explanation: Must have a memory trace when trying to do Trace Math,

Suggestions: Store a memory trace.

EventID: E8040453 (hex)

Message: 1108

"The reference marker was not found."

Severity: Error

Further explanation: Attempted to create a delta marker without first creating a reference marker (COM only).

EventID: E8040454 (hex)

Message: 1109

"Data and Memory traces are no longer compatible. Trace Math has been turned off."

Severity: Error

Further explanation: Warning - channel setting has changed while doing trace math.

Suggestions: Store another memory trace and turn trace math back on.

EventID: A8040455 (hex)
Message: 1110

"Data and Memory traces are not compatible. For valid trace math operations, memory and data traces must have similar measurement conditions."

Severity: Error

Further explanation: Tried to do trace math without compatible data and memory traces.

Suggestions: Store another memory trace.

EventID: E8040456 (hex)

Message: 1111

"Marker Bandwidth not found."

Severity: Error

Further explanation: Could not find a portion of trace that meets the specified bandwidth criteria.

EventID: E8040457 (hex)

Message: 1112

"The peak was not found."

Severity: Error

Further explanation: Could not find portion of trace that meets peak criteria.

Suggestions: See Marker Peak criteria.

EventID: E8040458 (hex)

Message: 1113

"The target search value was not found."

Severity: Error

Further explanation: Could not find interpolated data point that meets search value.

EventID: E8040459 (hex)

Message: 1114

"Reflection measurement, such as S11, must supply an auxiliary port to disambiguate 2-port
measurements on multiport instruments.

**Severity:** Error

**Further explanation:**

**EventID:** E804045A (hex)

**Message:** 1115

"The receiver power calibration has been turned off because the type of measurement or source port has changed, so the calibration is no longer valid."

**Severity:** Warning

**Further explanation:**

**EventID:** A804045B (hex)

**Message:** 1116

"Receiver power cal requires the active measurement to be of unratioed power."

**Severity:** Warning

**Further explanation:**

**EventID:** A804045C (hex)

**Message:** 1117

"There is currently no source power calibration associated with the channel and source port of the active measurement. A source power cal should be performed or recalled before performing a receiver power calibration."

**Severity:** Warning

**Further explanation:**

**EventID:** A804045D (hex)

**Message:** 1118

"The attempted operation can only be performed on a standard measurement type."

**Severity:** Error

**Further explanation:**
EventID: E804045E (hex)

Message: 1119

"The custom measurement cannot be loaded because it is not compatible with the Network Analyzer hardware."

Severity: Error

Further explanation:

Suggestions:

EventID: E804045F (hex)

Message: 1120

"The custom measurement cannot be loaded because it is not compatible with the Network Analyzer software."

Severity: Error

Further explanation:

EventID: E8040460 (hex)

Message: 1121

"The custom measurement load operation failed for an unspecified reason."

Severity: Error

Further explanation:

EventID: E8040461 (hex)

Message: 1122

"The custom measurement data processing has generated an unhandled exception, and will be terminated. The analyzer software may be in an unstable state and it is recommended that the analyzer software be shutdown and restarted."

Severity: Error

Further explanation:

EventID: E8040462 (hex)
Message: 1123
"The attempted operation can only be performed on a custom measurement type."
Severity: Error
Further explanation:
EventID: E8040463 (hex)

Message: 1124
"The requested custom measurement is not available."
Severity: Error
Further explanation:
EventID: E8040464 (hex)

Message: 1125
"The requested custom algorithm was not found."
Severity: Error
Further explanation:
EventID: E8040465 (hex)

Message: 1126
"Normalization cannot be turned on because the measurement does not have a valid divisor buffer."
Severity: Error
Further explanation:
EventID: E8040466 (hex)

Message: 1127
"The Raw Data requested by the measurement could not be provided."
Severity: Warning
Further explanation:
EventID: A8040467 (hex)
Message: 1128

"The selected Sweep Type does not allow Transform and Gating. Transform and Gating disabled."

Severity: Error

Further explanation:

EventID: E8040468 (hex)

Message: 1129

"ERROR: There was not enough disk space to create the FIFO data file."

Severity: Error

Message: 1130

"ERROR: This feature is not available on this model of hardware."

Severity: Error

Message: 1131

"The data provided has an invalid number of points. It could not be stored."

Severity: Error

EventID: E804046B (hex)

Message: 1132

"The measurement stored in the save/recall state has an invalid version. It could not be loaded."

Severity: Error

EventID: E804046C (hex)

Message: 1133

"This data format argument for this operation must be "naDataFormat_Polar"

Severity: Error

EventID: E804046D (hex)

Message: 1134
This data format argument for this operation must be a scalar data format

**Severity:** Error

**EventID:** E804046E (hex)

**Message:** 1135

The memory trace is not valid for the current measurement setup.

**Severity:** Error

**EventID:** E804046F (hex)

**Message:** 1136

This measurement is incompatible with existing measurements in this channel. Choose another channel.

**Severity:** Error

**EventID:** E8040470 (hex)

**Message:** 1137

Port extension correction is not available for offset frequency measurements. Port extension correction has been disabled.

**Severity:** Error

**EventID:** E8040471 (hex)

**Message:** 1138

Physical port number assignments for logical port mappings must be unique.

**Severity:** Error

**EventID:** E8040472 (hex)

**Message:** 1140

"Power saturation back-off value was not found"

**Severity:** Error

**Further explanation:** None
Suggestions: None

Message: 1141

"Power normal operating point was not found"

Severity: Error

Further explanation: None

Suggestions: None

Message: 1147

"Specified external device was not found."

Severity: Error

Further explanation: None

Suggestions: None

Message: 1179

"Commas are not allowed."

Severity: Error

Further explanation: None

Suggestions: None

Parser Errors

Message: 1281

"You have sent a read command to the analyzer without first requesting data with an appropriate output command. The analyzer has no data in the output queue to satisfy the request."

Severity: Error

EventID: 68050501 (hex)

Message: 1282
"You must remove the active controller from the bus or the controller must relinquish the bus before the analyzer can assume the system controller mode."

**Severity:** Error

**EventID:** E8050502 (hex)

**Message:** 1283

"The analyzer did not receive a complete data transmission. This is usually caused by an interruption of the bus transaction."

**Severity:** Error

**EventID:** E8050503 (hex)

**Message:** 1284

"The instrument status byte has changed."

**Severity:** Informational

**EventID:** 68050504 (hex)

**Message:** 1285

"The SCPI command received has caused error number %1: "%2.""

**Severity:** Informational

**EventID:** 68050505 (hex)

**Message:** 1286

"The INET LAN server has been started as process number %1."

**Severity:** Informational

**EventID:** 68050506 (hex)

**Message:** 1360

"Execution of the SCPI command has failed"

**Severity:** Error

**EventID:** E8050550 (hex)
Message: 1361
"The INET/LAN device is not accessible."
Severity: Error
EventID: E8050551 (hex)

Message: 1362
"The INET/LAN driver was not found."
Severity: Error
EventID: E8050552 (hex)

Message: 1363
"The INET/LAN driver was not found."
Severity: Error
EventID: E8050553 (hex)

Message: 1364
"The INET/LAN device is unable to acquire the necessary resources."
Severity: Error
EventID: E8050554 (hex)

Message: 1365
"The INET/LAN device generated a generic system error."
Severity: Error
EventID: E8050555 (hex)

Message: 1366
"Invalid address for the INET/LAN device."
Severity: Error
EventID: E8050556 (hex)

Message: 1367
"The INET I/O library was not found."

Severity: Error

EventID: E8050557 (hex)

Message: 1368

"An error occurred in the INET system."

Severity: Error

EventID: E8050558 (hex)

Message: 1369

"Access to the INET/LAN driver was denied."

Severity: Error

EventID: E8050559 (hex)

Message: 1370

"Could not load error system message dll."

Severity: Error

EventID: E805055A (hex)

Message: 1371

"ErrorSystemMessage.dll does not export the right function."

Severity: Error

EventID: E805055B (hex)

Message: 1372

"Custom scpi library was not able to be knitted"

Severity: Error

EventID: E805055C (hex)

Message: 1373
"Could not knit the sepi error messages from the ErrorSystemMessage lib"

**Severity:** Error

**EventID:** E805055D (hex)

**Message:** 1374

Command is obsolete with this software version.

**Severity:** Error

**EventID:** E808055E (hex)

**Message:** 1375

CALC measurement selection set to none. Use **Calc:Par:Sel**

**Severity:** Error

**EventID:** E808055F (hex)

**Message:** 1535

"Parser got command: %1."

**Severity:** Informational

**EventID:** 680505FF (hex)

---

**Display Errors 1536 - 1621**

**Message:** 1536

"Exceeded the maximum number of traces in each window. The trace for <x> will not be added to window <x>."

**Severity:** Warning

**Further explanation:** None

**Suggestions:** Create the trace in another window. See the analyzer window limits.

**EventID:** A8060600 (hex)

**Message:** 1537
"Exceeded the maximum of 16 data windows. New window will not be created."

**Severity:** Warning

**Further explanation:** None

**Suggestions:** Create the trace in an existing window. See the analyzer window limits.

**EventID:** A8060601 (hex)

---

"No Data Windows are present. Unable to complete operation."

**Severity:** Warning

**Further explanation:** Your remote SCPI operation tried to create a new measurement while there were no windows present

**Suggestions:** Create a new window before creating the measurement. See example Create a measurement using SCPI

**EventID:** A8060602 (hex)

---

"No data traces are present in the selected window. Operation not completed."

**Severity:** Warning

**Further explanation:** None

**EventID:** A8060603 (hex)

---

"Cannot complete request to arrange existing measurements in <x> windows due to the limit of <x> traces per window."

**Severity:** Informational

**Further explanation:** The arrange window feature cannot put the existing traces into the number of windows you requested because the maximum number of traces per window has been exceeded. See Arranging Existing Measurements

**Suggestions:** Either create more windows or delete some traces.

**EventID:** 68060604 (hex)
| Message: 1541 |
| "Unable to establish a connection with the specified printer." |
| **Severity:** Warning |
| **Further explanation:** None |
| **Suggestions:** Refer to Printer Help |
| **EventID:** A8060605 (hex) |

| Message: 1542 |
| "Printout canceled." |
| **Severity:** Informational |
| **EventID:** 68060606 (hex) |

| Message: 1616 |
| "Window not found." |
| **Severity:** Error |
| **Further explanation:** A window was specified in your program which does not exist. |
| **Suggestions:** Query the name of your window before specifying. |
| **EventID:** E8060650 (hex) |

| Message: 1617 |
| "Duplicate window ID specified." |
| **Severity:** Error |
| **Further explanation:** None |
| **EventID:** E8060651 (hex) |

| Message: 1618 |
| "Exceeded limit on number of windows." |
| **Severity:** Error |
Further explanation: There is a limit of 4 windows per screen.

EventID: E8060652 (hex)

Message: 1619

"Exceeded limit on number of traces/window."

Severity: Error

Further explanation: There is a limit of traces per window. See the Traces, Channels, and Windows on the analyzer.

Suggestions: Create the trace in another window

EventID: E8060653 (hex)

Message: 1620

"Trace not found."

Severity: Error

Further explanation: Your program tried to communicate with a non-existing trace.

Suggestions: Query the trace ID before writing to it.

EventID: E8060654 (hex)

Message: 1621

"The operating system does not recognize this printer."

Severity: Warning

EventID: A8060655 (hex)

Message: 1622

Duplicate trace ID specified.

Severity: Error

EventID: E8060656 (hex)

Channel Errors 1792 -1878
| Message: 1792 |
| "Sweep Complete."
| **Severity:** Informational |
| **Further explanation:** None |
| **Suggestions:** None |
| **EventID:** 68070700 (hex) |

| Message: 1793 |
| "All triggerable acquisitions have completed."
| **Severity:** Informational |
| **Further explanation:** |
| **EventID:** 68070701 (hex) |

| Message: 1794 |
| "The last trigger produced an aborted sweep."
| **Severity:** Informational |
| **Further explanation:** |
| **EventID:** 68070702 (hex) |

| Message: 1795 |
| "The segment list must be adjusted to have at least one active segment with more than 0 points to use segment sweep."
| **Severity:** Informational |
| **Further explanation:** You attempted to change **Sweep type** to Segment sweep, but there is either no segments defined or no sweep points in the defined segments |
| **Suggestions:** Define at least one segment with at least one measurement point. See Segment sweep for more information |
| **EventID:** 68070703 (hex) |

| Message: 1796 |

6646
"MSG_SET_CHANNEL_DIRTY"

Severity: Informational

Further explanation: This informational message occurs when a channel setting has changed but the channel still has data that was taken with the previous setting. The following CLEAR message occurs when new channel data is taken.

EventID: 68070704 (hex)

Message: 1797

"MSG_CLEAR_CHANNEL_DIRTY"

Severity: Informational

Further explanation: The previous SET message occurs when a channel setting has changed but the channel still has data that was taken with the previous setting. This CLEAR message occurs when new channel data is taken.

EventID: 68070705 (hex)

Message: 1798

Sweep time has changed from Auto to Manual mode. If desired to return to Auto mode, enter sweep time value of 0.

Severity: Informational

EventID: 68070706 (hex)

Message: 1799

"Set Sweep Completed"

Severity: Informational

Further explanation: This event occurs when a sweep and it's associated sweep calculations finish. This is typically when all sweeps on a channel complete.

EventID: 68070707 (hex)

Message: 1800

"Clear Sweep Completed"

Severity: Informational
Further explanation: This event occurs immediately after the SET SWEEP COMPLETED event. These two events set and clear the "Sweep Completed" bit (bit 4) on the SCPI Device Status register.

**EventID:** 68070708 (hex)

**Message:** 1801

"All Sweeps Completed and Processed"

**Severity:** Informational

Further explanation: This event occurs when all of the sweeps and sweep calculations are complete for a channel.

**EventID:** 68070709 (hex)

**Message:** 1802

Low Pass : Frequency limits have been changed.

**Severity:** Informational

**EventID:** 6807070A (hex)

**Message:** 1803

Low Pass : Number of points have been changed.

**Severity:** Informational

**EventID:** 6807070B (hex)

**Message:** 1804

Low Pass : Frequency limits and number of points have been changed.

**Severity:** Informational

**EventID:** 6807070C (hex)

**Message:** 1805

"Channel created"

**Severity:** Informational

**EventID:** 6807070D (hex)
Message: 1806
"Channel deleted"
Severity: Informational
EventID: 6807070E (hex)

Message: 1872
"Channel not found."
Severity: Error
Further explanation: A non-existent channel is being referenced under program control.
Suggestions: Query the channel number, then refer to it by number.
EventID: E8070750 (hex)

Message: 1873
"The requested sweep segment was not found."
Severity: Error
Further explanation: A non-existent sweep segment is being referenced under program control.
EventID: E8070751 (hex)

Message: 1874
"The sweep segment list is empty."
Severity: Error
Further explanation: Segment Sweep cannot be specified unless there is at least one defined segment. This error will only occur under remote control.
EventID: E8070752 (hex)

Message: 1875
"The number of points in active sweep segment list segments is 0."
Severity: Error
Further explanation: Segment Sweep cannot be specified unless there is at least data point specified
in a segment. This error will only occur under remote control.

**EventID:** E8070753 (hex)

**Message:** 1876

"The specified source attenuator is not valid."

**Severity:** Error

**Further explanation:** You tried to set the Attenuator property on the Channel object on a analyzer that doesn't have a source attenuator.

**EventID:** E8070754 (hex)

**Message:** 1877

"Log Frequency sweep cannot be selected with the current Number of Points. Please reduce Number of Points."

**Severity:** Error

**Further explanation:** The maximum number of points that can be used for Log sweep is 401.

**EventID:** E8070755 (hex)

**Message:** 1878

"The requested Number of Points is greater than can be selected for Log Frequency sweep."

**Severity:** Error

**Further explanation:** The maximum number of points that can be used for Log sweep is 401.

**EventID:** E8070756 (hex)

**Message:** 1879

"Response frequencies exceeded instrument range so Frequency Offset has been turned off."

**Severity:** Error

**Further explanation:** This error is returned whenever the instrument detects that the stimulus sweep setup and Frequency Offset settings result in computed response frequencies that exceed instrument limits. When this occurs, the instrument automatically turns off Frequency Offset to avoid the out-of-range conditions.

**Suggestions:** When this condition has occurred, change settings for either the stimulus frequencies or
Frequency Offset so that the Response frequencies are within instrument bounds. Once this is done, Frequency Offset can once again be turned on.

**EventID:** E8070757 (hex)

**Message:** 1880

The total number of points for all the given segments exceeds the maximum number of points supported. The segments were not changed.

**Severity:** Error

**EventID:** E8070758 (hex)

**Message:** 1881

This instance of the Channels object was not used to place the channels in Hold, so no channels were resumed.

**Severity:** Error

**EventID:** E8070759 (hex)

**Message:** 1882

The port number was outside the range of allowed port numbers.

**Severity:** Error

**EventID:** E807075A (hex)

**Message:** 1883

More ports than are present are required for this operation.

**Severity:** Error

**EventID:** E807075B (hex)

---

**General Errors**

**Message:** 2009

"Channel is not a mixer channel."

**Severity:** Error
**Further explanation:** None

**Suggestions:** None

**Message: 2010**

"The external testset type is invalid."

**Severity:** Error

**Further explanation:** None

**Suggestions:** None

**Message: 2011**

"No ports were specified"

**Severity:** Error

**Further explanation:** None

**Suggestions:** None

**Message: 2012**

"Cannot couple primary domain range."

**Severity:** Error

**Further explanation:** None

**Suggestions:** None

**Message: 2015**

"The copy channel operation failed because the target channel exists and is incompatible with the source channel."

**Severity:** Error

**Further explanation:** None

**Suggestions:** None

**Message: 2019**

"Invalid sweep number."
Severity: Error

Further explanation: None

Suggestions: None

Message: 2020

"IMD channel tracking not enabled."

Severity: Error

Further explanation: None

Suggestions: None

Message: 2022

"Compression analysis not enabled due to invalid parameter(s)"

Severity: Error

Further explanation: None

Suggestions: None

Message: 2023

"External receiver configuration not available."

Severity: Error

Further explanation: None

Suggestions: None

Message: 2025

"Error term data not found."

Severity: Error

Further explanation: None

Suggestions: None

Message: 2028

"The PNA platform does not support ON mode for virtual bridge. The only supported modes are
AUTO and OFF."

Severity: Error

Further explanation: None

Suggestions: None

**Message: 2048**

"The function you requested requires a capability provided by an option to the standard analyzer. That option is not currently installed."

Severity: Error

Further explanation: None

Suggestions: To view the options on your analyzer, click Help / About Network Analyzer. For more information see analyzer Options

EventID: 68080800 (hex)

**Message: 2049**

"The feature you requested is not available on the current instrument."

Severity: Error

Further explanation: None

EventID: 68080801 (hex)

**Message: 2050**

"The feature you requested is incompatible with the current instrument state."

Severity: Error

Further explanation: None

Suggestions: None

EventID: 68080802 (hex)

**Message: 2051**

"File<file> has been saved."
<table>
<thead>
<tr>
<th>EventID</th>
<th>Message</th>
<th>Severity</th>
<th>Further explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>68080803 (hex)</td>
<td>&quot;Attempt to save &lt;x&gt; failed.&quot;</td>
<td>Error</td>
<td>None</td>
</tr>
<tr>
<td>E8080804 (hex)</td>
<td>&quot;Attempt to recall file failed because &lt;x&gt; was not found.&quot;</td>
<td>Error</td>
<td>None</td>
</tr>
<tr>
<td>E8080805 (hex)</td>
<td>&quot;&lt;x&gt; has a bad header.&quot;</td>
<td>Error</td>
<td>None</td>
</tr>
<tr>
<td>E8080806 (hex)</td>
<td>&quot;Request to enter hibernate state.&quot;</td>
<td>Error</td>
<td>None</td>
</tr>
<tr>
<td>68080808 (hex)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Message: 2057**

"Power up from automatic hibernate state. Program received PBT_APMRESUME AUTOMATIC Message."

**Further explanation:** None

**EventID:** 68080809 (hex)

---

**Message: 2058**

"Power up from suspend hibernate state. Program received PBT_APMRESUMESUSPEND Message."

**Further explanation:** None

**EventID:** 6808080A (hex)

---

**Message: 2059**

"Power up from suspend hibernate state. Program received PBT_APMRESUMECRITICAL Message."

**Severity:** Warning

**Further explanation:** None

**EventID:** A808080B (hex)

---

**Message: 2060**

"Power up from unknown hibernate state UI recovery called. Program received no PBT_Message within the time allotted and is attempting recovery."

**Severity:** Warning

**Further explanation:** None

**EventID:** A808080C (hex)

---

**Message: 2061**

"<x> already exists. File is being overwritten."

**Further explanation:** Used only for remote applications

**EventID:** 6808080D (hex)

---

**Message: 2062**
"File has not been saved."

Severity: Error

Further explanation: Used only for remote applications

EventID: E808080E (hex)

Message: 2063

"File <x> has been recalled."

Further explanation: Used only for remote applications

EventID: 6808080F (hex)

Message: 2064

"State version in <x> is considered obsolete by this version of this code."

Severity: Error

Further explanation: You attempted to recall a file that is no longer valid.

Suggestions: You must recreate the file manually.

EventID: E8080810 (hex)

Message: 2065

"State version in <x> is newer than the latest version supported by this code."

Severity: Error

Further explanation: You attempted to recall a file that was created by a later version of the analyzer application.

Suggestions: You must recreate the file manually.

EventID: E8080811 (hex)

Message: 2066

"Error occurred while reading file <x>"

Severity: Error

Further explanation: The file may be corrupt.
Suggestions: Try to recreate the file.

**EventID:** E8080812 (hex)

**Message:** 2067

"Windows shell error: <x>"

**Severity:** Error

**Further explanation:** None

**EventID:** E8080813 (hex)

**Message:** 2068

Send message timed out returning: <x>.

**Severity:** Error

**Further explanation:** None

**EventID:** E8080814 (hex)

**Message:** 2069

"Changing GPIB mode to System Controller."

**Severity:** Informational

**Further explanation:** None

**EventID:** 68080815 (hex)

**Message:** 2070

"Changing GPIB mode to Talker Listener."

**Severity:** Informational

**Further explanation:** None

**EventID:** 68080816 (hex)

**Message:** 2071

"The Network Analyzer can not be put in GPIB System Controller mode until the GPIB status is Local. Stop any remote GPIB programs which may be using the Network analyzer, press the Macro/Local key
and try again."

**Severity:** Informational

**Further explanation:** See LCL and RMT Operation

**Suggestions:** Press the Macro/Local key and try again.

**EventID:** 68080817 (hex)

---

**Message:** 2120

"This method can not be invoked through a late-bound COM call."

**Severity:** Error

**Further explanation:** None

**Suggestions:** Use the alternate method described in the COM programming documentation

**EventID:** E8080878 (hex)

---

**Message:** 2128

"The specified format is invalid."

**Severity:** Error

**Further explanation:** None

**EventID:** E8080850 (hex)

---

**Message:** 2129

"WINNT exception caught by Automation layer."

**Severity:** Error

**Further explanation:** None

**EventID:** E8080851 (hex)

---

**Message:** 2130

"Bad port specification."

**Severity:** Error

**Further explanation:** None
**EventID:** E8080852 (hex)

**Message:** 2131

"Failed to find a printer."

**Severity:** Error

**Further explanation:** None

**Suggestions:** See Connecting to a Printer

**EventID:** E8080853 (hex)

**Message:** 2132

"Manual trigger ignored."

**Severity:** Error

**Further explanation:** None

**EventID:** E8080854 (hex)

**Message:** 2133

"Attempt to set trigger failed."

**Severity:** Error

**Further explanation:** None

**EventID:** E8080855 (hex)

**Message:** 2134

"Macro execution failed."

**Severity:** Error

**Further explanation:** None

**EventID:** E8080856 (hex)

**Message:** 2135

"Specified macro definition is incomplete."
Severity: Error

Further explanation:

EventID: E808057 (hex)

Message: 2137

"Block data length error."

Severity: Error

Further explanation: See Getting Data from the Analyzer

EventID: E808059 (hex)

Message: 2139

"Requested data not found."

Severity: Error

Further explanation: None

EventID: E80805B (hex)

Message: 2142

"The parameter supplied was out of range, so was limited to a value in range before being applied to the instrument."

Severity: Success

Further explanation: None

Suggestions: View range limits before sending programming commands.

EventID: 280805E (hex)

Message: 2143

The parameter supplied was out of range, so was limited to a value in range before being applied to the instrument.

Severity: Error

EventID: E80805F (hex)
| Message: 2144 |
| "Request failed. The required license was not found." |
| **Severity:** Error |
| **Further explanation:** None |
| **EventID:** E8080860 (hex) |

| Message: 2145 |
| "A remote call to the front panel has returned hresult <x>" |
| **Severity:** Error |
| **Further explanation:** This may indicate a problem with the front panel |
| **Suggestions:** Contact Technical support |
| **EventID:** E8080861 (hex) |

| Message: 2146 |
| The recall operation failed. |
| **Severity:** Error |
| **Further explanation:** |
| **EventID:** E8080862 (hex) |

| Message: 2147 |
| Attempt to save file failed. |
| **Severity:** Error |
| **Further explanation:** |
| **EventID:** E8080863 (hex) |

| Message: 2148 |
| Recall attempt failed because file was not found. |
| **Severity:** Error |
| **Further explanation:** |
EventID: E8080864 (hex)

Message: 2149
Recall file has a bad header.

Severity: Error

Further explanation:

EventID: E8080865 (hex)

Message: 2150
Recall file version is obsolete and no longer compatible with this instrument.

Severity: Error

Further explanation:

EventID: E8080866 (hex)

Message: 2151
The recall file contains an istate version newer than this instrument. A remote call to the front panel has returned hresult %1

Severity: Error

Further explanation:

EventID: E8080867 (hex)

Message 2152
"Front Panel <x>

Severity: Error

Further explanation: None

EventID: E8080868 (hex)

Message 2153
"Front Panel message"

Severity: Informational
Further explanation: None

EventID: 68080869 (hex)

Message 2154
"Power Service <x>

Severity: Error

Further explanation: There is more than 1 instance of powerservice running. There should only be one running. This might happen after running install shield - especially when upgrading the CPU board.

Suggestions: Try rebooting. If this persists, please call Customer Support.

EventID: E808086A (hex)

Message 2155
"Power Service <x>

Severity: Informational

Further explanation: None

EventID: 6808086B (hex)

Message 2156
"The Keysight Technologies GPIB driver can not be loaded or unloaded."

Severity: Error

Further explanation: None

Suggestions: If the problem persists, from the analyzer desktop, right-click on My Computer. Click Properties, Click Hardware Tab. Click Device Manager Button. Expand GPIB Devices. Right-click and click Uninstall all GPIB interfaces devices. Reboot the analyzer.

EventID: E808086C (hex)

Message 2157
"The National Instruments GPIB driver can not be loaded or unloaded."

Severity: Error

Further explanation: None
Suggestions: If the problem persists, from the analyzer desktop, right-click on My Computer. Click Properties, Click Hardware Tab. Click Device Manager Button. Expand GPIB Devices. Right-click and click Uninstall all GPIB interfaces devices. Reboot the analyzer.

EventID: E808086D (hex)

Message 2158
"The Keysight GPIB driver is loaded but it can not start its parser."

Severity: Error

Further explanation: None

EventID: E808086E (hex)

Message: 2159
The front panel is in remote mode.

Severity: Warning

EventID: A808086F (hex)

Message: 2160
The Registry Key specified could not be found.

Severity: Error

EventID: E8080870 (hex)

Message: 2161
An overcurrent condition has been detected on a probe plugged into the front panel.

Severity: Warning

EventID: A8080871 (hex)

Message: 2162
The operation timed out.

Severity: Error

EventID: E8080872 (hex)

Message 2163
"The Network Analyzer executed a preset."

Severity: Informational

Further explanation: None

EventID: 68080873 (hex)

Message 2164

"Access to file denied."

Severity: Error

Further explanation: This means that the system can not open an output file for writing. Most likely because the file is write protected.

Suggestions: Pick another file name or file directory, check floppy disk hard disk write access.

EventID: E8080874 (hex)

Message 2165

"File type is structured storage."

Severity: Informational

Further explanation: None

EventID: 68080875 (hex)

Message 2166

"The trigger operation failed."

Severity: Error

Further explanation: None

EventID: E8080876 (hex)

Message 2167

"Argument out of range error."

Severity: Error

Further explanation: None
Suggestions: None

EventID: E8080877 (hex)

**Message: 2169**
The given COM object is not a custom application

**Severity: Error**

EventID: E8080879 (hex)

**Message: 2170**
The eventID supplied was not recognized as a valid analyzer eventID

**Severity: Error**

EventID: E808087A (hex)

**Message: 2171**
The operation was canceled.

**Severity: Error**

EventID: E808087B (hex)

**Message: 2172**
High security level cannot be disabled directly. Only an instrument preset or recall of lower security instrument state will reset this security level.

**Severity: Error**

EventID: E808087C (hex)

**Message: 2173**
Local lockout mode is on. The analyzer application will not accept input from front panel, keyboard or mouse until this mode is turned off from a remote interface.

**Severity: Error**

EventID: E808087D (hex)

**Message: 2174**
The SnP request is not valid for the selected measurement.

**Severity:** Error

**EventID:** E808087E (hex)

**Message:** 2175

Preset is not supported while this dialog or wizard is open. Close the dialog or wizard and then try again.

**Severity:** Error

**EventID:** E808087F (hex)

**Message:** 2176

The function you requested requires a capability provided by an option to the standard analyzer. That option is not currently installed.

**Severity:** Error

**EventID:** E8080880 (hex)

**Message:** 2177

Catastrophic error. Crash dump recorded at <n>

**Severity:** Error

**EventID:** E8080881 (hex)

**Message:** 2178

In the context of a noise calibration, this would occur if the analyzer was unable to set the state of the tuner Ecal module.

**Severity:** Error

**EventID:** E8080882 (hex)

**Message:** 2179

Failed to open gen.lic.

**Severity:** Error

**EventID:** E8080883 (hex)
Message: 3002

Bad port specification.

Severity: Error
About Error Messages

Analyzer errors and Operating System errors are displayed and logged in an error file. You can choose how to display errors, or choose to not display errors at all.

- Error Display
- View Error Log
- List of VNA Errors
- SCPI Errors

Other System topics

Error Display

By default, error messages appear on the screen for a brief period. You can choose to have them stay on the screen until you click an OK button, or have them not appear at all. When they stay on the screen, a Help button is available to provide further assistance.

How to select the display of Error Messages

Using Hardkey/SoftTab/Softkey

1. Press System > Help > Error Display....
On Preset, these settings revert to their defaults (enabled, timed popups).

**Enable Messages** Check to display all error messages as they occur. Clear to suppress the display of error messages. You can still view them in the error log.

**Calibration Error Message Windows**

- **Status bar Display** Displays error messages on the screen for a duration of time proportional to the length of the message. You can then view the message in the error log and get further assistance.

- **Confirmation Dialog boxes** Displays error messages in a standard dialog box. You then choose OK or Cancel to close the dialog box, or press Help to get further information on the error message.

**View Error Log**

The analyzer Error Log is a list of all events that have occurred. (Events are used in programming the analyzer using COM.) Analyzer errors are a subset of events. Only events with severity codes of ERROR are displayed on the screen as they occur. From the error log, you can access further help with an error by selecting the error and clicking Help.

**How to view the Error Log**

Using **Hardkey/SoftTab/Softkey**

1. Press **System > Help > View Error Log**....

**Error Log** dialog box help
Network analyzer errors only  Select to view only analyzer errors. Clear to view all errors that occur on all applications of the computer.

Description  Error message that appears on the analyzer screen.

A - Event ID  Error message number

B - Date the Error occurred

C - Time the Error occurred

D - Severity Code - All events have one of the following severity codes:

- SUCcess - the operation completed successfully
- INFormational - events that occur without impact on the measurement integrity
- WARning - events that occur with potential impact on measurement integrity
- ERRor - events that occur with serious impact on measurement integrity

E - Application in which the error occurred.

OK  Closes the Dialog box

Help  Provides further information on the selected Error message

To clear the Error Log:
1. From the **System > Main** menu click **Minimize Application**.

2. On the desktop, select **Start, Control Panel**

3. On the Control Panel, click **Administrative Tools**

4. On the Administrative Tools window, click **Event Viewer**

5. On the Event Viewer window, right-click **Application**

6. Select **Clear all Events**

7. If you want to save a file with the contents of the Event Log, click **Yes**. Otherwise, click **No**

To restore the VNA application, click on the VNA Analyzer taskbar button at the bottom of the screen.
Analyzer Accessories

- Coax Mechanical Calibration Kits
- Waveguide Mechanical Calibration Kits
- Electronic Calibration (ECal)
- Mechanical Verification Kits
- Adapter and Accessory Kits
- Test Port Cables
- USB Peripherals
- Connector Care and ESD Supplies

Other Support topics

For product and order information:

- Visit www.Keysight.com/find/accessories
- Use the search function to locate information about a particular accessory or view the entire RF and Microwave Test Accessories Catalog.

Accessories are available in these connector types:

- 50 ohm Type-N
- 75 ohm Type-N
- 3.5 mm
- 7 mm (APC-7)
- 7-16
- 2.92 mm
- 2.4 mm
Test port cables and a calibration kit are necessary for a complete measurement system.

A verification kit is used to verify corrected system performance.

See the connector type for each VNA model

### Coax Mechanical Calibration Kits

<table>
<thead>
<tr>
<th>Model</th>
<th>Connector Type</th>
<th>Frequency Upper Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>85032B</td>
<td>Type-N (50 Ohm)</td>
<td>6 GHz</td>
</tr>
<tr>
<td>85032F</td>
<td>Type-N (50 Ohm)</td>
<td>9 GHz</td>
</tr>
<tr>
<td>85054B</td>
<td>Type-N (50 Ohm)</td>
<td>18 GHz</td>
</tr>
<tr>
<td>85036E</td>
<td>Type-N (75 Ohm)</td>
<td>3 GHz</td>
</tr>
<tr>
<td>85050B</td>
<td>7 mm</td>
<td>18 GHz</td>
</tr>
<tr>
<td>85033D</td>
<td>3.5 mm</td>
<td>6 GHz</td>
</tr>
<tr>
<td>85038A</td>
<td>7-16</td>
<td>7.5 GHz</td>
</tr>
<tr>
<td>85033E</td>
<td>3.5 mm</td>
<td>9 GHz</td>
</tr>
<tr>
<td>85052B</td>
<td>3.5 mm</td>
<td>26.5 GHz</td>
</tr>
<tr>
<td>85052C</td>
<td>3.5 mm TRL</td>
<td>26.5 GHz</td>
</tr>
<tr>
<td>85056K</td>
<td>2.92 mm</td>
<td>50 GHz</td>
</tr>
<tr>
<td>85056A</td>
<td>2.4 mm</td>
<td>50 GHz</td>
</tr>
<tr>
<td>85058B/E</td>
<td>1.85 mm (data-based)</td>
<td>67 GHz</td>
</tr>
<tr>
<td>85059A</td>
<td>1.00 mm (data-based)</td>
<td>DC to 110 GHz</td>
</tr>
<tr>
<td>85059B</td>
<td>1.00 mm (data-based)</td>
<td>DC to 120 GHz</td>
</tr>
</tbody>
</table>

### Waveguide Mechanical Calibration Kits
<table>
<thead>
<tr>
<th><strong>Model</strong></th>
<th><strong>Connector Type</strong></th>
<th><strong>Frequency Range</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>X11644A</td>
<td>WR-90</td>
<td>8.2-12.4 GHz</td>
</tr>
<tr>
<td>P11644A</td>
<td>WR-62</td>
<td>12.4-18 GHz</td>
</tr>
<tr>
<td>K11644A</td>
<td>WR-42</td>
<td>18-26.5 GHz</td>
</tr>
<tr>
<td>R11644A</td>
<td>WR-28</td>
<td>26.5-40 GHz</td>
</tr>
<tr>
<td>Q11644A</td>
<td>WR-22</td>
<td>33-50 GHz</td>
</tr>
<tr>
<td>U11644A</td>
<td>WR-19</td>
<td>40-60 GHz</td>
</tr>
<tr>
<td>V11644A</td>
<td>WR-15</td>
<td>50-75 GHz</td>
</tr>
</tbody>
</table>

**Electronic Calibration (ECal)**

<table>
<thead>
<tr>
<th><strong>Model</strong></th>
<th><strong>Connector Type</strong></th>
<th><strong>Frequency Range</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>RF Two-Port</td>
<td></td>
<td></td>
</tr>
<tr>
<td>85091C</td>
<td>7 mm (APC-7)</td>
<td>300 kHz-9 GHz</td>
</tr>
<tr>
<td>85092C</td>
<td>Type-N (50 ohm)</td>
<td>300 kHz-9 GHz</td>
</tr>
<tr>
<td></td>
<td>Port B available with 3.5 mm or 7-16a</td>
<td></td>
</tr>
<tr>
<td>85093C</td>
<td>3.5 mm</td>
<td>300 kHz-9 GHz</td>
</tr>
<tr>
<td></td>
<td>Port B available with Type-N (50 ohm) or 7-16a</td>
<td></td>
</tr>
<tr>
<td>85096C</td>
<td>Type-N (75 ohm)</td>
<td>300 kHz-3 GHz</td>
</tr>
<tr>
<td>85098C</td>
<td>7-16a</td>
<td>300 kHz-7.5 GHz</td>
</tr>
<tr>
<td></td>
<td>Port B available with Type-N (50 ohm) or 3.5 mm</td>
<td></td>
</tr>
<tr>
<td>85099C</td>
<td>Type-F</td>
<td>300 kHz-3 GHz</td>
</tr>
</tbody>
</table>

<p>| RF Four-Port |                   |                     |
| N4431B       | 3.5mm (f) (four-port), Type-N (f) (four-port), Mixed connector types | 9 kHz^b^-13.5 GHz |
| N4432A Option 020 | Type-N (f) (four-port) | 300 kHz-18 GHz (available Feb. 2006) |
| N4432A Option 030 | APC 7 (four-port) | 300 kHz-18 GHz (available Feb. 2006) |</p>
<table>
<thead>
<tr>
<th>Model</th>
<th>Connector Type</th>
<th>Frequency Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>N4433A</td>
<td>3.5mm (f) (four-port)</td>
<td>300 kHz-20 GHz</td>
</tr>
<tr>
<td>Option 010</td>
<td></td>
<td>(available Feb. 2006)</td>
</tr>
</tbody>
</table>

**Microwave Two-Port**

<table>
<thead>
<tr>
<th>Model</th>
<th>Connector Type</th>
<th>Frequency Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>N4690D</td>
<td>Type-N (50 ohm)</td>
<td>DC/300 kHz-18 GHz</td>
</tr>
<tr>
<td>N4691D</td>
<td>3.5 mm</td>
<td>DC/300 kHz-26.5 GHz</td>
</tr>
<tr>
<td>N4692D</td>
<td>2.92 mm</td>
<td>DC/10 MHz-40 GHz</td>
</tr>
<tr>
<td>N4693D</td>
<td>2.4 mm</td>
<td>DC/10 MHz-50 GHz</td>
</tr>
<tr>
<td>N4694D</td>
<td>1.85 mm</td>
<td>DC/10 MHz-67 GHz</td>
</tr>
<tr>
<td>N4696D</td>
<td>7 mm</td>
<td>DC/300 kHz-18 GHz</td>
</tr>
<tr>
<td>N4690B</td>
<td>Type-N (50 ohm)</td>
<td>300 kHz-18 GHz</td>
</tr>
<tr>
<td>N4691B</td>
<td>3.5 mm</td>
<td>300 kHz-26.5 GHz</td>
</tr>
<tr>
<td>N4692A</td>
<td>2.92 mm</td>
<td>10 MHz-40 GHz</td>
</tr>
<tr>
<td>N4693A</td>
<td>2.4 mm</td>
<td>10 MHz-50 GHz</td>
</tr>
<tr>
<td>N4694A</td>
<td>1.85 mm</td>
<td>10 MHz-67 GHz</td>
</tr>
<tr>
<td>N4696BA</td>
<td>7 mm</td>
<td>300 kHz-18 GHz</td>
</tr>
</tbody>
</table>

a Limits ECal module high frequency to 7.5 GHz.

b Performance from 9 kHz to 300 kHz is valid only for the E5071C with firmware version A.09.10 and above, and E5080A with firmware A.11.70.03 and above.

c N469xD models are supported with firmware version A.12..60.02 and above.

**Verification Kits**

<table>
<thead>
<tr>
<th>Model</th>
<th>Connector Type</th>
<th>Frequency Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>85055A</td>
<td>Type-N (50 Ohm)</td>
<td>300 kHz-9 GHz</td>
</tr>
<tr>
<td>85053B</td>
<td>3.5 mm</td>
<td>300 kHz-26.5 GHz</td>
</tr>
<tr>
<td>85057B</td>
<td>2.4 mm</td>
<td>.045-50 GHz</td>
</tr>
<tr>
<td>85059V</td>
<td>1.00 mm</td>
<td>DC to 120 GHz</td>
</tr>
<tr>
<td>R11645A</td>
<td>WR-28</td>
<td>26.5-40 GHz</td>
</tr>
<tr>
<td>Q11645A</td>
<td>WR-22</td>
<td>33-50 GHz</td>
</tr>
</tbody>
</table>

**Adapters and Accessory Kits**
<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>11878A</td>
<td>Type-N to 3.5 mm Adapter Kit</td>
</tr>
<tr>
<td>11525A</td>
<td>Type-N (m) to 7 mm (APC-7)</td>
</tr>
<tr>
<td>11853A</td>
<td>Type-N Accessory Kit</td>
</tr>
<tr>
<td>11900B</td>
<td>2.4 mm (f) to 2.4 mm (f)</td>
</tr>
<tr>
<td>11900C</td>
<td>2.4 mm (f) to 2.4 mm (m)</td>
</tr>
<tr>
<td>85130G</td>
<td>Test Port Adapter Set, 2.4 mm (f) to 2.4 mm (m,f)</td>
</tr>
<tr>
<td>11901B</td>
<td>2.4 mm (f) to 3.5 mm (f)</td>
</tr>
<tr>
<td>11901D</td>
<td>2.4 mm (f) to 3.5 mm (m)</td>
</tr>
<tr>
<td>85130F</td>
<td>Test Port Adapter Set, 2.4 mm (f) to 3.5 mm (m,f)</td>
</tr>
<tr>
<td>11902B</td>
<td>2.4 mm (f) to 7 mm (APC-7)</td>
</tr>
<tr>
<td>11920A</td>
<td>1 mm (m) to 1 mm (m)</td>
</tr>
<tr>
<td>11920B</td>
<td>1 mm (f) to 1 mm (f)</td>
</tr>
<tr>
<td>11920C</td>
<td>1 mm (m) to 1 mm (f)</td>
</tr>
<tr>
<td>11921A</td>
<td>1 mm (m) to 1.85 mm (m)</td>
</tr>
<tr>
<td>11921B</td>
<td>1 mm (f) to 1.85 mm (f)</td>
</tr>
<tr>
<td>11921C</td>
<td>1 mm (m) to 1.85 mm (f)</td>
</tr>
<tr>
<td>11921D</td>
<td>1 mm (f) to 1.85 mm (m)</td>
</tr>
<tr>
<td>11922A</td>
<td>1 mm (m) to 2.4 mm (m)</td>
</tr>
<tr>
<td>11922B</td>
<td>1 mm (f) to 2.4 mm (f)</td>
</tr>
<tr>
<td>11922C</td>
<td>1 mm (m) to 2.4 mm (f)</td>
</tr>
<tr>
<td>11922D</td>
<td>1 mm (f) to 2.4 mm (m)</td>
</tr>
</tbody>
</table>

Test Port Cables
### Connector and ESD Supplies

**See ESD topic**

**See more Connector Care supplies**

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>9300-1367</td>
<td>Adjustable antistatic wrist strap</td>
</tr>
<tr>
<td>9300-0980</td>
<td>Antistatic wrist strap grounding cord (5 foot)</td>
</tr>
<tr>
<td>9300-0797</td>
<td>Static control table mat (2 foot x 4 foot) with earth ground wire</td>
</tr>
<tr>
<td>9300-1126</td>
<td>ESD heel strap</td>
</tr>
<tr>
<td>1401-0248</td>
<td>ESD Safe End-Cap, Type-N (m)</td>
</tr>
<tr>
<td>1401-0247</td>
<td>ESD Safe End-Cap, Type-N (f)</td>
</tr>
<tr>
<td>1401-0214</td>
<td>Standard End-Cap, Type-N (m)</td>
</tr>
<tr>
<td>1401-0225</td>
<td>Standard End-Cap, Type-N (f)</td>
</tr>
</tbody>
</table>

---

### USB Peripherals

**Model** | **Description**
---|---
N4688A | **CD RW drive** - with USB cable.
N4689A | **USB 4-port hub** - for connecting additional USB peripherals.
82357A | **USB/GPIB Interface** - for controlling GPIB devices through USB. Learn more about using the 82357A with the VNA.

---

### Connectors and Supplies

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N4697E</td>
<td>1.85 mm (f) to 1.85 mm (rugged f) flexible (single)</td>
</tr>
<tr>
<td>N4697F</td>
<td>1.85 mm (rugged f, f) to 1.85 mm (rugged m, rugged f) flexible (set)</td>
</tr>
<tr>
<td>N6315A</td>
<td>Type-N (m) to Type-N (f), 16 in. (single)</td>
</tr>
<tr>
<td>N6314A</td>
<td>Type-N (m) to Type-N (m), 24 in. (single)</td>
</tr>
<tr>
<td>85133D</td>
<td>2.4 mm (f) to 2.4 mm (m, f) semi-rigid (set)</td>
</tr>
<tr>
<td>85133F</td>
<td>2.4 mm (f) to 2.4 mm (m, f) flexible (set)</td>
</tr>
<tr>
<td>85134D</td>
<td>2.4 mm (f) to 3.5 mm (m, f) semi-rigid (set)</td>
</tr>
<tr>
<td>85134F</td>
<td>2.4 mm (f) to 3.5 mm (m, f) flexible (set)</td>
</tr>
</tbody>
</table>
The Keysight 82357B is an adapter that creates a GPIB Interface from one of your unused VNA USB ports.

**Applications**

The 82357B can be used to connect a GPIB device using the VNA USB for any VNA application. In addition, the 82357B can be used to connect a power meter for a source power calibration.

**Installing the 82357B USB/GPIB Interface**

1. Download and install firmware VNA revision 3.0 or greater. To check the revision of your VNA firmware, click Help then About Network Analyzer.

**Configure the 82357B USB/GPIB Interface**

When the 82357B is connected to the VNA USB, the driver is automatically installed.

Normally, you do NOT need to edit these settings. The 82357B USB/GPIB Interface is configured automatically as the next unused VISA interface. Keysight Connection Expert allows you to change the setting.

If the VISA Interface Name appears as GPIB0 or GPIB1, these Interfaces must be returned to their default settings for the 82357B to work properly with the VNA. See Configure for VISA / SICL to learn how.

**Connecting the 82357B USB/GPIB Interface**

The following diagram illustrates how to connect GPIB test equipment using the USB/GPIB Interface.
• Plug the USB/GPIB Interface into any unused VNA USB port.
• The driver installation and connection is performed automatically.

**Communicating with Equipment Connected to the USB/GPIB Interface**

• The Frequency Converter Application will automatically find and communicate with test equipment that is connected to the USB/GPIB Interface.

• Source power calibration: Select **GPIB** at the *Power Meter Settings* dialog and specify the GPIB address of the power meter.

• To control other devices through your own program using the 82357B, you must include the new GPIB Interface number when addressing the devices.
Firmware Update

VNA firmware updates are available to you at no cost in a self-extracting Install Shield file. The update includes the VNA application, Online help, and Service Utilities.

To manually check the version of firmware on the VNA, click **System > Help > About NA...**

**Tip:** Use **Move App to Back** to cause the VNA application to move behind this application on the screen.

The following options are available for you to update your VNA application:

- **Auto-Check and Firmware Update** If your VNA is connected to the Internet, these utilities will automatically check for, download, and install, the new firmware and associated files when the VNA application is started. You will be prompted before this occurs.

- **Website Access** If your VNA is NOT connected to the Internet, but you have a PC that is, you can download the VNA firmware and associated files to a storage medium.

**Note: After a firmware update...**

- Custom Cal Kits must be imported. Learn more

- If a different desktop icon named "Network Analyzer" exists, the shortcut to the VNA application will assume the same icon. Right-click on the desktop, then click **Refresh**.

**Other Support Topics**

**Auto-Check**

With Internet access to your VNA, Auto-Check automatically and regularly checks the Internet for new VNA firmware revisions. If a new revision is found, a notification message prompts you to run the **Firmware Update** utility, which then performs the actual download.

Without Internet access to your VNA, Auto-Check provides a reminder prompt at the selected intervals.

Auto-Check is run only when the VNA application is started. Once the VNA application is running, it will not check for updates again until it is restarted.

When Auto-Check runs, it checks the following conditions:
- Is there an active connection to the Internet?
- Is the Auto-Check utility enabled?
- Is it time to check for new firmware?
- Does new firmware exist?

If all of these conditions are true, Auto-Check shows the following dialog box.

If all of these conditions are NOT true, or to change these settings at any time, press **System**, then **Service**, then **Update Firmware**. From within the utility, click **AutoCheck**. These preferences are stored in the VNA registry. Future firmware updates will not change these settings.

---

**VNA Auto-Check** dialog box help

**Enable**  When the VNA application is started, Auto-Check will search the VNA website for firmware updates at the selected time interval.

**Disable**  When the VNA application is started, Auto-Check will NOT search the VNA website for firmware updates.

**Time Interval**  Select the time interval Auto-Check is to search for firmware updates.

**Accept**  Starts update process.

**Ignore**  No further action is taken until the selected time interval has elapsed.

**Remind Me Later:**  This window is displayed again after 1-20 days depending upon the time interval selected.

---

**Firmware Update Utility**
**How to start Firmware Update**

Connect the VNA to the Internet. A LAN connection is recommended because a firmware download can take many hours using a modem.

**Using Hardkey/SoftTab/Softkey**

1. Press **System > Service > Update Firmware**.

**Using a mouse**

1. Click **Utility**
2. Select **System**
3. Select **Service**
4. Select **Update Firmware**

1. Click **Check for Updates**.
2. If updates exist, click **Download & Install**.

---

**AgileUpdate** dialog box help

**Note:** Your privacy is important to Keysight. The update utility does NOT send ANY information from the VNA to the server. It only downloads from the server to the VNA.
**Restart**  Click to restart from the beginning.

**Configure**  Click to launch the Configure dialog box.

**Clean-up**  Click to delete all but the two most recent install shield packages from the VNA hard drive.

**Firmware History**  Available after clicking Check for Updates.

**Auto-Check**  Launches the Auto-Check dialog box.

**Item / Application**  Lists the items available for download at the firmware website.

- Click items with i to read more information about the download.
- Items in RED should be downloaded and installed individually.
- Multi-language help includes all help files except English.

**Note:** The firmware includes the help file. Therefore, only the firmware checkbox will be selected if a new version for both the firmware and the help file are available.

**Select Source**

**Default Website**  The Keysight site that contains update FW.

**Other Specified URL**  Click if you were instructed to get firmware from a different website.

**Check Customer FW Releases**  Check this box to also check Customer Releases in addition to Production Releases. This setting provides you with the very latest VNA firmware. Customer Releases are fully supported but have not yet been tested in all production models. Customer Releases take precedence over Production Releases. This setting is remembered and applied the next time update utility is run.

**Special Access Code...**  Type in the code if you were given one from Keysight Technical Support. Otherwise, leave blank.

**Make Latest Firmware Available...**  Select this checkbox if you want to download the latest firmware, even if it is not new.

**Check for Updates**  Click to look for firmware updates at the Keysight website. If there are newer versions, the files will be listed.

**Download and Install**  When updates are found, this selection becomes available. Some files may be pre-checked. Be sure the corresponding boxes are checked for the files you want to download.
Then click to download and install the update.

**Download Only**  Click to download the files to the analyzer hard disk and install the files at a later time. At that time, click Install from File.

---

### Configuration dialog box help

**Note:** If the update utility will not connect, try to access ANY Internet website. Contact your local IT department if necessary.

**Proxy Setting**

- **No Proxy or Default Proxy**  Click if you use a LAN connection. The update utility will automatically use the proxy specified in Internet Explorer.

- **Use specified Proxy / Port**  Click to enter the proxy name and port. The format is: `proxyName:portNumber`. (The proxy port number is typically 8088).

**Internet timeout**  If you are using an automatic dial-up Internet connection you may need to increase the timeout.

**Current Connection Status**  Shows the current status of the VNA connection to the Internet.

**Note:** These settings are NOT saved; they must be re-entered each time the update utility is run.

---

### Updating firmware

2. Terminate the VNA application by pressing System > Main > Exit.

3. Transfer the file from your PC to your VNA using LAN or USB Pen drive.

4. Double-click the file on the VNA and follow the instruction.

**Warning:** You can save the update file to your PC, but do not attempt to install the VNA application on your PC. It will alter system settings and can result in system crashes.
VNA Configurations and Options

Included with each VNA is a mouse and keyboard. This topic presents VNA models that are supported with this firmware release and the available options.

- PNA-X Series
- VNA N522xB Series
- VNA N523xB (Updated PNA-L) Series
- mmWave Model
- Measurement Receiver N5264B
- Common VNA Options
- Certification Options for ALL Models
- *OPT? and Options COM Behavior
- Warranty Period

See Also

- VNA Series Configuration Guide (requires an Internet connection)
- Click Help then About Network Analyzer to view the options that are installed on your VNA.

Other Support Topics

PNA-X Series

See Block Diagrams

- N5249B: 10 MHz to 8.5 GHz
- N5241B: 10 MHz to 13.5 GHz
- N5242B: 10 MHz to 26.5 GHz
- N5244B: 10 MHz to 43.5 GHz
- N5245B: 10 MHz to 50.0 GHz
- N5247B: 10 MHz to 67 GHz
- N5264B: Measurement Receiver [Learn more]

**PNA-X Options**

See other options that MAY be offered on the PNA-X models

See [NVNA Brochure](#) for more PNA-X options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Required Model/Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>None</td>
<td>2-port base model. Includes six front-panel access loops.</td>
</tr>
<tr>
<td>205</td>
<td>N5241B and N5242B only</td>
<td>2-port base model adds six front-panel jumpers, R1 reference receiver switch, and low frequency extension (LFE).</td>
</tr>
<tr>
<td>217</td>
<td>NOT available on N5247B S93080A/B</td>
<td>2-port standard test set (includes six front-panel access loops), power range, and source and receiver attenuators (extended power range).</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Model</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>N5241B/42B/49B</td>
</tr>
<tr>
<td></td>
<td></td>
<td>N5244B/45B</td>
</tr>
<tr>
<td></td>
<td></td>
<td>N5247B</td>
</tr>
</tbody>
</table>

**Note:** For N5247B models, the extended power range is available ONLY with Opt 219 and 419.

| 219    | None                   | 2-port standard test set (includes six front-panel access loops), power range, source and receiver attenuators (extended power range), and bias-tees. |
|        |                        | **Model** | **Source Attn** | **Receiver Attn** |
|        |                        | N5241B/42B/49B | 0 to 65 dB in 5 dB steps | 0 to 35 dB in 5 dB steps |
|        |                        | N5244B/45B | 0 to 60 dB in 10 dB steps | 0 to 35 dB in 5 dB steps |
|        |                        | N5247B | 0 to 50 dB in 10 dB steps | 0 to 50 dB in 10 dB steps |

<p>| 222    | NOT available on N5247B | 2-port standard test set (includes six front-panel access loops), power range, source and receiver attenuators (extended power range), internal second source, a combiner, and mechanical switches to the 2-port analyzer. |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Allows high-power measurements up to 20 Watts (+43 dBm). from 10 MHz to 26.5 GHz. Similar to the PNA-X -219 or -419 but deletes the bias tees from the test set. Learn more.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>224</td>
<td>S93080A/B</td>
<td>2-port standard test set (includes six front-panel access loops), power range, source and receiver attenuators (extended power range), internal second source, a combiner, mechanical switches to the 2-port analyzer, and bias tees.</td>
</tr>
<tr>
<td>401</td>
<td>None</td>
<td>4-port model base model. Includes twelve front-panel access loops.</td>
</tr>
<tr>
<td>417</td>
<td>NOT available on N5247B</td>
<td>4-port standard test set (includes twelve front-panel access loops), power range, internal second source (Option 080 recommended), and source and receiver attenuators (extended power range). See table above for values. Note: For N5247B models, the extended power range is available ONLY with Opt 219 and 419.</td>
</tr>
<tr>
<td>419</td>
<td>(S93080A/B recommended)</td>
<td>4-port standard test set (includes twelve front-panel access loops), power range, internal second source (Option 080 recommended), and source and receiver attenuators (extended power range), and bias-tees. See table above for values.</td>
</tr>
<tr>
<td>422</td>
<td>NOT available on N5247B</td>
<td>4-port standard test set (includes six front-panel access loops), power range, source and receiver attenuators (extended power range), internal second source, a combiner, and mechanical switches to the 4-port analyzer. Allows high-power measurements up to 20 Watts (+43 dBm). from 10 MHz to 26.5 GHz. Similar to the PNA-X -219 or -419 but deletes the bias tees from the test set. Learn more.</td>
</tr>
<tr>
<td>423</td>
<td>S93080A/B</td>
<td>4-port standard test set (includes six front-panel access loops), power range, source and receiver attenuators (extended power range), internal second source, a combiner, mechanical switches to the 4-port analyzer, and bias tees.</td>
</tr>
<tr>
<td>425</td>
<td>All Models</td>
<td>4-port configurable test set, source and receiver attenuators, internal second source, combiner, mechanical switches, and low frequency extension (LFE).</td>
</tr>
<tr>
<td>425</td>
<td>S93029A/B</td>
<td>4-port configurable test set, source and receiver attenuators, internal second source, combiner, mechanical switches, low frequency extension (LFE), and noise receiver.</td>
</tr>
<tr>
<td>UNY</td>
<td>PNA-X</td>
<td>Enhanced Phase Noise performance for the RF sources</td>
</tr>
<tr>
<td>-----</td>
<td>-----------</td>
<td>-----------------------------------------------------</td>
</tr>
<tr>
<td>XSB</td>
<td>PNA-X</td>
<td>Adds a synthesizer output from 10 MHz to 13.5 GHz to the SRC3 connector on the rear panel. Requires Opt. 422 or 423.</td>
</tr>
</tbody>
</table>

**PNA N522xB Series**

See Block Diagrams.

See Specs

- **N5221B**: 10 MHz to 13.5 GHz
- **N5222B**: 10 MHz to 26.5 GHz
- **N5224B**: 10 MHz to 43.5 GHz
- **N5225B**: 10 MHz to 50.0 GHz
- **N5227B**: 10 MHz to 67 GHz

The N522xB models are identical to the PNA-X series except:

- N522xB 2-port models are NOT available with 2 sources.
- N522xB option numbering is slightly different.
- N522xB models do NOT have internal RF switches or combiners (no RF Path Configuration). This has many measurement implications.
- N522xB models do NOT have rear-panel access to RF Paths.
- N522xB models do NOT offer a Noise Receiver (Opt S93029A/B).

**PNA N522xB Options**

See other options that MAY be offered on the N522xB models
<table>
<thead>
<tr>
<th>Option</th>
<th>Required Model/Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>None</td>
<td>2-port model with single RF source.</td>
</tr>
<tr>
<td>201</td>
<td>All Models</td>
<td>To Opt 200, adds six front-panel jumpers and R1 reference receiver switch.</td>
</tr>
<tr>
<td>205</td>
<td>All Models</td>
<td>To Opt 200, adds six front-panel jumpers, R1 reference receiver switch, and low frequency extension (LFE).</td>
</tr>
<tr>
<td>217</td>
<td>NOT available on N5227B 200 and S93080A/B</td>
<td>To Opt 201, adds source and receiver attenuators.</td>
</tr>
<tr>
<td>219</td>
<td>200</td>
<td>To Opt 217 adds bias-tees between each source and each test port.</td>
</tr>
<tr>
<td>220</td>
<td>All Models</td>
<td>2-port model adds source and receiver attenuators, and low frequency extension (LFE).</td>
</tr>
<tr>
<td>400</td>
<td>None</td>
<td>4-port model with two sources.</td>
</tr>
<tr>
<td>401</td>
<td>400</td>
<td>To Opt 400, adds 12 front-panel jumpers and R1 reference receiver switch.</td>
</tr>
<tr>
<td>405</td>
<td>All Models</td>
<td>4-port base model adds six front-panel jumpers, R1 reference receiver switch, and low frequency extension (LFE).</td>
</tr>
<tr>
<td>417</td>
<td>400</td>
<td>To Opt 401 adds source and receiver attenuators. See table above for values.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> For N5227B models, the extended power range is available ONLY with Opt 219 and 419.</td>
</tr>
<tr>
<td>419</td>
<td>400 (S93080A/B recommended)</td>
<td>To Opt 417 adds bias-tees between each source and each test port.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> For N5227B models, the extended power range is available ONLY with Opt 219 and 419.</td>
</tr>
<tr>
<td>420</td>
<td>All Models</td>
<td>4-port model adds source and receiver attenuators, and low frequency extension (LFE).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>Source Attn</th>
<th>Receiver Attn</th>
</tr>
</thead>
<tbody>
<tr>
<td>N5221B/22B</td>
<td>0 to 65 dB in 5 dB steps</td>
<td>0 to 35 dB in 5 dB steps</td>
</tr>
<tr>
<td>N5224B/25B</td>
<td>0 to 60 dB in 10 dB steps</td>
<td>0 to 35 dB in 5 dB steps</td>
</tr>
<tr>
<td>N5227B</td>
<td>0 to 50 dB in 10 dB steps</td>
<td>0 to 50 dB in 10 dB steps</td>
</tr>
</tbody>
</table>

_N523xB (Updated PNA-L) Series_
See Specs

- N5231B: 300 kHz to 13.5 GHz  2-port / 4-port
- N5232B: 300 kHz to 20.0 GHz  2-port / 4-port
- N5234B: 10 MHz to 43.5 GHz  2-port ONLY
- N5235B: 10 MHz to 50.0 GHz  2-port ONLY
- N5239B: 300 kHz to 8.5 GHz  2-port ONLY

The N523xB models are identical to the N5230C series except:

- N523xB models have a PNA-X look and feel.
- N523xB models are NOT available with 2 sources.
- ALL N5231B, N5232B, and N5239B models require a Delta Match calibration.

N523xB Options

See other options that MAY be offered on the N523xB models

<table>
<thead>
<tr>
<th>Option</th>
<th>Required Model/Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>None</td>
<td>Base 2-port model.</td>
</tr>
<tr>
<td>216</td>
<td>None</td>
<td>To base 2-port model, adds:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Six front-panel jumpers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Source Attenuator: 60 dB with 10 dB steps.</td>
</tr>
<tr>
<td>400</td>
<td>None</td>
<td>Base 4-port model. Available ONLY on N5231B and N5232B</td>
</tr>
<tr>
<td>416</td>
<td>None</td>
<td>To base 4-port model, adds:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Six front-panel jumpers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Source Attenuator: 60 dB with 10 dB steps.</td>
</tr>
</tbody>
</table>

Millimeter Wave VNA
<table>
<thead>
<tr>
<th>PNA Model</th>
<th>Frequency Range</th>
<th>Ports</th>
<th>Connector Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>N5250C (discontinued)</td>
<td>10 MHz to 110 GHz</td>
<td>2</td>
<td>1.0 mm</td>
</tr>
<tr>
<td>N5251B</td>
<td>10 MHz to 110 GHz</td>
<td>2</td>
<td>1.0 mm</td>
</tr>
</tbody>
</table>

Test heads to 325 GHz are also available.

**PNA N5290A/N5291A Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Required Model/Options</th>
<th>Frequency Range</th>
<th>Ports</th>
<th>Connector Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>N5222B VNA with Option 205 N5292A Test Set with Options 200 and 222 Two N5293AX03 Frequency Extenders (N5290A) Two N5295AX03 Frequency Extenders (N5291A)</td>
<td>900 Hz to 110 GHz (N5290A) 900 Hz to 120 GHz (N5291A)</td>
<td>2</td>
<td>3.5 mm at VNA Ports 3.5 mm at test set</td>
</tr>
<tr>
<td>202</td>
<td>N5227B VNA with Option 205 N5292A Test Set with Options 200 and 224 Two N5293AX03 Frequency Extenders (N5290A) Two N5295AX03 Frequency Extenders (N5291A)</td>
<td>900 Hz to 110 GHz (N5290A) 900 Hz to 120 GHz (N5291A)</td>
<td>2</td>
<td>2.4 mm at VNA Ports 3.5 mm at test set</td>
</tr>
<tr>
<td>401</td>
<td>N5242B VNA with Option 425 N5292A Test Set with Options 400 and 442 Four N5293AX03 Frequency Extenders (N5290A) Four N5295AX03 Frequency Extenders (N5291A)</td>
<td>900 Hz to 110 GHz (N5290A) 900 Hz to 120 GHz (N5291A)</td>
<td>4</td>
<td>3.5 mm at VNA Ports 3.5 mm at test set</td>
</tr>
<tr>
<td>402</td>
<td>N5242B VNA with Option 425, S93029A/B</td>
<td>900 Hz to 110 GHz (N5290A)</td>
<td>4</td>
<td>3.5 mm at VNA Ports</td>
</tr>
<tr>
<td>N5292A Test Set with Options 400 and 442</td>
<td>900 Hz to 120 GHz (N5291A)</td>
<td>3.5 mm at test set</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>---------------------------</td>
<td>-------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Four N5293AX03 Frequency Extenders (N5290A)</td>
<td>900 Hz to 110 GHz (N5290A)</td>
<td>2.4 mm at VNA Ports</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Four N5295AX03 Frequency Extenders (N5291A)</td>
<td>900 Hz to 120 GHz (N5291A)</td>
<td>3.5 mm at test set</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Measurement Receiver**

Block Diagrams are available at the end of each specifications document.

<table>
<thead>
<tr>
<th>PNA Model</th>
<th>Frequency Range</th>
<th>Ports</th>
<th>Connector Type</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>N5264B</td>
<td>IF Frequencies only</td>
<td>0</td>
<td>N/A</td>
<td>1</td>
</tr>
</tbody>
</table>

Learn more

**Options**

**108** - Built-in 26.5 GHz LO source with +10 dBm output power.

**S93118A/B** - Fast-CW mode enables 500 million point data buffer.

**Common Options**

The following options are common to more than one VNA family. For PNA-B model and E5080A which is shipped after Sep 2017, the software option is provided as a product number S93xxxA for PNA and S96xxxA for ENA(E5080A). The fixed, transportable, perpetual and time limitation options are available.
<table>
<thead>
<tr>
<th>Software Product/Option</th>
<th>Available on models:</th>
<th>Description</th>
</tr>
</thead>
</table>
| S93007A/B              | ALL models           | **Automatic Fixture Removal**  
Mathematically removes, or de-embeds, a characterized test fixture from displayed measurement results. Learn more |
| S93010A/B              | ALL models           | **Time-domain**  
Adds time-domain capability to analyzer. The serial number of the analyzer must be specified when ordering this kit. Software upgrade. Learn more about Time Domain Learn how this option is enabled. |
| S93011A/B              | ALL models           | **Enhanced Time Domain Analysis**  
Adds time domain reflectometry (TDR) and time domain transmission (TDT) capability to analyzer. Learn more.  
**Note:** The S93011A/B license can be used for the standard Time Domain measurements. |
| S93015A/B              | ALL models           | **Real-time S-Parameter and Power Measurement Uncertainty**  
Display the measurement uncertainty dynamically ('real-time') on the same screen as the measurement trace. Learn more. |
<p>| 020                    | PNA-X N522xB         | Add IF inputs on the rear panel for antenna and millimeter-wave. |
| 021                    | PNA-X N522xB         | Add pulse modulator to internal Source1. Learn more. |
| 022                    | PNA-X 2-source N522xB models | Add pulse modulator to internal Source2. Learn more. |
| S93025A/B              | PNA-X N522xB         | Basic pulsed-RF measurements. Learn more. |
| S93026A/B              | PNA-X                | <strong>Integrated Pulsed Application</strong> |</p>
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>N522xB</td>
<td>Provides average pulse and point-in-pulse measurements.</td>
<td>Learn more.</td>
</tr>
<tr>
<td>S93027B</td>
<td>PNA-X with 029</td>
<td>Allows PNA-X with option 029 to use specialized mechanical noise tuners on the input port. Learn more.</td>
</tr>
<tr>
<td>S93029A/B</td>
<td>PNA-X with S93080A/B and one of the following: 219, 222, 224, 419, 423 or 422</td>
<td>Noise Figure Application</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Adds hardware and firmware for high-accuracy noise figure measurements on amplifiers or converters using internal low-noise receivers or a standard VNA receiver. Learn more.</td>
</tr>
<tr>
<td>H29</td>
<td>N5244B or N5245B with 423 and S93080A/B.</td>
<td>(Obsolete) Same as Opt S93029A/B for the N5242B, but on a N5244B or N5245B. Learn more.</td>
</tr>
<tr>
<td>S930317B</td>
<td>PNA/PNA-X</td>
<td>Phase Noise Application, up to 70 GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note: This feature applies ONLY to instruments with serial prefix 6021 and above.</td>
</tr>
<tr>
<td>S930321B</td>
<td>PNA/PNA-X</td>
<td>Phase Noise Application, up to 125 GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note: This feature applies ONLY to instruments with serial prefix 6021 and above.</td>
</tr>
<tr>
<td>S930700B</td>
<td>PNA/PNA-X</td>
<td>Modulation Distortion, up to 8.5 GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn more.</td>
</tr>
<tr>
<td>S930701B</td>
<td>PNA/PNA-X</td>
<td>Modulation Distortion, up to 13.5 GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn more.</td>
</tr>
<tr>
<td>S930702B</td>
<td>PNA/PNA-X</td>
<td>Modulation Distortion, up to 26.5 GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn more.</td>
</tr>
<tr>
<td>S930704B</td>
<td>PNA/PNA-X</td>
<td>Modulation Distortion, up to 43.5 GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn more.</td>
</tr>
<tr>
<td>S930705B</td>
<td>PNA/PNA-X</td>
<td>Modulation Distortion, up to 50 GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn more.</td>
</tr>
<tr>
<td>Model</td>
<td>Model</td>
<td>Feature</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td>S930707B</td>
<td>PNA/PNA-X</td>
<td><strong>Modulation Distortion, up to 67 GHz</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="#">Learn more</a></td>
</tr>
<tr>
<td>S93072B</td>
<td>PNA-X</td>
<td><strong>Arbitrary Waveform Generation on All Ports</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>This application software allows you to generate arbitrary waveforms including multitone signals, limited to 6.2866 microseconds max period. It must be on a frequency grid of ((M/N) \times 600 \text{ MHz}). <a href="#">Learn more</a></td>
</tr>
<tr>
<td>S93080A/B</td>
<td>PNA-X</td>
<td><strong>Frequency Offset Mode (FOM)</strong></td>
</tr>
<tr>
<td></td>
<td>N522xB</td>
<td>Enables you to set the VNA source independently from where the receivers are tuned. This capability is important for measuring mixers and converters. <a href="#">Learn more</a></td>
</tr>
<tr>
<td>S93082A/B</td>
<td>ALL models</td>
<td><strong>Scalar Mixer Measurements (SMC)</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Allows Only the Scalar Mixer Converter (SMC) portion of the Frequency Converter Measurement Application. Provides the same intuitive user-interface, easy calibration, and external source control for making fixed and swept LO Scalar Mixer measurements. When used with a multiport VNA or external test set, SMC is only available on VNA ports 1 and 2. PNA requires Opt S93080A/B. ENA S96082A includes FOM function. <a href="#">Learn more</a></td>
</tr>
<tr>
<td>S93083A/B</td>
<td>PNA-X</td>
<td><strong>Frequency Converter Application (FCA)</strong></td>
</tr>
<tr>
<td></td>
<td>N522xB</td>
<td>Provides an intuitive user-interface for making extremely accurate conversion loss and absolute group delay measurements on mixers and converters. Exceptional amplitude and phase accuracy is achieved through two calibration techniques: Scalar Mixer Calibration and Vector Mixer Calibration. The application also provides automatic control of all of Keysight's major signal sources. Requires Opt S93080A/B. <a href="#">Learn more</a></td>
</tr>
<tr>
<td>Model Code</td>
<td>Description</td>
<td>Details</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>S93084A/B</td>
<td>Embedded LO</td>
<td>Provides the ability to measure frequency converters that have an embedded LO. Requires at least one Converter App option. Learn more.</td>
</tr>
<tr>
<td>S93086A/B</td>
<td>Gain Compression Application. (GCA)</td>
<td>Provides fast and accurate gain compression measurements. Learn more.</td>
</tr>
<tr>
<td>S93087A/B</td>
<td>Swept IMD and IM Spectrum.</td>
<td>Provides fast and accurate Swept IMD and IM Spectrum measurements on amplifiers and converters. Learn more. Requires Opt S93080A/B.</td>
</tr>
<tr>
<td>S93088A/B</td>
<td>Source Phase Control</td>
<td>Provides coherent phase measurements. Learn more.</td>
</tr>
<tr>
<td>S93089A/B</td>
<td>Differential IQ</td>
<td>Provides controls to set any source to any frequency range, any power level, and any phase. Then measure any receiver, any frequency range, with match correction. Learn more.</td>
</tr>
<tr>
<td>S930900A/B</td>
<td>Spectrum analysis, up to 8.5 GHz</td>
<td>Learn more.</td>
</tr>
<tr>
<td>S930901A/B</td>
<td>Spectrum analysis, up to 13.5 GHz</td>
<td>Learn more.</td>
</tr>
<tr>
<td>S930902A/B</td>
<td>Spectrum analysis, up to 26.5 GHz</td>
<td>Learn more.</td>
</tr>
<tr>
<td>S930904A/B</td>
<td>Spectrum analysis, up to 43.5 GHz</td>
<td>Learn more.</td>
</tr>
<tr>
<td>Part Number</td>
<td>Models Supported</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>S930905A/B</td>
<td>N5225B, N5227B, N5224B, N5225B, N5235B, N5245B, N5247B</td>
<td>Spectrum analysis, up to 50 GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn more.</td>
</tr>
<tr>
<td>S930907A/B</td>
<td>N5227B, N5247B</td>
<td>Spectrum analysis, up to 67 GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn more.</td>
</tr>
<tr>
<td>S930909A/B</td>
<td>N5222B, N5224B, N5225B, N5227B, N5242B, N5244B, N5245B, N5247B</td>
<td>Spectrum analysis, up to 90 GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn more.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn more.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn more.</td>
</tr>
<tr>
<td>108</td>
<td>N5264B Only</td>
<td>Adds an internal 10 MHz to 26.5 GHz LO source. Learn more.</td>
</tr>
<tr>
<td>S93110A/B</td>
<td>4-Port PNA-X models with 2 sources</td>
<td>Active (Hot) Parameters</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn more.</td>
</tr>
<tr>
<td>S93111A/B</td>
<td>4-Port PNA-X models with 2 sources</td>
<td>Active (Hot) Parameters - restricted to 50 GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn more.</td>
</tr>
<tr>
<td>Option</td>
<td>Supported Models</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>UK6</td>
<td>ALL except N5250A</td>
<td>Complete set of measurement data which was acquired from testing your VNA to published specifications. Includes calibration label, calibration certificate, and data report. Conforms to ISO 9001.</td>
</tr>
<tr>
<td>1A7</td>
<td>ALL except N5250A</td>
<td>Complete set of measurement data which was acquired from testing your VNA to published specifications. Includes calibration label, ISO 17025 calibration certificate, data report, measurement uncertainties and guard bands on all specifications. Conforms to ISO17025 and ISO 9001.</td>
</tr>
</tbody>
</table>

**Certification Options for ALL Models**

The following options are available for your VNA.

<table>
<thead>
<tr>
<th>Option</th>
<th>Supported Models</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S93118A/B</td>
<td>PNA-X</td>
<td>Fast CW Mode Enables 500 million point data buffer. Learn more.</td>
</tr>
<tr>
<td>S93460A/B</td>
<td>4-port PNA-X and N522xB</td>
<td>iTMSA Adds firmware for Integrated True Mode Balanced measurements. Learn more.</td>
</tr>
<tr>
<td>S93898A/B</td>
<td>ALL models</td>
<td>Built-in performance test software</td>
</tr>
</tbody>
</table>

**Opt? and Options COM Behavior**

Some of the VNA option numbers returned when using the **Opt?** SCPI command or `IApplication::Options` COM command are not the same as the option numbers returned when using the `SYST:CAP:LIC:CAT?` command or `IApplication::Licenses` COM command. The following table shows how the common option numbers map to the option numbers reported when using **Opt?**
command or the `IApplication::Options` COM command.

<table>
<thead>
<tr>
<th>Software Product Number</th>
<th>*OPT? SCPI or Options COM Response</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S93007A/B, S95007A, S96007A, S97010A</td>
<td>007</td>
<td>Automatic Fixture Removal</td>
</tr>
<tr>
<td>S93010A/B, S95010A, S96010A, S97010A</td>
<td>010</td>
<td>Time-domain</td>
</tr>
<tr>
<td>S93015A/B</td>
<td>015</td>
<td>Dynamic Uncertainty for S-Parameters</td>
</tr>
<tr>
<td>S93025A/B</td>
<td>025</td>
<td>Add four internal pulse generators</td>
</tr>
<tr>
<td>S93026A/B, S95026A, S97026A</td>
<td>008</td>
<td>Integrated Pulsed Application</td>
</tr>
<tr>
<td>S93029A/B,</td>
<td>028</td>
<td>Noise Figure application only</td>
</tr>
<tr>
<td>S93029A/B, S95029B, S97029B</td>
<td>028 + 080</td>
<td>Noise Figure application + Frequency Offset Mode (FOM) for converter applications</td>
</tr>
<tr>
<td>S95029A, S97029A</td>
<td>029</td>
<td>Noise Figure application + Noise Figure receiver</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Noise Figure receiver only</td>
</tr>
<tr>
<td>S93080A/B</td>
<td>080</td>
<td>Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S93082A/B, S95082A, S96082A, S97082A</td>
<td>082 + 080 (Others), 082 (ENA)</td>
<td>Scalar Mixer Converter (SMC) + Frequency Offset Mode (FOM) for converter applications</td>
</tr>
<tr>
<td>S93083A/B, S95083A, S97083A</td>
<td>083 + 080</td>
<td>Frequency Converter Application (FCA) + Frequency Offset Mode (FOM) for converter applications</td>
</tr>
<tr>
<td>S93084A/B, S95084A, S97084A</td>
<td>084 + 080</td>
<td>Embedded LO + Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S93086A/B,</td>
<td>086 + 080</td>
<td>Gain Compression Application (GCA) + Frequency Offset Mode (FOM) for converter applications</td>
</tr>
<tr>
<td>Code</td>
<td>Code</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>S96086A, S95086A, S97086A</td>
<td>086</td>
<td>Gain Compression Application (GCA)</td>
</tr>
<tr>
<td>S93087A/B</td>
<td>087 + 080</td>
<td>Swept IMD and IM Spectrum + Frequency Offset Mode (FOM) for converter applications</td>
</tr>
<tr>
<td>S93088A/B</td>
<td>088</td>
<td>Source Phase Control</td>
</tr>
<tr>
<td>S93089A/B</td>
<td>089 + 080</td>
<td>Differential IQ + Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S95090A, S97090A</td>
<td>090 + 080</td>
<td>Spectrum analysis + Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S930900A/B</td>
<td>090 + 900</td>
<td>Spectrum analysis, up to 8.5 GHz + Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S930901A/B</td>
<td>090 + 901</td>
<td>Spectrum analysis, up to 13.5 GHz + Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S930902A/B</td>
<td>090 + 902</td>
<td>Spectrum analysis, up to 26.5 GHz + Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S930904A/B</td>
<td>090 + 904</td>
<td>Spectrum analysis, up to 43.5 GHz + Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S930905A/B</td>
<td>090 + 905</td>
<td>Spectrum analysis, up to 50 GHz + Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S930907A/B</td>
<td>090 + 907</td>
<td>Spectrum analysis, up to 67 GHz + Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S930909A/B</td>
<td>090 + 909</td>
<td>Spectrum analysis, up to 90 GHz + Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S93093A/B</td>
<td>093 + 080</td>
<td>Extend Spectrum Analyzer to 110 GHz + Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S93094A/B</td>
<td>094 + 080</td>
<td>Extend Spectrum Analyzer to above 110 GHz + Frequency Offset Mode (FOM)</td>
</tr>
<tr>
<td>S93118A/B</td>
<td>118</td>
<td>Fast CW Mode</td>
</tr>
<tr>
<td>S93460A/B</td>
<td>460</td>
<td>iTMSA</td>
</tr>
<tr>
<td>S93551A/B, S95551A, S97551A</td>
<td>551</td>
<td>N-port capabilities</td>
</tr>
<tr>
<td>Model</td>
<td>Code</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>S96790A</td>
<td>790</td>
<td>Measurement Wizard Assistant Software</td>
</tr>
<tr>
<td>S93898A/B</td>
<td>898</td>
<td>Built-in performance test software</td>
</tr>
<tr>
<td>S94510A/B</td>
<td>510</td>
<td>Nonlinear component characterization</td>
</tr>
<tr>
<td>S94511A/B</td>
<td>511</td>
<td>Nonlinear component characterization, restricted to 50 GHz</td>
</tr>
<tr>
<td>S94514A/B</td>
<td>514</td>
<td>Nonlinear X-parameters</td>
</tr>
<tr>
<td>S94518A/B</td>
<td>518</td>
<td>Nonlinear pulse envelope domain</td>
</tr>
<tr>
<td>S94520A/B</td>
<td>520</td>
<td>Arbitrary load-impedance X-parameters</td>
</tr>
<tr>
<td>S94521A/B</td>
<td>520</td>
<td>Arbitrary load-control, X-parameters</td>
</tr>
<tr>
<td>S94522A/B</td>
<td>520</td>
<td>Arbitrary load-control, device characterization</td>
</tr>
</tbody>
</table>

**Documentation**

**Description**


A documentation CD-ROM is no longer included with each VNA shipment (Feb.2005).

To download a service guide for your VNA, or the latest version of VNA Help, visit [www.Keysight.com/find/pna](http://www.Keysight.com/find/pna), search for your VNA model, then click Library.

**VNA Warranty Period**

The actual warranty on your instrument depends on the date it was ordered as well as whether or not any warranty options were purchased at that time. To determine the exact warranty on your instrument, contact Keysight Technologies with the model and serial number of your instrument.

For online information about Keysight's service and support products visit:
Option Enable

The software option can be installed by yourself. See https://www.keysight.com/find/LicensingSupport about detailed Keysight license information.

On your PC


2. Log in or register your name if first time log-in.

3. Click “You can add a new certificate”
4. Locate the **Software License Entitlement Certificate**.

5. Enter your “Order Number” and “Certificate Number” on Software Entitlement Certificate.

6. Follow the instructions to register the ordered software licenses.

7. Click “You can request new licenses”

8. Select your required license (ex. S9xxxxx)
9. Add your **HOST ID**.

**Node-locked / Transportable / Trial Licenses**

**PXI VNA / S9405xB Simulator**

Select the host to assign licenses to

- **Model Number**: PCSERNO
- **Serial Number**: FF7996258

**Bench Top VNA**

The license file can be copied to the root directory of the USB device and then connected to the VNA. The Keysight License Manager is always running, and it scans the root directory of any new drive looking for license files. When the file is found, the licenses are automatically installed, and a notification is displayed on the screen. If the licenses are not installed automatically, then the following instructions should be followed.
USB VNA

Floating License for all VNAs and Simulator

USB Portable License for all VNAs and Simulator

Click “Assign License” at the bottom.

10. Follow the instructions.

11. The message with license file (.lic) is sent to your email address.

On the analyzer or controller for analyzer

Node-locked / Transportable / Trial Licenses

Note: Do not install the floating/USB licenses on Keyight License Manager. Use Keysight License Manager 6 instead.

1. Execute the Keyight License Manager from Start Menu in Windows
2. Click “What if I have a license file to install”, then follow the instruction to install the license file.

Detailed instructions can also be found in the Keysight License Manager help.

**Floating License**

The floating license allows you to borrow/return the license from the license server. Your local controller can be a license server if required.

1. Make sure if the license has already been installed in the license server.

2. Execute the Keysight License Manager 6 as Administrator from Start Menu in Windows on the analyzer or controller for analyzer.

3. Click **Specify a remote floating license server**, then enter the host name or IP address of the license server.

4. Select **Borrow License** Tab, then click **Borrow a floating license**.

5. Select the required license and specify the return date.

6. Select **View License** Tab, then confirm if the selected license is installed.

Detailed instructions can also be found in the Keysight Licensing Administrator's Guide

Setup a license server

1. Download a Keysight License Manager 6 from https://www.keysight.com/find/LicensingSupport and install it on a PC which you will assign as a license server. If you want to use your controller as a server, the installation is not necessary. See next.

2. Execute the Keysight License Manager 6 as Administrator from Start Menu in Windows.

3. Click Add/remove a license on your local machine

4. Select Add license to the floating license serve to be used by other machines.

5. Click Browse… to select license file (.lic), then click Next.

6. Select View License Tab, then confirm if the selected license is installed.

Use your controller as a license server

The network connection is not required when your controller is used as a license server.

1. Execute the Keysight License Manager 6 as Administrator from Start Menu in Windows on the analyzer or controller for analyzer.

2. Click Add/remove a license on your local machine

3. Select Select a product to license and use on the machine.

4. Select PNA, ENA or PXIVNA. Select PXIVNA for USB VNA.

5. Click Browse… to select license file (.lic), then click Next.

6. Select View License Tab, then confirm if the selected license is installed.

Known Issue

When the VNA does not detect the floating license, execute the following procedure.

1. Execute C:\Windows\system32\services.msc and find EEsof EDA License Server.
2. If the status is either blank (Stopped) or status type is Automatic (Delayed Start), then right click and select **Properties**.

3. Change the status type from Automatic (Delayed Start) to Automatic, then click **Start** of the Service Status.
4. Confirm if the Status is changed to **Started** and **Automatic**.
USB Potable License

The application/option is enabled only when the analyzer or controller has both license and the corresponding USB dongle. A license file can be installed on many analyzers or controllers. The application is activated while the dongle is plugged in.

1. Execute the Keysight License Manager 6 as Administrator from Start Menu in Windows on the analyzer or controller for analyzer.

2. Click *Add/remove a license on your local machine*.

3. Select *Select a product to license and use on the machine*.

4. Select PNA, ENA or PXIVNA. Select PXIVNA for USB VNA.

5. Click *Browse…* to select license file (.lic), then click *Next*.

6. Select *View License* Tab, then confirm if the selected license is installed.

7. When you use the application on the analyzer, connect the dongle in the USB port to enable the capability for license.

*Note:* If your analyzer or controller does not detect the dongle, download and install the FLEXID10USBDriver package for your platform from https://www.keysight.com/find/LicensingUsbDriver.

Detailed instructions can also be found in the Keysight Licensing Administrator's Guide.


About HOST ID

The HOST ID will be asked when you get the license file.

Host ID for Fixed / Transportable / Trial Licenses
PXI VNA/ Bench Top VNA / Simulator

HOST ID is shown in the Keysight License Manager. Keyight License Manager can be executed from Start Menu in Windows.

The HOST ID syntax for Bench Top VNA is {Model No.},{Serial Number}

The HOST ID syntax for PXI VNA and S94050B/51B Simulator is PCSERNO, {two letters and eight
numbers (this is nothing related with PC serial number)
Host ID for Floating

HOST ID is Mac address

On the server machine, execute `getmac /v /fo list`. Your host ID is listed as the Physical Address of the Network Adapter.
Host ID for USB Portable

HOST ID is printed on a USB dongle.

The USB dongle can be ordered as an option of USB portable licence or Keysight P/N E8900-80036 Hardware Key-USB Port FLEX10.
Instrument Calibration

An instrument calibration is a process where the analyzer performance is measured to ensure that it operates within specifications. If any performance parameter does not conform to the published specifications, adjustments are made to bring the performance into conformance.

Why Should I Get an Instrument Calibrated?

Over time, the active components in the analyzer age and the performance may degrade or drift.

To ensure that the analyzer is performing to the published specifications, you should have an instrument calibration performed periodically.

How Often Should I Get an Instrument Calibrated?

It is your responsibility to determine the calibration period which best meets your requirements. However, a 12 to 18 month calibration cycle is appropriate for most users.

There are two things to consider: performance drift and connector wear.

- The instrument specifications are set to consider the performance drift that may occur over a 24 month period. Therefore, getting the instrument calibrated at 24 month intervals ensures that the analyzer maintains performance within the operating specifications. If you need the analyzer to maintain more consistent operation, you may want to have the instrument calibrated more often than the recommended 24-month interval.

- Connector wear is a bigger factor and depends on the number of connections that are made. The test ports become noticeably worn after 500 to 700 connections. This could represent about 12 months with average use. With more frequent connections, the calibration cycle should be sooner. You can extend the time between calibrations and thereby save money by using connector savers and by performing proper Connector Care.

How Do I Get an Instrument Calibrated?

To get the instrument calibrated, send it to one of the Keysight Technologies service centers. See Technical Support.

The analyzer must be fully functional when it is sent to the service center, or they will charge for their repair services. If the analyzer is being used in a secure environment where the hard drive can not be sent with the analyzer, a second hard drive must be purchased and configured for use with the analyzer in an "unclassified" environment before the analyzer is sent to the service center.

To perform the instrument calibration yourself, you must have the following required items:
What Are My Choices of Instrument Calibration?

The following types of instrument calibration are available from Keysight Technologies at the time of initial order:

**Standard**
Includes a certificate of calibration stating the instrument has been calibrated and is operating within the published specifications.

**Option UK6**
Available ONLY at the initial shipment. Includes the test data from the calibration and the certificate of calibration stating the instrument has been calibrated and is operating within the published specifications.

**Option A6J**
Available ONLY at the initial shipment. Includes the test data and measurement uncertainties from the calibration and the certificate of calibration stating the instrument has been calibrated using a process in compliance with ANSI Z540 and is operating within the published specifications.

**Option 1A7**
Available ONLY at the initial shipment. Includes the test data and measurement uncertainties from the calibration and the certificate of calibration stating the instrument has been calibrated using a process in compliance with ISO 17025 and is operating within the published specifications.

The following types of instrument calibration are available from Keysight Technologies service center:

**Keysight Calibration**
Includes the test data from the calibration and the certificate of calibration, stating the instrument has been calibrated and is operating within the published specifications.

**ANSI Z540 Calibration**
Includes the test data from the calibration and the certificate of calibration, stating the instrument has been calibrated using a process in compliance with ANSI Z540.1 and is operating within the published specifications.

**ISO 17025 Calibration**
Includes the test data from the calibration and the certificate of calibration, stating the instrument has been calibrated using a process in compliance with ISO 17025 and is operating within the published specifications.

For more information on these options, visit [www.Keysight.com/find/calibration](http://www.Keysight.com/find/calibration).
Other Resources

The following network analysis resources are also available.

Document Resources

Application Notes

Third-Party Resources

For information about test fixtures and part handlers, contact:

   Inter-Continental Microwave  
   www.icmicrowave.com

For information about probing equipment and accessories, contact:

   Cascade Microtech, Inc.  
   www.cascademicrotech.com
SCPI Errors

-100 to -200 Command Errors

-200 to -299 Execution Errors

-300 to -399 SCPI Specified Device-Specific Errors

-400 to -800 Query and System Errors

100 to 230 VNA-specific Errors

See Also

Analyzer Error messages.

-100 to -200 Command Errors

A command error indicates that the test set's GPIB parser has detected an IEEE 488.2 syntax error. When one of these errors is generated, the command error bit in the event status register is set.

-100 std_command Command - This event bit (Bit 5) indicates a syntax error, or a semantic error, or a GET command was entered, see IEEE 488.2, 11.5.1.1.4.

-101 std_invalidChar Invalid character - Indicates a syntactic elements contains a character which is invalid for that type.

-102 std_syntax Syntax - Indicates that an unrecognized command or data type was encountered. For example, a string was received when the device does not accept strings.

-103 std_invalidSeparator Invalid separator - The parser was expecting a separator and encountered an illegal character. For example, the semicolon was omitted after a program message unit.

-104 std_wrongParamType Data type - The parser recognized a data element different than one allowed. For example, numeric or string data was expected but block data was encountered.

-105 std_GETNotAllowed GET not allowed - Indicates a Group Execute Trigger was received within a program message. Correct the program so that the GET does not occur within the program code.
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-108</td>
<td>std_tooManyParameters</td>
</tr>
<tr>
<td></td>
<td>Parameter not allowed - Indicates that more parameters were received than expected for the header. For example, *ESE common command only accepts one parameter, so *ESE 0,1 is not allowed.</td>
</tr>
<tr>
<td>-109</td>
<td>std_tooFewParameters</td>
</tr>
<tr>
<td></td>
<td>Missing parameter - Indicates that less parameters were received than required for the header. For example, *ESE requires one parameter, *ESE is not allowed.</td>
</tr>
<tr>
<td>-110</td>
<td>std_cmdHeader</td>
</tr>
<tr>
<td></td>
<td>Command header - Indicates an error was detected in the header. This error is used when the device cannot detect the more specific errors -111 through -119.</td>
</tr>
<tr>
<td>-111</td>
<td>std_headerSeparator</td>
</tr>
<tr>
<td></td>
<td>Header separator - Indicates that a character that is not a legal header separator was encountered while parsing the header.</td>
</tr>
<tr>
<td>-112</td>
<td>std_IDTooLong</td>
</tr>
<tr>
<td></td>
<td>Program mnemonic too long - Indicates that the header contains more that twelve characters, see IEEE 488.2, 7.6.1.4.1.</td>
</tr>
<tr>
<td>-113</td>
<td>std_undefinedHeader</td>
</tr>
<tr>
<td></td>
<td>Undefined header - Indicates the header is syntactically correct, but it is undefined for this specific device. For example, *XYZ is not defined for any device.</td>
</tr>
<tr>
<td>-114</td>
<td>std_suffixOutOfRange</td>
</tr>
<tr>
<td></td>
<td>Header suffix out of range - Indicates the value of a header suffix attached to a program mnemonic makes the header invalid.</td>
</tr>
<tr>
<td>-120</td>
<td>std_numericData</td>
</tr>
<tr>
<td></td>
<td>Numeric data - This error, as well as errors</td>
</tr>
<tr>
<td>-121</td>
<td>std_invalidCharInNumber</td>
</tr>
<tr>
<td></td>
<td>Invalid character in number - Indicates an invalid character for the data type being parsed was encountered. For example, an alpha in a decimal numeric or a &quot;9&quot; in octal data.</td>
</tr>
<tr>
<td>-123</td>
<td>std_exponentTooLarge</td>
</tr>
<tr>
<td></td>
<td>Exponent too large - Indicates the magnitude of an exponent was greater than 32000, see IEEE 488.2, 7.7.2.4.1.</td>
</tr>
<tr>
<td>-124</td>
<td>std_decimalTooLong</td>
</tr>
<tr>
<td></td>
<td>Too many digits - Indicates the mantissa of a decimal numeric data element contained more than 255 digits excluding leading zeros, see IEEE 488.2, 7.7.2.4.1.</td>
</tr>
<tr>
<td>-128</td>
<td>std_numericNotAllowed</td>
</tr>
<tr>
<td></td>
<td>Numeric data not allowed - Indicates that a legal numeric data element was received, but the device does not accept one in this position for the header.</td>
</tr>
<tr>
<td>-130</td>
<td>std_suffix</td>
</tr>
<tr>
<td></td>
<td>Suffix - This error, as well as errors -131 through -139, are generated when parsing a suffix. This particular error message is used if the device cannot detect a more specific error.</td>
</tr>
<tr>
<td>-131</td>
<td>std_badSuffix</td>
</tr>
<tr>
<td></td>
<td>Invalid suffix - Indicates the suffix does not follow the syntax described in IEEE 488.2, 7.7.3.2, or the suffix is inappropriate for this device.</td>
</tr>
<tr>
<td>-134</td>
<td>std_suffixTooLong</td>
</tr>
<tr>
<td></td>
<td>Suffix too long - Indicates the suffix contain more than 12 characters, see IEEE 488.2, 7.7.3.4.</td>
</tr>
<tr>
<td>Error Code</td>
<td>Error Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-138 std_suffixNotAllowed</td>
<td>Suffix not allowed - Indicates that a suffix was encountered after a numeric element that does not allow suffixes.</td>
</tr>
<tr>
<td>-140 std_charData</td>
<td>Character data - This error, as well as errors.</td>
</tr>
<tr>
<td>-141 std_invalidCharData</td>
<td>Invalid character data - Indicates that the character data element contains an invalid character or the particular element received is not valid for the header.</td>
</tr>
<tr>
<td>-144 std_charDataTooLong</td>
<td>Character data too long - Indicates the character data element contains more than twelve characters, see IEEE 488.2, 7.7.1.4.</td>
</tr>
<tr>
<td>-148 std_charNotAllowed</td>
<td>Character data not allowed - Indicates a legal character data element was encountered where prohibited by the device.</td>
</tr>
<tr>
<td>-150 std_stringData</td>
<td>String data - This error, as well as errors.</td>
</tr>
<tr>
<td>-151 std_stringInvalid</td>
<td>Invalid string data - Indicates that a string data element was expected, but was invalid, see IEEE 488.2, 7.7.5.2. For example, an END message was received before the terminal quote character.</td>
</tr>
<tr>
<td>-158 std_stringNotAllowed</td>
<td>String data not allowed - Indicates that a string data element was encountered but was not allowed by the device at this point in parsing.</td>
</tr>
<tr>
<td>-160 std_blockData</td>
<td>Block data - This error, as well as errors -161 through -169, are generated when parsing a block data element. This particular error message is used if the device cannot detect a more specific error.</td>
</tr>
<tr>
<td>-161 std_badBlock</td>
<td>Invalid block data - Indicates a block data element was expected, but was invalid, see IEEE 488.2, 7.7.6.2. For example, an END message was received before the end length was satisfied.</td>
</tr>
<tr>
<td>-168 std_blockNotAllowed</td>
<td>Block data not allowed - Indicates a legal block data element was encountered, but not allowed by the device at this point in parsing.</td>
</tr>
<tr>
<td>-170 std_expr</td>
<td>Expression - This error, as well as errors -171 through -179, are generated when parsing an expression data element. This particular error message is used if the device cannot detect a more specific error.</td>
</tr>
<tr>
<td>-171 std_invalidExpression</td>
<td>Invalid expression - Indicates the expression data element was invalid, see IEEE 488.2, 7.7.7.2. For example, unmatched parentheses or an illegal character.</td>
</tr>
<tr>
<td>-178 std_exprNotAllowed</td>
<td>Expression data not allowed - Indicates a legal expression data was encountered, but was not allowed by the device at this point in parsing.</td>
</tr>
</tbody>
</table>
-180 std_macro  Macro - This error, as well as error -181 through -189, are generated when defining a macro or execution a macro. This particular error message is used if the device cannot detect a more specific error.

-181 std_validOnlyInsideMacro  Invalid outside macro definition - Indicates that a macro parameter place holder was encountered outside of a macro definition.

-183 std_invalidWithinMacro  Invalid inside macro definition - Indicates that the program message unit sequence, sent with a *DDT or a *DMC command, is syntactically invalid, see IEEE 488.2, 10.7.6.3.

-184 std_macroParm  Macro parameter - Indicates that a command inside the macro definition had the wrong number or type of parameters.

-200 to -299 Execution Errors

These errors are generated when something occurs that is incorrect in the current state of the instrument. These errors may be generated by a user action from either the remote or the manual user interface

-200 std_execGen  Execution - This event bit (Bit 4) indicates a PROGRAM DATA element following a header was outside the legal input range or otherwise inconsistent with the device's capabilities, see IEEE 488.2, 11.5.1.1.5.

-201 std_invalidWhileInLocal  Invalid while in local

-202 std_settingsLost  Settings lost due to rtl

-203 std_commandProtected  Command protected - Indicates that a legal password-protected program command or query could not be executed because the command was disabled.

-210 std_trigger  Trigger

-211 std_triggerIgnored  Trigger ignored

-212 std_armIgnored  Arm ignored

-213 std_initIgnored  Init ignored

-214 std_triggerDeadlock  Trigger deadlock

-215 std_armDeadlock  Arm deadlock

-220 std_parm  Parameter - Indicates that a program data element related error occurred.

-221 std_settingsConflict  Settings conflict - Indicates that a legal program data element was parsed but could not be executed due to the current device state.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-222</td>
<td>std_dataOutOfRange: Data out of range - Indicates that a legal program data element was parsed but could not be executed because the interpreted value was outside the legal range defined by the devices.</td>
</tr>
<tr>
<td>-223</td>
<td>std_tooMuchData: Too much data - Indicates that a legal program data element of block, expression, or string type was received that contained more data than the device could handle due to memory or related device-specific requirements.</td>
</tr>
<tr>
<td>-224</td>
<td>std_illegalParmValue: Illegal parameter value - Indicates that the value selected was not part of the list of values given.</td>
</tr>
<tr>
<td>-225</td>
<td>std_noMemoryForOp: Out of memory - The device has insufficient memory to perform the requested operation.</td>
</tr>
<tr>
<td>-226</td>
<td>std_listLength: Lists not same length - Attempted to use LIST structure having individual LIST's of unequal lengths.</td>
</tr>
<tr>
<td>-230</td>
<td>std_dataCorruptOrStale: Data corrupt or stale - Indicates invalid data, a new reading started but not completed since the last access.</td>
</tr>
<tr>
<td>-231</td>
<td>std_dataQuestionable: Data questionable - Indicates that measurement accuracy is suspect.</td>
</tr>
<tr>
<td>-232</td>
<td>std_invalidFormat: Invalid format.</td>
</tr>
<tr>
<td>-233</td>
<td>std_invalidVersion: Invalid version - Indicates that a legal program data element was parsed but could not be executed because the version of the data is incorrect to the device. For example, a not supported file version, a not supported instrument version.</td>
</tr>
<tr>
<td>-240</td>
<td>std_hardware: Hardware - Indicates that a legal program command or query could not be executed because of a hardware problem in the device.</td>
</tr>
<tr>
<td>-241</td>
<td>std_hardwareMissing: Hardware missing - Indicates that a legal program command or query could not be executed because of missing device hardware. For example, an option was not installed.</td>
</tr>
<tr>
<td>-250</td>
<td>std_massStorage: Mass storage - Indicates that a mass storage error occurred. The device cannot detect the more specific errors described for errors -251 through -259.</td>
</tr>
<tr>
<td>-251</td>
<td>std_missingMassStorage: Missing mass storage - Indicates that a legal program command or query could not be executed because of missing mass storage.</td>
</tr>
<tr>
<td>-252</td>
<td>std_missingMedia: Missing media - Indicates that a legal program command or query could not be executed because of missing media. For example, no disk.</td>
</tr>
<tr>
<td>-253</td>
<td>std_corruptMedia: Corrupt media - Indicates that a legal program command or query could not be executed because of corrupt media. For example, bad disk or wrong format.</td>
</tr>
<tr>
<td>-254</td>
<td>std_mediaFull: Media full - Indicates that a legal program command or query could not be executed because the media is full. For example, there is no room left on the disk.</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-255</td>
<td>std_directoryFull - Indicates that a legal program command or query could not be executed because the media directory was full.</td>
</tr>
<tr>
<td>-256</td>
<td>std_fileNotFound - Indicates that a legal program command or query could not be executed because the file name was not found on the media.</td>
</tr>
<tr>
<td>-257</td>
<td>std_fileName - Indicates that a legal program command or query could not be executed because the file name on the device media was in error. For example, an attempt was made to read or copy a nonexistent file.</td>
</tr>
<tr>
<td>-258</td>
<td>std_mediaProtected - Indicates that a legal program command or query could not be executed because the media was protected. For example, the write-protect switch on a memory card was set.</td>
</tr>
<tr>
<td>-260</td>
<td>std_expression - Expression</td>
</tr>
<tr>
<td>-261</td>
<td>std_math - Math in expression</td>
</tr>
<tr>
<td>-270</td>
<td>std_macroExecution - Macro - Indicates that a macro related execution error occurred.</td>
</tr>
<tr>
<td>-271</td>
<td>std_macroSyntax - Macro syntax - Indicates that a syntactically legal macro program data sequence, according to IEEE 488.2, 10.7.2, could not be executed due to a syntax error within the macro definition.</td>
</tr>
<tr>
<td>-272</td>
<td>std_macroExec - Macro execution - Indicates that a syntactically legal macro program data sequence could not be executed due to some error in the macro definition, see IEEE 488.2, 10.7.6.3.</td>
</tr>
<tr>
<td>-273</td>
<td>std_badMacroName - Illegal macro label - Indicates that the macro label was not accepted, it did not agree with the definition in IEEE 488.2, 10.7.3</td>
</tr>
<tr>
<td>-274</td>
<td>std_macroPlaceholderMa - Macro parameter - Indicates that the macro definition improperly used a macro parameter placeholder, see IEEE 4882, 10.7.3.</td>
</tr>
<tr>
<td>-275</td>
<td>std_macroTooLong - Macro definition too long - Indicates that a syntactically legal macro program data sequence could not be executed because the string of block contents were too long for the device to handle, IEEE 488.2, 10.7.6.1.</td>
</tr>
<tr>
<td>-276</td>
<td>std_macroRecursion - Macro recursion - Indicates that a syntactically legal macro program data sequence count not be executed because it would be recursive, see IEEE 488.2, 10.7.6.6.</td>
</tr>
<tr>
<td>-277</td>
<td>std_cantRedefineMacro - Macro redefinition not allowed - Indicates that redefining an existing macro label, see IEEE 488.2, 10.7.6.4.</td>
</tr>
<tr>
<td>-278</td>
<td>std_macroNotFound - Macro header not found - Indicates that a legal macro label in the *GMS?, see IEEE 488.2, 10.13, could not be executed because the header was not previously defined.</td>
</tr>
<tr>
<td>-280</td>
<td>std_program - Program</td>
</tr>
<tr>
<td>-281</td>
<td>std_cantCreateProgram - Cannot create program</td>
</tr>
</tbody>
</table>
-282 std_illegalProgramName  Illegal program name
-283 std_illegalVarName      Illegal variable name
-284 std_programRunning     Program currently running
-285 std_programSyntax      Program syntax
-286 std_programRuntime     Program runtime
-290 std_memoryUse          Memory use
-291 std_execOutOfMemory    Out of memory
-292 std_nameNotFound       Referenced name does not exist
-293 std_nameAlreadyExists  Referenced name already exists
-294 std_incompatibleType   Incompatible type

-300 to -399 SCPI Specified Device-Specific Errors

A device-specific error indicates that the instrument has detected an error that occurred because some operations did not properly complete, possibly due to an abnormal hardware or firmware condition. For example, an attempt by the user to set an out of range value will generate a device specific error. When one of these errors is generated, the device specific error bit in the event status register is set.

-300 std_deviceSpecific     Device specific - This event bit (Bit 3) indicates that a device operation did not properly complete due to some condition, such as overrange see IEEE 488.2, 11.5.1.1.6.
-310 std_system             System
-311 std_memory             Memory - Indicates some physical fault in the devices memory, such as a parity error.
-312 std_PUDmemoryLost      PUD memory lost - Indicates protected user data saved by the *PUD command has been lost, see IEEE 488.2, 10.27.
-313 std_calMemoryLost      Calibration memory lost - Indicates that nonvolatile calibration data used by the *CAL? command has been lost, see IEEE 488.2, 10.2.
-314 std_savRclMemoryLost   Save/recall memory lost - Indicates that the nonvolatile data saved by the *SAV command has been lost, see IEEE 488.2, 10.33.
-315 std_configMemoryLost   Configuration memory lost - Indicates that nonvolatile configuration data saved by the device has been lost.
-320 std_storageFault       Storage fault - Indicates that the firmware detected a fault when using data storage. This is not an indication of physical damage or failure of any mass storage element.
-321 std_outOfMemory        Out of memory - An internal operation needed more memory than was available
-330 std_selfTestFailed     Self-test failed - Indicates a problem with the device that is not covered by a specific error message. The device may require service.
-340 std_calFailed Calibration failed - Indicates a problem during calibration of the device that is not covered by a specific error.
-350 std_queueOverflow Queue overflow - Indicates that there is no room in the queue and an error occurred but was not recorded. This code is entered into the queue in lieu of the code that caused the error.
-360 std_comm Communication - This is the generic communication error for devices that cannot detect the more specific errors described for error -361 through -363.
-361 std_parity Parity in program message - Parity bit not correct when data received for example, on a serial port.
-362 std_framing Framing in program message - A stop bit was not detected when data was received for example, on a serial port (for example, a baud rate mismatch).
-363 std_inputBufferOverrun Input buffer overrun - Software or hardware input buffer on serial port overflows with data caused by improper or nonexistent pacing.

A Query error is generated either when data in the instrument's GPIB output queue has been lost, or when an attempt is being made to read data from the output queue when no output is present or pending.

-400 std_queryGen Query - This event bit (Bit 2) indicates that an attempt to read data from the Output Queues when no output is present or pending, to data in the Output Queue has been lost see IEEE488.2, 11.5.1.1.7.
-410 std_interrupted Query INTERRUPTED - Indicates the test set has been interrupted by a new program message before it finishes sending a RESPONSE MESSAGE see IEEE 488.2, 6.3.2.3.
-420 std_terminated Query UNTERMINATED - Indicates an incomplete Query in the program see IEEE 488.2, 6.3.2.2.
-430 std_deadlocked Query DEADLOCKED - Indicates that the Input Buffer and Output Queue are full see IEEE 488.2, 6.3.1.7.
-440 std_responseNotAllowed Query UNTERMINATED after indefinite response - Indicates that a query was received in the same program message after a query requesting an indefinite response was executed see IEEE 488.2, 6.5.7.5.

-500 std_powerOn Power on
-600 std_userRequest User request
-700 std_requestControl Request control
-800 std_operationComplete Operation complete
## Analyzer-Specific (Positive) SCPI Errors

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>dupWindNum &quot;Duplicate window number&quot;</td>
</tr>
<tr>
<td>101</td>
<td>windNumNotFound &quot;Window number not found&quot;</td>
</tr>
<tr>
<td>102</td>
<td>failedWindCreate &quot;Window creation failed&quot;</td>
</tr>
<tr>
<td>103</td>
<td>noCalcParamSelection &quot;CALC measurement selection set to none&quot;</td>
</tr>
<tr>
<td>104</td>
<td>dupMeasName &quot;Duplicate measurement name&quot;</td>
</tr>
<tr>
<td>105</td>
<td>dataNotFound &quot;Requested data not available&quot;</td>
</tr>
<tr>
<td>106</td>
<td>measNotFound &quot;Requested measurement not found&quot;</td>
</tr>
<tr>
<td>107</td>
<td>traceNotFound &quot;Requested trace not found&quot;</td>
</tr>
<tr>
<td>108</td>
<td>notImplemented &quot;Mnemonic not yet implemented&quot;</td>
</tr>
<tr>
<td>109</td>
<td>noDocument &quot;No measurement container found&quot;</td>
</tr>
<tr>
<td>110</td>
<td>dupTraceNum &quot;Duplicate trace number&quot;</td>
</tr>
<tr>
<td>111</td>
<td>titleStrTooLong &quot;Title string exceeds 50 characters&quot;</td>
</tr>
<tr>
<td>112</td>
<td>memoryNotFound &quot;Requested memory not found&quot;</td>
</tr>
<tr>
<td>113</td>
<td>exceedMaxTraces &quot;Exceeded the maximum number of traces per window&quot;</td>
</tr>
<tr>
<td>114</td>
<td>SerNumNotFound &quot;The serial number was not found. Please store the serial number.&quot;</td>
</tr>
<tr>
<td>115</td>
<td>LoadFailed &quot;The state was not loaded. Please check the file name.&quot;</td>
</tr>
<tr>
<td>116</td>
<td>StoreFailed &quot;The state was not stored. Please check the file and path names.&quot;</td>
</tr>
<tr>
<td>117</td>
<td>File &quot;An in the File operation occurred. Please check file and path names.&quot;</td>
</tr>
<tr>
<td>118</td>
<td>measChanConflict &quot;Measurement does not belong to specified channel.&quot;</td>
</tr>
<tr>
<td>119</td>
<td>exceedMaxWindows &quot;Exceeded the maximum number of data windows&quot;</td>
</tr>
<tr>
<td>120</td>
<td>markerNotFound &quot;The specified marker was not found.&quot;</td>
</tr>
<tr>
<td>121</td>
<td>diagnostic &quot;Diagnostic .&quot;</td>
</tr>
<tr>
<td>122</td>
<td>channelNotFound &quot;The specified channel was not found.&quot;</td>
</tr>
<tr>
<td>123</td>
<td>exceedMaxMeasurements &quot;Exceeded the maximum number of allowed mesurements.&quot;</td>
</tr>
<tr>
<td>124</td>
<td>parameterOutOfRange &quot;The specified value was out of range.&quot;</td>
</tr>
<tr>
<td>125</td>
<td>userRangeNotValid &quot;The currently selected user range is not valid.&quot;</td>
</tr>
<tr>
<td>126</td>
<td>referenceMarkerNotFound &quot;The reference marker is not active.&quot;</td>
</tr>
<tr>
<td>127</td>
<td>sweepSegmentNotFound &quot;The sweep segment was not found.&quot;</td>
</tr>
<tr>
<td>128</td>
<td>markerNotDelta &quot;The specified marker is not a delta marker.&quot;</td>
</tr>
<tr>
<td>129</td>
<td>printoutFailed &quot;Attempt to output to a printer failed.&quot;</td>
</tr>
<tr>
<td>130</td>
<td>memory_trace_not_compatible &quot;Memory not compatible. Trace Math not applied.&quot;</td>
</tr>
<tr>
<td>131</td>
<td>trace_math_reset &quot;Memory not compatible. Trace Math turned off.&quot;</td>
</tr>
<tr>
<td>132</td>
<td>hw_read_failed &quot;Hardware read failed.&quot;</td>
</tr>
</tbody>
</table>

6729
133  hw_write_failed  "Hardware write failed."
134  dsp_active  "Failed because DSP was not halted."
135  secure_memory  "Attempt to access secure memory region."
136  snum_protected  "The serial number is protected."
137  snum_format_bad  "The serial number format is bad."
138  snum_already_set  "The serial number is already set."
139  hw_setting_failed  "Hardware setting failed."
140  cal_access_failed  "Calibration data access failed."
141  db_access_failed  "Database access failed."
142  memory_range_exceeded  "Command exceeds usable memory range."
143  lost_phase_lock  "Phase lock has been lost."
144  over_power  "Detected too much power at input."
145  ee_wrt_failed  "EEPROM write failed."
146  yig_cal_failed  "YTO calibration failed."
147  ramp_cal_failed  "Analog ramp calibration failed."
148  dspcom_bad  "DSP communication failed."
149  no_license_found  "Request failed. The required license was not found."
150  argLimited  "The argument was out of range"
151  markerBWNotFound  "The Marker Bandwidth was not found."
152  peakNotFound  "The Peak was not found."
153  targetNotFound  "The Target search value was not found."
154  calNotImpl  "The Calibration feature requested is not implemented."
155  calClassNotValidForCalType  "SENS:CORR:CCH measurement selection set to none"
156  calNotValidForConfidenceChe  "Selected measurement does not have a calibration valid for Confidence Check"
157  invalidPort  "Specified port is out of range"
158  invalidPortPath  "ROUT:PATH:DEF:PORT  x, y does not match measurement; setting to defaults"
159  ioInvalidWrite  "Attempted I/O write while port set to read only."
160  ioInvalidRead  "Attempted I/O read from write only port."
161  calsetNotFound  "Requested Cal Set was not found in Cal Set Storage."
162  noCalSetSelected  "There is no Cal Set currently selected for the specified channel."
163  cantDeleteCalSetInUse  "Cannot delete a Cal Set while it is being used."
164  calsetStimChange  "Channel stimulus settings changed to match selected Cal Set."
165  exceedMaxCalSets  "Exceeded the maximum number of cal sets."
"A valid calibration is required before correction can be turned on."

"The attempted operation can only be performed on a standard measurement type." 

"A valid divisor buffer is required before normalization can be turned on." 

"Receiver power cal requires the measurement to be of unratioed power." 

"Could not configure the Electronic Calibration system. Check to see if the module is plugged into the proper connector."

"This measurement does not support memory operations" 

"This measurement does not support normalize operations."

"User characterization was not found in the Electronic Calibration module." 

"The data provided has an invalid number of points. It could not be stored." 

"The source power measurement failed. Please check GPIB, power meter settings and sensor connections." 

"The custom cal type does not support remote sessions." 

"An unexpected error occurred in the custom cal module." 

"The specified cal kit file does not exist or the file format is not recognized." 

"Guided cal initialization failed." 

"Security level of High or greater can only be disabled by instrument preset or recall of lower security instrument state." 

"An instrument option that is required for this command is non installed." 

"Parameter not valid." 

"Fixturing: the requested S2P file cannot be opened." 

"A user calset with this name already exists." 

"An attempt was made to acquire calibration data before the system was properly initialized." 

"Requested to measure a cal standard connection step undefined for this cal. Please check number of connection steps and their descriptions." 

"A request was made for control of a testset which does not appear to be connected." 

"The requested operation is not valid for the current sweep type."
"Could not generate the error terms."
"The connector specified is invalid."
"The Calibration Kit specified is invalid."
"The requested feature is not available in the current context."
"Cannot load error terms into the calibration sequence for the requested port, because the cal sequence does not involve that port."
"The hardware does not support the ALC mode specified."
"Invalid path element."
"The specified trigger connection is not available."
"Unleveled, source 1, out 1."
"Trace trigger requires point sweep mode, no internal triggers and must be in channel scope."
"Could not detect the specified ECal module. Please ensure that it is connected and that you are specifying its ID."
"The INITiate command must be successfully issued before steps of a remote ECal user characterization can be accessed."
"The specified calibration kit name is not recognized as a kit that has a definition factory-installed on the PNA."
"WARNING: Receiver overload on one or more ports."
"Power calibration must be enabled for the guided calibration before the power calibration settings can be modified."
"The property value specified is not valid in this context."
"The requested interface is not configured for this instrument."
"Illegal parameter(s) in the SCPI command."
"ADC overflow on one or more ports. Restart averaging if applied."
Technical Support

Click on the region of interest.

For more contact information, visit http://www.Keysight.com/find/contactus

Other Support Topics

United States:
(tel) (+1) 800-829-4444
(alt) (+1) 303 662 3999
(fax) (+1) 888 900 8921

Canada
(tel) 1 877 894 4414
(fax) 1 (905) 206 4120

Europe:

Austria
(tel) 0820 87 44 11*
(fax) 0820 87 44 22
<table>
<thead>
<tr>
<th>Country</th>
<th>(tel)</th>
<th>(alt)</th>
<th>(fax)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Belgium</td>
<td>(+32) (0)2 404 9340</td>
<td>(+32) (0)2 404 9000</td>
<td>(+32) (0)2 404 9395</td>
</tr>
<tr>
<td>Denmark</td>
<td>(+45) 7013 1515</td>
<td>(+45) 7013 7313</td>
<td>(+45) 7013 1555</td>
</tr>
<tr>
<td>Finland</td>
<td>08 0052 4000</td>
<td>(+358) 10 855 2100</td>
<td>(+358) 92 536 0176</td>
</tr>
<tr>
<td>France</td>
<td>0825 010 700*</td>
<td>(+33) (0)1 6453 5623</td>
<td>0825 010 701*</td>
</tr>
<tr>
<td>Germany</td>
<td>01805 24 6333*</td>
<td>01805 24 6330*</td>
<td>01805 24 6336*</td>
</tr>
<tr>
<td>Ireland</td>
<td>(+353) (0)1 890 924 204</td>
<td>(+353) (0)1 890 924 206</td>
<td>(+353) (0)1 890 924 024</td>
</tr>
<tr>
<td>Israel</td>
<td>6734</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Italy
(tel) (+39) (0)2 9260 8484
(fax) (+39) (0)2 9544 1175

Luxemburg
(tel) (+32) (0)2 404 9340
(alt) (+32) (0)2 404 9000
(fax) (+32) (0)2 404 9395

Netherlands
(tel) (+31) (0)20 547 2111
(alt) (+31) (0)20 547 2000
(fax) (+31) (0)20 547 2190

Russia
(tel) (+7) 095 797 3963
(alt) (+7) 095 797 3900
(fax) (+7) 095 797 3901

Spain
(tel) (+34) 91 631 3300
(alt) (+34) 91 631 3000
(fax) (+34) 91 631 3301

Sweden
(tel) 0200 88 22 55*
(alt) (+46) (0)8 5064 8686
Switzerland (French)
(tel) 0800 80 5353 opt. 2*
(alt) (+33) (0)1 6453 5623
(fax) (+41) (0)22 567 5313

Switzerland (German)
(tel) 0800 80 5353 opt. 1*
(alt) (+49) (0)7031 464 6333
(fax) (+41) (0)1 272 7373

Switzerland (Italian)
(tel) 0800 80 5353 opt. 3*
(alt) (+39) (0)2 9260 8484
(fax) (+41) (0)22 567 5314

United Kingdom
(tel) (+44) (0)7004 666666
(alt) (+44) (0)7004 123123
(fax) (+44) (0)7004 444555

Japan:
(tel) 0120 421 345
(alt) (+81) 426 56 7832
(fax) 0120 421 678

Latin America:

Mexico
(tel) (+52) 55 5081 9469
<table>
<thead>
<tr>
<th>Region</th>
<th>Tel</th>
<th>Alternative Tel</th>
<th>Fax</th>
<th>Alternative Fax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brazil</td>
<td>(+55) 11 4197 3600</td>
<td></td>
<td>(+55) 11 4197 3800</td>
<td></td>
</tr>
<tr>
<td>Australia</td>
<td>1800 629 485</td>
<td>1800 143 243</td>
<td>1800 142 134</td>
<td></td>
</tr>
<tr>
<td>New Zealand</td>
<td>0 800 738 378</td>
<td></td>
<td>64 4 495 8950</td>
<td></td>
</tr>
<tr>
<td>Asia Pacific</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>China</td>
<td>800 810 0189</td>
<td>(+86) 10800 650 0021</td>
<td>800 820 2816</td>
<td></td>
</tr>
<tr>
<td>Hong Kong</td>
<td>800 930 871</td>
<td>(+852) 3197 7889</td>
<td>(+852) 2 506 9233</td>
<td></td>
</tr>
<tr>
<td>India</td>
<td>1600 112 929</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
(fax) 000800 650 1101

**Malaysia**

(tel) 1800 888 848
(alt) 1800 828 848
(fax) 1800 801 664

**Singapore**

(tel) 1800 375 8100
(fax) (+65) 6836 0252

**South Korea**

(tel) 080 769 0800
(alt) (+82) 2 2004 5004
(fax) (+82) 2 2004 5115

**Taiwan**

(tel) 0800 047 866
(alt) 00801 651 317
(fax) 0800 286 331

**Thailand**

(tel) 1800 226 008
(alt) (+66) 2 268 1345
(fax) (+66) 2 661 3714
Software Support

Each software license includes free software updates and support for the specific instrument application for the duration of your support subscription. A perpetual license includes software updates and support for the first year, after which support can be renewed annually for a fee. A subscription-based license includes software updates and support through the term of the license.

There are three dialogs that provide support information:

- My Software Support
- My Support ID
- Licensed Features

Access Software Support

Using Hardkey/SoftTab/Softkey


Using a mouse

1. Click Help.
2. Select Tech Support.

My Software Support

This dialog displays software-specific support information.

Refer to the Keysight PNA Series Firmware History support page to check the firmware version history specific to the date displayed in the Software Version Date column in the My Software Support dialog:
**Model Number** - Displays the model number of the feature.

**Description** - Displays a description of the feature.

**Support Expiration** - Displays the support expiration date.

**License Expiration** - Displays the license type.

**Contact Keysight** button - Opens the dialog showing contact information:

![Contact Keysight Dialog](image)

**My Support ID**

This dialog displays your support IDs used to register for software updates, services, and technical support. The support ID is typically the same as the host ID used to serve licenses. Your product configuration may have more than one support ID, each comprised of multiple fields of information.
Model Number - Displays the model number of the feature.

Description - Displays a description of the feature.

Support ID - Displays your support ID, which is typically the same as the host ID.

Copy icon - Copy the corresponding ID information to the clipboard.

Copy All to Clipboard... button - Copies all the contents of this dialog to the clipboard.

Licensed Features

This dialog displays a list of new features in the VNA firmware and corresponding license.

On start-up, the following dialog appears if the user has at least one feature with an expired support contract. In this case, the columns Current Support Contract and Enabled are hidden. ONLY the expired features/licenses are shown.
In the normal case, all licenses are shown and all columns are shown.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Displays the measurement feature.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model Number</td>
<td>Displays the license model number.</td>
</tr>
<tr>
<td>Model Description</td>
<td>Displays a description of the license model.</td>
</tr>
<tr>
<td>Release Date</td>
<td>Displays the license model firmware release date.</td>
</tr>
<tr>
<td>Current Support Contract</td>
<td>Displays the support contract number for the specific feature.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Indicates whether the feature is enabled or not.</td>
</tr>
<tr>
<td>Help</td>
<td>Link to information about the feature.</td>
</tr>
<tr>
<td>Show at Startup</td>
<td>When enabled, will display the Licensed Features dialog at startup. If unchecked and default preferences are restored, the Show at Startup will be checked.</td>
</tr>
</tbody>
</table>
Open Source Software License

JPEG Software Package

This software is based in part on the work of the Independent JPEG Group.

TightVNC

Redistribution of TightVNC is licensed under the General Public License version 2. Source code of TightVNC and a copy of the GPLv2 may be found in the directory of \opensource\TightVNC.

Portions of this software are distributed under one or more Open Source terms and are not warranted and supported by Keysight. This disclaimer does not affect any statutory rights that may exist in any country of distribution. The disclaimed Open Source software portions include the following software package(s): TightVNC. The text of the license for each software package is contained in a directory reflecting the name of the Open Source software that is found in the "\opensource" directory. The author and not Keysight grants a license to use and further distribute the Open Source software. Any license to use and further distribute the Open Source software is granted by the author(s) of such Open Source software in their Open Source license, not by Keysight in this license instrument. The author(s)’s license terms, if any, are found in text files and, if applicable, the source code of the separate Open Source software packages.

TO THE EXTENT PERMITTED BY LOCAL LAW:

Keysight provides the Open Source software listed above "as is" and any express or implied warranties, including, but not limited to any warranty of non-infringement, the implied warranties of merchantability, satisfactory quality, reasonable care and skill, and fitness for a particular purpose are expressly disclaimed; and

Keysight shall not be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including: procurement of substitute goods or services; loss of use, data, or profits; or business interruption) arising from the use of open source software, however caused and regardless of whether such claims are based upon contract, strict liability or tort (except gross negligence or willful misconduct of Keysight), or any other legal theory even if advised of the possibility of such damage and even if it has been ensured that such data can be reconstructed with reasonable expenditure from data material provided in machine-readable form.

Commercial Software License

MathWorks MATLAB Compiler Runtime v7.14
MathWorks MATLAB Compiler; Permission to use the MatLab MCR library is defined in a license agreement between MatLab and you, the Licensee. A copy of the license agreement may be found in “c:\Program Files (x86)\MATLAB\MATLAB Compiler Runtime” directory.
Diagnostic Tools, Utilities, and Adjustments

The following Tools, Utilities, and Adjustments are available to help you keep your VNA at peak performance.

**Diagnostic Tools**

- Operators Check
- System Verification
- Display Test

**Utilities**

- Receiver Display
- Restore ECAl Memory
- Speaker Volume
- Receiver Temperature

**Adjustments**

*Not all of the adjustments listed below are valid for every VNA model. Only the adjustments needed for each particular VNA will be listed.*

To access the service adjustments on the VNA, click **Utility**, then **System**, then **Service**, then **Adjustment Routines**...

**Standard and TDR Adjustments**

**Standard Adjustments**
Adjustments should be performed in the order listed (in green.)
All adjustments require that you be an administrator on this analyzer.
### TDR Adjustments

<table>
<thead>
<tr>
<th>Adjustment</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10MHz Freq Adjustment</td>
<td>Adjusts frequency reference to exactly 10MHz.</td>
</tr>
<tr>
<td>EE Default Adjustment</td>
<td>Initializes or adjusts EEPROM values for several assemblies.</td>
</tr>
<tr>
<td>Source Adjustment</td>
<td>Adjusts source power level vs frequency.</td>
</tr>
<tr>
<td>Synthesizer DAC Adj.</td>
<td>Adj Synth for lowest phase noise.</td>
</tr>
<tr>
<td>IF Gain Adjustment</td>
<td>Adjusts Gain of IF Mux</td>
</tr>
<tr>
<td>Receiver Characterization</td>
<td>Adjusts receiver high level amplitude response.</td>
</tr>
<tr>
<td>Receiver Adjustment</td>
<td>Adjusts all receivers for flat frequency response.</td>
</tr>
<tr>
<td>LFE Receiver Adjustment</td>
<td>Adjusts LFE receivers for flat frequency response.</td>
</tr>
<tr>
<td>IF Response Adjustment</td>
<td>Adjusts IF Freq amplitude, phase, &amp; offsets.</td>
</tr>
<tr>
<td>Noise Adjustment</td>
<td>Adjusts noise IF gain and noise power flatness.</td>
</tr>
<tr>
<td>System Default Cal</td>
<td>Creates system cal for TDR and other uses.</td>
</tr>
</tbody>
</table>

Adjustments should be performed in the order listed (in green.)

All adjustments require that you be an administrator on this analyzer.
- 10 MHz Reference Frequency Adjustment
- EE Default Adjustment
- Synthesizer Bandwidth Adjustment
- Source Adjustment
- Synthesizer DAC Adj.
- IF Gain Adjustment
- Receiver Characterization
- Receiver Adjustment
- LFE Receiver Adjustment
- IF Response Adjustment
- **IF Response Adjustment** (applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above.)

- Noise (Figure) Adjustment

- System Default Calibration

**mmWave System Adjustments**

When a Millimeter mode is detected, the following adjustments are displayed.

- LO Adjustment for mm Mode
- LFE Receiver Adj.
- IF Gain Adj for mm Mode
- Source and Receiver Adj.
- Receiver Characterization
- IF Response Adj. for mm Mode
The performance of the network analyzer is specified in two ways: system specifications, and instrument specifications. It is the end user’s responsibility to determine which set of specifications is applicable to their use of the analyzer.

A network analyzer measurement “system” includes the analyzer, calibration kit, test cables, and any necessary adapters. The system verification software in the analyzer is used to verify the system’s conformance to the “system” specifications. A “pass” result demonstrates that the analyzer, test cables, and adapters, perform correctly as a system. It DOES NOT demonstrate that any one component performs according to its individual specifications. A change to any part of this measurement system requires a re-verification of the system.

Instrument specifications specify the network analyzer’s uncorrected measurement port characteristics and its output and input behavior. The analyzer performance tests are used to verify the analyzer’s conformance to “instrument” specifications.

The system verification utility verifies the analyzer system specifications by automatically measuring the magnitude and phase for all four S-parameters for each verification device, and comparing the values against the following:

- Factory measured data from files on the verification disk
- Limit lines based on the measurement uncertainty

System Verification requires the use of a calibration kit and verification kit which has been certified within the past 12 months by Keysight. System Verification can NOT be used to perform this kit certification.

Operator's Check should also be performed to verify the basic operation of the analyzer.

- Equipment Used in the System Verification
- Precautions for Handling Airlines
- Flow Diagram of Procedure
- Procedure for System Verification
- If the System Fails the Verification Test
- Interpreting the Verification Results
**Notes**

- Although the performance for all S-parameters is measured, the S-parameter phase uncertainties are less important for verifying system performance. Therefore, the limit lines will not appear on the printouts.

- System Verification can NOT be run with a Multiport test set enabled. However, you can run a performance check as described in the Test Set User's Guide. See the N44xx User's Guide.

- The M9485A supports only ports 1 to 4. When the M9485A has any M9377As, system verification is not supported.

### Equipment Used in the System Verification

#### VNA Models with 3.5 mm test ports

<table>
<thead>
<tr>
<th>Equipment Type</th>
<th>3.5 mm</th>
<th>Type-N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calibration kit</td>
<td>85052C/D, 85033D/E</td>
<td>85054D, 85032B/E/F</td>
</tr>
<tr>
<td>or</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECAL Module</td>
<td>N4691A</td>
<td>N4690A</td>
</tr>
<tr>
<td>Verification kit</td>
<td>85053B</td>
<td>85055A</td>
</tr>
<tr>
<td>RF Cable(s)</td>
<td>Single: 85131C/E</td>
<td>Single: 85132C/E</td>
</tr>
<tr>
<td></td>
<td>Pair: 85131D/F</td>
<td>Pair: 85132D/F</td>
</tr>
<tr>
<td>Adapters</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Single: 85130C and one 7mm-to-Type-N from 85054B cal kit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pair: Two 7mm-to-Type-N from 85054B cal kit</td>
</tr>
</tbody>
</table>

#### VNA Models with 2.4 mm test ports

<table>
<thead>
<tr>
<th>Equipment Type</th>
<th>2.4 mm</th>
<th>3.5 mm</th>
<th>Type-N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calibration kit</td>
<td>85056A/D</td>
<td>85052C/D</td>
<td>85054B/D</td>
</tr>
<tr>
<td>or</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECAL Module</td>
<td>N4693A</td>
<td>N4691A</td>
<td>N4690A</td>
</tr>
<tr>
<td>Verification kit</td>
<td>85057B</td>
<td>85053B</td>
<td>85055A</td>
</tr>
<tr>
<td>RF Cable(s)</td>
<td>Single: 85133C/E</td>
<td>Single: 85134C/E</td>
<td>Single: 85135C/E</td>
</tr>
<tr>
<td></td>
<td>Pair: 85133D/F</td>
<td>Pair: 85134D/F</td>
<td>Pair: 85135D/F</td>
</tr>
<tr>
<td>Adapters</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Single: 85130F and two 7mm-to-Type-N from 85054B cal kit</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pair: None</td>
<td>Pair: Two 7mm-to-Type-N from 85054B cal kit</td>
</tr>
</tbody>
</table>

#### VNA Models with 1.85 mm test ports

<table>
<thead>
<tr>
<th>Equipment Type</th>
<th>1.85 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6752</td>
</tr>
<tr>
<td>Equipment Type</td>
<td>1.85 mm</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Calibration kit or ECAL Module</td>
<td>85058B</td>
</tr>
<tr>
<td>Verification kit</td>
<td>N4694A</td>
</tr>
<tr>
<td>RF Cable(s)</td>
<td>Single: N4697E</td>
</tr>
<tr>
<td></td>
<td>Pair:  N4697F</td>
</tr>
<tr>
<td>Adapters</td>
<td>None</td>
</tr>
</tbody>
</table>

**Cable Substitution**

The test port cables specified for the analyzer have been characterized for connector repeatability, magnitude and phase stability with flexing, return loss, insertion loss, and aging rate. Since test port cable performance is a significant contributor to the system performance, cables of lower performance will increase the uncertainty of your measurement. It is highly recommended that the test port cables be regularly tested.

If the system verification is performed with a non-Keysight cable, ensure that the cable meets or exceeds the operation of the specified cable. Refer to the cable User's Guide for specifications.

**Cable Flex Factor**

Flex Factor determines how much of the cable phase uncertainty to include in determining the limit lines.

- Set to 0% (zero) if the cables are held down in a fixture and are not allowed to move during the calibration and verification.
- Set to 100% if the cables are allowed to move a lot.

**Calibration Kit Substitution**

Non-Keysight calibration kits are not recommended nor supported.

**Precautions for Handling Airlines**

When you are using the airlines in the verification kit, observe the following practices to ensure good measurement techniques.

- Be very careful not to drop the airline's center or outer conductor. Damage will result if these devices are
dropped.

- Use proper Electro-Static Discharge (ESD) procedures.
- Clean your hands or wear gloves as skin oils will cause a change in electrical performance.

Flow Diagram of Procedure

The operational flow of the software is depicted by the flowchart shown below.
Procedure for System Verification

1. If you want printed test outputs, connect a printer to the analyzer. Let the analyzer warm up for at least 30 minutes.
2. Insert the verification kit USB memory into the analyzer USB port.

3. Press **System** > **Service** > **Verification** > **System Verification**. The System Verification window similar to this will be displayed.

**System Verification Dialog**

4. In the **Calibration Kit** box, select the calibration kit or ECal module that is being used. The corresponding verification kit to use appears in the **Verification Kit** box.

5. Under **Printer Output** click on any of the following options.

   - **Print Tabular Data**: Prints the verification data in tabular form which includes measured data and uncertainty limits. Refer to a tabular data example, later in this topic.

   - **Print Graphs**: Prints the verification data in graphical form. The graphic form includes the measured data trace, factory supplied data trace and uncertainty limits. Refer to a plot data example, later in this topic.

   - **File Tabular Data**: Writes the verification data in tabular form to a text file in the D:\ directory.

   - **File Graphs**: Saves a screen image in .PNG format in the D:\ directory.

**Note**: If you want printed output, it is assumed you have already installed the Windows driver for your particular printer, and have tested that you can print to the printer from the network analyzer. This software is designed to print to whichever printer is currently set as the Default printer (see Printers in the Windows Control Panel).
6. To modify the number of ports to be verified, to change the number of devices to measure, or to use a previously stored verification calibration, click on the Configure tab and make the desired selections.
   - For the system verification to be truly adequate, the software must measure all devices in the kit with a recent calibration applied. Removing and reattaching any test port cables or adapters invalidates all previous calibrations.

7. Click Run.

8. Follow the instructions on the analyzer for performing the system verification, inserting the verification devices as prompted.

**Note for 3 Port analyzer:**
The System Verification Procedure is repeated three times. The first time, **Ports 1 and 2** are measured as a pair; then **Ports 1 and 3** are measured; and lastly, **Ports 2 and 3** are measured.

**Note for 4 Port analyzer:**
The System Verification Procedure is repeated two times. The first time, **Ports 1 and 2** are measured as a pair, then **Ports 3 and 4** are measured.

---

**Step-by-Step Process Description**

1. Depending upon the selected choice in the Calibration submenu of the Configure menu, the user is either prompted to recall a previous calibrated instrument state, or is guided through a full 2-port calibration using the selected calibration kit. For ECal, the ECal module is connected just once; a standby message is posted while the software is performing the calibration.

2. The user is prompted to connect the first verification device.

3. The software reads the factory measured data for that device and uncertainty values for that data (CITIfiles) from the floppy disk supplied with the verification kit.

4. The software sends the factory measured data, calibration kit and instrument state information to the uncertainty calculator DLL, which generates uncertainty values specific to the analyzer.

5. The analyzer first sets up for magnitude measurements of all four S-parameters, each parameter in a separate window (lin mag for $S_{11}$ and $S_{22}$, log mag for $S_{21}$ and $S_{12}$). Each of the factory measured S-parameters are fed to the appropriate window as a memory trace. Limit line offsets are calculated as the sum of the factory measured data uncertainties and analyzer uncertainties reported by the DLL. Upper and lower limits are displayed (factory measured data + uncertainty sum, factory measured data - uncertainty sum). The analyzer takes a sweep, limit test is turned on and PASS/FAIL status is reported in each of the four windows.

6. The user clicks a button when ready to view phase measurements. The four windows get updated for phase format, phase memory traces, phase limits and PASS/FAIL result.

7. If the limit test of any of the four S-parameters (magnitude or phase) indicates a FAIL status, the software
suggests troubleshooting tips and asks if the user would like to repeat measurement of that device or proceed to the next device. If proceeding to the next device, the factory measured data and uncertainties for the next device are read from floppy, the uncertainty DLL gets called with this next set of factory measured data, and the four measurement windows get updated for magnitude measurement of the next device.

8. The software follows this same process until all selected devices have been measured, at which point a summary window is displayed containing the set of PASS/FAIL results for all four parameters of each device.

**If the System Fails the Verification Test**

**IMPORTANT:** Inspect all connections. Do not remove the cable from the analyzer test port. This will invalidate the calibration that you have done earlier.

1. Repeat this verification test. Make good connections with correct torque specifications for each verification device.

2. Disconnect, clean and reconnect the device that failed the verification test. Then measure the device again.

3. If the analyzer still fails the test, check the measurement calibration by viewing the error terms as described in "Front Panel Access to Error Terms" on page 4-7 of the Service Guide.

4. Refer to the graphic below, for additional troubleshooting steps.

**Verification Fails Flowchart**
Interpreting the Verification Results

The graphic below shows an example of typical verification results with **Tabular Data** selected in the **Printer Output** area of the **System Verification** window. A graphic later in this topic shows an example of typical verification results with **Measurement Plots** selected in the **Printer Output** area of the **System Verification** windows. These printouts include a comparison of the data from your measurement results with the traceable data and corresponding uncertainty specifications. Use these printouts to determine whether your measured data falls within the total uncertainty limits at all frequencies.

The tabular data consists of:

- Frequency of the data points (in MHz).
- Lower limit line as defined by the total system uncertainty specification.
- Results of the measurement.
- Upper limit line as defined by the total system uncertainty specification.
- Test status (PASS or FAIL) of that measurement point.
The printed graphical results show:

- Upper limit points as defined by the total system uncertainty specifications.
- Lower limit points as defined by the total system uncertainty specifications.
- Data measured at the factory.
- Results of measurements.
- Measurement parameter names and formats (Lin Mag or Log Mag).
- Serial number of device (00810).
- Device being measured (Sys Ver 20 dB attenuator).
Tip: Use Move App to Back to cause the VNA application to move behind this application on the screen.

Overview

The Operator's Check should be performed when you first receive your VNA, and any time you wish to have confidence that the VNA is working properly.

Notes

- The Operator's Check does not verify performance to specifications. To verify VNA performance to specifications, run System Verification.
- Allow the VNA to warm up for 90 minutes before considering a failed test to be valid.
- The Operator's Check can NOT be run with a Multiport test set enabled. However, you can run a performance check as described in the Test Set User's Guide. See the N44xx User's Guide.

The Pass/Fail criteria used in the Operator's Check identifies obvious failures in the following portions of the VNA hardware:

- Repeatability of the RF switch in the test set
- Attenuation ranges of the test port attenuators (if installed).
- Calibration of the receivers
- Frequency response of the receivers
- Phase lock and leveling
- Noise floor and trace noise
How to Run the Operator's Check

Using Hardkey/SoftTab/Softkey

1. Press **System** > **Service** > **Operator's Check**.

1. Follow the instructions to proceed with the test.

Standard Dialog

The following dialog is displayed for standard VNAs.

Wideband Millimeter Wave Dialog

The following dialog is displayed for wideband millimeter wave systems.
This dialog box will look slightly different, depending on VNA model number and installed options. Some of the tests are not run if the appropriate option is not installed.

To learn about how each test is performed, click one of the tests on the right of the dialog. For example, the following information dialog is launched when **Leveling** is clicked:
Operators Check dialog box help

Note: It is normal for a momentary unleveled condition to appear during portions of the Operators Check.

Configure

Prompt for attachment of Short / Open  If you do not have enough shorts or opens for all test ports, you will be prompted to move the standard to the next test port. Connect either a short or open to port 1, then click Begin.

Shorts / Opens are attached to all ports  Connect either a short or open for each test port, then click Begin. All ports are tested without interruption. You can mix shorts and opens on the test ports.

VNA  Shows information about the VNA that is being tested.

Legend  Shows the status icons used in the Operator's Check and their meaning. Pending Pass means that a portion of the testing has been completed successfully.

Results  Shows the current status of each test. Click on the test name to learn how that test is performed. This may help in troubleshooting failed tests. If any tests Fail, refer to Chapter 3 of the VNA service guide.

Begin  Starts the Operator's Check.

View Results  Shows all results in text format. Failed items are preceded by ====》》》.

This text file can be printed or saved with a unique file name to compare results with previous or subsequent testing.

Exit  Ends the program and closes the window.
**Display Test**

The VNA screen should be bright with all annotations and text readable. The display test allows you to check for non-functioning pixels and other problems.

**Note** If the display is dim or dark, refer to “Troubleshooting LCD Display Problems” in the VNA Service Guide.

**What Is a Damaged Pixel?**

A pixel is a picture element that combines to create the image on the display. They are about the size of a small pin point. Damaged pixels can be either “stuck on” or “dark.”

- Stuck on pixel - red, green, or blue; always displayed regardless of the display setting. It will be visible on a dark background.
- Dark pixel - always dark; displayed against a background of its own color.

**How to Run the Display Test**

Press **System > Service > Diagnostics**, and then click **Display Test...**

A multi-color screen is displayed. Be prepared to look for the symptoms described below. Click the Start Test button. To continue to the next test, click the moving Next Test button. The button moves to allow you to see all of the display. After the test is completed, the display defaults to the network analyzer screen.

**How to Identify a Faulty Display**

One or more of the following indicate a bad display:

- Complete row or column of “stuck on” or “dark” pixels
- More than six “stuck on” pixels (but not more than three green)
- More than twelve “dark” pixels (but not more than seven of the same color)
- Two or more consecutive “stuck on” pixels or three or more consecutive “dark” pixels (but no more than one set of two consecutive dark pixels)
- “Stuck on or “dark” pixels less than 6.5 mm apart (excluding consecutive pixels)

If any of these symptoms occur, your display is considered faulty. See the Service Guide for your VNA.
model.
10 MHz Reference Frequency Adjustment

This routine adjusts the analyzer's internal time-base to exactly 10 MHz by changing a DAC value. This DAC value is stored in the analyzer's non-volatile memory. This routine should only be necessary in the following situations:

- The frequency reference assembly is replaced.
- The 10 MHz reference has drifted significantly from the factory adjusted value.

**WARNING:** The range of this adjustment is only about 20 Hz. It is highly recommended that a very accurate frequency standard be used to measure this 10 MHz signal.

**Frequency Counter Compatibility**

This procedure uses SCPI commands (over GPIB) to communicate with the frequency counter. It should work with the Keysight R5313xA, 5315xA, 53181A series of counters as well as the older 5350 series.

If no compatible counters are available, select the "Manual" mode of operation.

**Procedures**

**Note:** You must be logged onto the VNA as an Administrator to perform an adjustment. [Learn more.]

Click **Utility**, then **System**, then **Service**, then **Adjustment Routines**...

At the Adjustments selection, click **10 MHz Frequency Adjustment**

**Procedure for GPIB Counters Only**

1. Connect the analyzer rear panel 10 MHz Reference output to the frequency counter.
2. Connect a GPIB cable from the analyzer to the counter. Make sure no other controllers are active on the same connection.
3. If applicable, connect the house frequency standard to the counter reference input.
4. Set the counter GPIB address to 03. Ensure that the counter is the only device at this address.
5. On the VNA, press **System** > **Service** > **Adjustment Routines**..., then **10 MHz Freq. Adjust**.
6. Click **Begin Adj.** The application adjusts the internal reference for minimal error and stores the results.

7. Click **Read Freq** to trigger another reading of the 10 MHz signal.

8. Read the current DAC value stored in the analyzer's non-volatile memory (value = 0 - 4095).

9. When the status area indicates the adjustment is complete, click **Exit**.

**Procedure for Non-GPIB Counters**

1. Connect the counter input to the rear panel 10 MHz Reference Output.

2. Set the counter to at least 1 Hz resolution.

3. If applicable, connect the house-frequency standard to the counter reference input.

4. In the analyzer **System > Service > Adjustment Routines...**, then click **10 MHz Freq. Adjust**

5. Under **Frequency Counter**, select **Manual**.

   - Adjust the slider bar **arrows** until the frequency counter reads 10.0 MHz at your desired level of accuracy.
   - Click **Exit** to save the results.

**Data Storage**

- The correction data is stored in the EEPROM on the 10MHz Ref board assembly.

**Note:** If the counter is misreading the frequency, it may be necessary to attenuate the input, or set the input impedance to 50 ohms, or both.
Source Adjustment

Source Adjustment is a **SERVICE** Routine which should be performed when a component in the source chain is replaced, or when the VNA fails an annual calibration. It adjusts the VNA source power for flatness across its full frequency range.

This topic does **NOT** discuss **Source Power Calibration**, which calibrates a VNA source over the current measurement range.

Required Equipment

**Note:** The power sensor depends on the VNA frequency range. Depending on the VNA model, two power sensors may be required to test the full frequency range. The VNA front panel connector type will determine the cable used and if an adapter is required with the power sensor(s).

See list of supported power meters and sensors.

See VNA Accessories

Procedure

**Note:** You must be logged onto the VNA as an Administrator to perform an adjustment. [Learn more.]

1. Refer to your power meter documentation to ensure the proper calibration factors for the power sensor have been entered into the table on the power meter.

2. Connect a GPIB cable between the power meter and network analyzer (use the System Controller GPIB port if applicable.)

3. Ensure the power sensor(s) are connected to the power meter.

4. Click **Utility**, then **System**, then **Service**, then **Adjustment Routines**...

5. At the Adjustments selection, click **Source Adjustment**

6. There are 3 different version of the Source Calibration software; all are slightly different. All have a button that is labeled "Calibrate" or "Adjust". This is the button that will begin the calibration process. Some versions will also have a button labeled "Verify" that will test the source calibration without making any changes. Other selections are for factory personnel use only.

7. Once begun, you must enter the power meter and sensor information. The software will verify the power meter and sensor. You are then prompted to connect the sensor(s) and cable as needed.

**Connecting sensors to the VNA**
Additional Information

All ports are tested on all VNAs. Source calibration takes approximately 10 to 45 minutes to complete depending on the frequency range and model number of the VNA.

Troubleshooting

In the event there is a problem with Source Adjustment, please refer to the "Troubleshooting" chapter in the VNA Service Guide.

Data Storage

- The correction data is stored in the flash memory on the Test Set Mother Board.
IF Gain Adjustment

This adjustment sets the IF attenuator/gain stages to present a relatively consistent level to the ADC versus frequency for a given amount of signal. This maximizes the dynamic range of the instrument by preventing any IF compression at higher levels. The adjustment range is about 18dB with one adjustment value per band per input: R1, R2, R3, R4, A, B, C, and D (if applicable.)

The program first finds the peak of each band’s frequency response with a fixed input level. It then re-measures those peaks with a power meter. From this information, it determines the optimum gain or attenuation needed to keep all bands as similar as possible for any given frequency.

When to perform

IF Gain Adjustment should be performed when any of the following occur:

- An assembly in the reference receiver path (R1,R2) is replaced.
- The Test Set Motherboard is replaced
- The IF Mux assembly or SPAM assembly is replaced

Required Equipment

- Power Meter and Sensor(s) - These must cover the full frequency range of the VNA.
- GPIB cable, LAN, or USB cable depending upon the type of Power Meter/Sensor being used.
- Adapters as needed

Procedure

Note: You must be logged onto the VNA as an Administrator to perform an adjustment. Learn more.

Click Utility, then System, then Service, then Adjustment Routines...

At the Adjustments selection, click IF Gain Adjustment.
Note: Connect the Power Meter/Sensor to the VNA USB, GPIB, or LAN input depending on the type of Power Meter/Sensor being used. The program will prompt you where and when to attach the RF input for the Power Sensor(s).

Ensure the Reference Channel paths are properly configured and the connections are properly torqued.

1. Click **Begin**

2. Follow the instructions displayed in the program.

The adjustment takes about 5 minutes to complete for a 26.5 GHz VNA. Higher frequency units may take longer.

The Default menu selection (under the advanced menu) is for factory personnel only. This will preset all values to default levels for trouble shooting purposes only. If this is selected, a full IF gain adjustment will need to be performed.

**Data Storage**
- The correction data is stored in the flash memory on the Test Set Mother Board.
Receiver Adjustment

This program adjusts the network analyzer receivers for a flat response across its full frequency range. This adjustment is for service only; not for measurement calibration.

Required Equipment

See list of supported power meters and sensors.

See VNA Accessories

Notes

- The power sensor depends on the VNA frequency range. Depending on the VNA model, two power sensors may be required to test the full frequency range. The VNA front panel connector type will determine the cable used and if an adapter is required with the power sensor(s).

- In this adjustment, a power sensor with a specified lower frequency limit of 50 MHz may be used on all instrument that have a lower frequency limit of 10 MHz. Any added uncertainty is negligible.

- If using an older style sensor (without built-in correction factors), refer to your power meter documentation to ensure the proper calibration factors for the power sensor have been entered into a table on the power meter.

- You must be logged onto the VNA as an Administrator to perform an adjustment. Learn more.

Procedure

1. Click Utility, then System, then Service, then Adjustment Routines...

2. At the Adjustments selection, click Receiver Adjustment

3. Connect a GPIB cable between the power meter and network analyzer.

4. Ensure the power sensor(s) are connected to the power meter.

5. The software presents you with two choices:

   a. Click Inspect Flatness to observe flatness of receiver response versus frequency. Although there is no
explicit specification for receiver flatness, Receiver Calibration should improve Transmission and Reflection Tracking error terms which are specified.

b. Click **Calibrate** to begin the receiver calibration process. The software prompts you to connect the sensor(s), cable and adapter as needed (see the following graphics).

**Connecting sensor(s) to the VNA**

**Connecting adapter and cable between sensor and VNA**

**Through connection using the specified cable**
**Additional Information**

- Receiver Adjustment tests all VNA receivers, taking approximately 15 and 45 minutes. Length is dependent on frequency range and number of ports.

- Upon completion of the Receiver Adjustment, a transmission measurement of a good quality cable should appear to be smooth, with slightly increasing loss versus frequency. A reflection measurement of a short or open should appear to be a flat line across the entire frequency range with only a dB or two of variation/ripple. If instead, you see variations of 10-40dB, then the VNA may have a mixer problem. Typically, this means the uncorrected low end frequency phase relative to other receivers is different. This causes the correction algorithm to "blow up" and provide wildly incorrect data. This is almost always a hardware problem and typically one or more receivers must be replaced. The adjustment procedure has a quick test for this and it will show a warning message if excess phase shift is detected, however this test is not definitive and may not always catch every problem.

**Troubleshooting**

In the event there is a problem with Receiver Adjustment, please refer to the "Troubleshooting" chapter in your VNA Service Guide.

**Data Storage**

- The correction data is stored in the flash memory on the Test Set Mother Board.
The Receiver Display as a Troubleshooting Tool

The Receiver Display is a Troubleshooting Tool. It enables the analyzer to isolate faulty functional groups within its own Measurement System. Traces for each Receiver are Displayed in individual windows. Identifying discrepancies of the traces in these windows can help isolate the faulty assembly.

For a thorough description of Receiver Display and the troubleshooting steps see the Service Guide for your VNA. You can download the Service Guide from our website: http://na.support.keysight.com/pna/ or http://na.support.keysight.com/pxi/

How to Start the Receiver Display

**Using Hardkey/SoftTab/Softkey**

1. Press **System > Service > Diagnostics > Receiver Display...**

**Using a mouse**

1. Click **Utility**
2. Select **System**
3. Select **Service**
4. Select **Diagnostics**
5. Select **Receiver Display...**
Mechanical Counter

This feature keeps track of mechanical attenuator and switch changes. The firmware increments counters each time commands to change the attenuator or switch settings are sent. The dialog reads back the counters and allows a technician to reset the counters if the hardware components are replaced. This feature exists on all PNAs running firmware A.13.60 and above. There are no user SCPI commands to read these counter values remotely.

- To read/reset counters, press **System > Service > Diagnostics > Mechanical Counter...** The Mechanical Cycle Counter dialog is displayed:

![Mechanical Cycle Counter Dialog](image)

- Select the device to reset by clicking on its radio button located on the left side of the device name then click on the **Modify** button. The **Model and Serial Entry** dialog is displayed:
Follow the instructions in the dialog.

- Enter as much data as can be found for the new assembly and enter the count in the Enter Desired Count numeric field.
IF Path Configuration Settings

This dialog is used to set many IF receiver settings for models.

In this topic:

- VNA-X and N522xB IF Frequencies
- How to Start the IF Path Configuration dialog
- IF (Receiver) Path Configuration dialog box
- Expanded Block Diagram and Descriptions
- IF Path Configuration using COM and SCPI

Other IF Access Topics

Auto IF Frequencies

Wideband/Normal IF path:

Note: For the following discussion, RF = Receiver Frequency

With DSP Version 4:

- RF < 53 MHz: IF = 2.535211 MHz  \([3 \times (60e6 / 71)]\)
- RF >= 53 MHz: IF = 7.605634 MHz  \([9 \times (60e6 / 71)]\)

With DSP Version 5, the IF frequency is dependent on the RF AND the current IFBW setting:

- All RF; IF Bandwidth >= 1MHz: (All Models)
<table>
<thead>
<tr>
<th>IFBW Setting</th>
<th>IF Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 MHz</td>
<td>7.692 MHz</td>
</tr>
<tr>
<td>1.5 MHz</td>
<td>7.368 MHz</td>
</tr>
<tr>
<td>2 MHz</td>
<td>8.450 MHz</td>
</tr>
<tr>
<td>3 MHz</td>
<td>8.163 MHz</td>
</tr>
<tr>
<td>5 MHz</td>
<td>6.897 MHz</td>
</tr>
<tr>
<td>7 MHz</td>
<td>10.53 MHz</td>
</tr>
<tr>
<td>10 MHz</td>
<td>15.38 MHz</td>
</tr>
<tr>
<td>15 MHz</td>
<td>22.22 MHZ</td>
</tr>
</tbody>
</table>

- IF Bandwidth <= 600 kHz:
  - RF >= 53 MHz; All models: IF = 7.438017 MHz \([(9 \times (100e6 / 121)]\]
  - RF <= 53 MHz; PNA-X models: IF = 2.479339 MHz \([(3 \times (100e6 / 121)]\]
  - RF <= 53 MHz; N522xB models: IF = 826.446 kHz \([1 \times (100e6 / 121)]\]

**Narrowband IF path:**

- IF = 10.70 MHz
- Bandwidth = 30 kHz

**Note:** The IFBW is limited to 600 kHz when performing Swept IMD measurements even if the **Wide IF path** is selected.

**Manually change the IF frequency**

**The IF frequency** can be changed to any value between +14.9999 MHz and -14.9999 MHz using **SENS:IF:FREQ** or **IFFrequency** commands.

- With DSP Version 4 - 34 and above, min and max IF frequencies up to +/- 20.1 MHz are available.
- With DSP Version 5, min and max IF frequencies up to +/- 38 MHz are available.
Performance is degraded drastically above +/- 14.9999 MHz.

Learn about DSP Version

How to start the IF Path Configuration dialog

Using **Hardkey/SoftTab/Softkey**

1. Press **Setup > Internal Hardware > IF Path Config...**

Using a mouse

1. Click **Instrument**
2. Select **Setup**
3. Select **Internal Hardware**
4. Select **IF Config...**

**Programming Commands**

**IF (Receiver) Path Configuration** dialog box help

![Diagram of IF Path Configuration](image)

The IF path, represented in the block diagram at the top of the dialog, is duplicated for each of the receivers (A, B, C, D, R1, R2, R3, R4). In addition, each path can be configured differently for each channel.

Element - Indicates an element in the expanded block diagram.

**IF Input** (1) - Available on the PNA-X and N522xB with Opt 020. Internal input is a test port or
reference receiver input. External Input is through the PNA-X and N522xB rear-panel connectors.

**IF Attenuator** (3) - Specify IF attenuation for the narrowband path of the selected receiver.

**IF Filter** (2) - Select Wideband or Narrowband (includes the ability to pulse gates).

**IF Gain** (10) - Set to **Auto** by default, the following are reasons to change the IF Gain:

- For **millimeter systems** that do NOT use the external millimeter test set, the millimeter head output IFs are routed directly to the RF receivers. You may want to change the IF gain to improve the noise figure of the receivers.

- External couplers are often used for high-power test setups. The VNA automatically adjusts the IF Gain depending on the frequency of operation in order to correct for several aspects of the hardware, including the large coupled-arm roll off at low frequencies (below 700MHz). If you replace the internal coupler with one having a different low-frequency roll off, then you may also want to change the IF gain to avoid overdriving the receiver.

- When using the rear panel direct IF inputs, the gain is set low when in standard operation and very high when in **millimeter mode**. You may want to control the gain of the direct IF inputs to improve noise figure or to avoid overdriving the receivers.

**ADC Filter** (5) - Select **Auto**, **Narrow** (9 MHz or 11 MHz), or **Wide** (16 MHz or 38 MHz) filter. Learn more.

**Couple all IF paths** - Check to make the same setting for all receivers.

Expanded Block Diagram (IF / Pulse Generators / DSP)
Blue boxes are configurable elements. Click a blue box, or scroll down, to see how to make settings using SCPI and COM commands.

**Receiver / IF Path (top block)**

Scroll up for descriptions of the Receiver / IF Paths blocks. Most of these elements can be set from the front-panel User Interface (UI).

**ADC:** (Analog to Digital Converter) This block, responsible for quantifying receiver measurements, is triggered when BOTH the Meas Trig line AND P0 pulse generator line are TRUE. In addition, the Meas Trig signal MUST be TRUE before or at the same time as the P0 signal.

**Meas Trig Signal**

- When VNA Trigger source is set to **Internal**, the Meas Trig line is ALWAYS TRUE. Internal trigger source is the best setting for making Pulse measurements. This means that the P0 line determines when pulse measurements are made.
- When VNA Trigger source is set to **External**, the Meas Trig line can be configured from the Meas Trig (External) dialog.

**P0 (Trig) Signal** - See below.

**Pulse Generators (middle block in above diagram)**
See how to make these settings remotely

**Switch 6:** Represents **Internal** or **External** triggering for the pulse generator. This setting, and the External Trigger Pulse settings, are made on the Trigger dialog, **Pulse Trigger tab.**

On the Integrated Pulse App, **Pulse Setup dialog,** when Pulse Trigger source is set to External, then External is selected automatically in the Pulse Gen dialog.

- **Internal** - The pulse generator is internally triggered and puts out a periodic pulse train with a period defined by the pulse generator settings.

- **External** - The pulse period is ignored, and the pulse generator puts out one set of pulses (P0-P4) per external trigger. All five pulse outputs have unique delay and pulse width settings.

  External trigger input is on the **Pulse I/O connector** pin 7 (PulseSyncIn). The PulseSyncIn line provides a trigger signal into the Pulse Generators. If a level trigger is still valid when the first set of pulses is finished, another set will be generated. Only one set of pulses is emitted when edge triggering is used.

  The External pulse input polarity (positive or negative) and type (edge or level) is configurable only with DSP version: 4.0 FPGA: 34 or higher. Learn more. Otherwise, the pulse generators respond only to positive, level input trigger signals.

**P0:** When P0 is enabled, it is hardwired to trigger the data acquisition ADCs. See ADC (above).

- If the data acquisition system is not ready (Meas Trig = NOT TRIGGERED), the P0 trigger is ignored.

- If the pulse generator is internally triggered, then the data acquisition system receives periodic triggers.

- If the pulse generator is externally triggered, then the data acquisition system receives a trigger each time the pulse generator is triggered.

Data acquisition is synchronized to the pulse generator **ONLY** when P0 is enabled. This is equivalent to enabling **Synchronize ADCs using pulse trigger** on the Pulse Generators Setup dialog. Data acquisition begins on the rising edge of P0. The width of P0 does NOT directly matter as data acquisition does not stop when P0 goes false. The following describes how the P0 generator triggers data acquisition:

- **Step mode sweeps** of any sweep type: By default, each P0 rising edge triggers a single data point. When **point averaging** is on, all of the measurements (subpoints) that are required to average each point are made with a single trigger. To individually trigger the acquisition of each subpoint, send the subPointTrigger (SCPI
or COM) command.

- **CW sweeps**: Each trigger initiates acquisition for the entire sweep. This is currently used for wideband pulse profiling.

**P1 thru P4** These four pulse generator outputs are hardwired to rear panel outputs on the Pulse I/O connector (pins 10 - 13). They are also routed to two switches (#4 and #7 on the above diagram) along with the following three lines:

  - **Rear Panel**  External pulse generator input from Pulse I/O connector pin 8 (RFPulseModIn).
  
  - **OFF**  Pulse is constantly in LOW state causing gate and source to be OFF.
  
  - **ON**  Pulse is constantly in HIGH state causing gate and source to be ON.

**Switch 7** Pulse Modulation - 1 of 7 lines to each of the sources. **Important**: When internally modulating the sources, source leveling must be set to Open-loop.

**Rear-panel Outputs**: Pulse I/O connector (pins 10 - 13) hardwired.

**Source1 and Source2 pulse modulators**: (#8 and #9 on the above diagram)

**DSP (bottom block)**

  - Filters the ADC (digital) output from top block and outputs data to the VNA display.

  - See SCPI and COM commands to control DSP settings.

**See Also**

  - Pulse Settings
  
  - Remote RF Path Configuration
  
  - Rear Panel Pulse I/O connector

**IF Path Configuration using COM and SCPI**

Most of the following elements, highlighted in BLUE in the above Block Diagram, have settings that are made using SCPI or COM commands. In general, the command specifies an element name and a
setting.

- See SCPI command
- See COM object and example.
- These are the same commands that are used to make settings in the RF Path Configurator.

<table>
<thead>
<tr>
<th>Ref#</th>
<th>Element Name</th>
<th>Description</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&quot;IFSWn&quot;</td>
<td>For 2-port PNA-X and N522x, n = A, B, R1, R2 For 4-port PNA-X and N522x, n = A, B, C, D, R (for R1 to R4) For example: &quot;IFSWB&quot; Requires Opt 020 external IF inputs on the rear panel</td>
<td>&quot;Internal&quot; &quot;External&quot; Rear Panel IF connectors. 4-port use R for Ref 1 to 4</td>
</tr>
<tr>
<td>2</td>
<td>&quot;IFPathn&quot;</td>
<td>For 2-port PNA-X and N522x, n = A, B, R1, R2 For 4-port PNA-X and N522x, n = A, B, C, D, R1, R2, R3, R4 &quot;IFSigPathAll&quot; makes setting for ALL receivers.</td>
<td>&quot;WBF&quot; Wide Band Filter Path (default) &quot;NBF&quot; Narrow Band Filter Path</td>
</tr>
<tr>
<td>3</td>
<td>&quot;NBFATNn&quot;</td>
<td>For 2-port PNA-X and N522x, n = A, B, R1, R2 For 4-port PNA-X and N522x, n = A, B, C, D, R1, R2, R3, R4 For example: &quot;NBFATNB&quot;</td>
<td>0 to 31 in 1 dB steps For example: &quot;28&quot;</td>
</tr>
<tr>
<td>4</td>
<td>&quot;IFGaten&quot;</td>
<td>For 2-port PNA-X and N522x, n = A, B, R1, R2 For 4-port PNA-X and N522x, n = A, B, C, D, R1, R2, R3, R4 For example: &quot;IFGateB&quot;</td>
<td>&quot;On&quot; Gate is always ON &quot;Off&quot; Gate is always OFF &quot;RearPanel&quot; (use Pulse IO pins 1 to 5) &quot;Pulse1&quot;</td>
</tr>
</tbody>
</table>
### 5 IFAntiAliasFilter

- This filter is labeled **ADC Filter** on the IF Path Configuration dialog.
- This setting affects ALL receivers. It can NOT be made for individual receivers.
- The ADC Filter values depend on the DSP Version. Learn more.

### 6 PulseTrigInput

**Requires Opt S93025A/B - Four Internal Pulse Generators**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Internal</strong></td>
<td>Internal Pulse In - pulse generators are triggered each period.</td>
</tr>
<tr>
<td><strong>External</strong></td>
<td>External Pulse Synch In - Pulse I/O pin 7 - An external trigger signal is required to trigger the pulse generators for each pulse.</td>
</tr>
</tbody>
</table>

<name> An external pulse generator to be configured as the Primary Pulse Trigger.

### 7 PulseModDrive

Select from 1 of 7 lines to modulate the OUT1 path of Sources 1 and 2.

**Important:** When Pulse 1-4 is selected to

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>On</strong></td>
<td>Pulse Mod drive is always ON, leaving &quot;SRC1</td>
</tr>
</tbody>
</table>
| **Off** | Pulse Mod drive is always
modulate the sources, *source leveling must be set to Open-loop.*

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>8</strong></td>
<td>&quot;Src1Out1PulseModEnable&quot;</td>
</tr>
<tr>
<td>Requires Opt 021 - Source1 Pulse Modulator</td>
<td>&quot;Enable&quot;</td>
</tr>
<tr>
<td>Requires Opt 022 - Source2 Pulse Modulator</td>
<td>&quot;Disable&quot;</td>
</tr>
<tr>
<td><strong>9</strong></td>
<td>&quot;Src2Out1PulseModEnable&quot;</td>
</tr>
<tr>
<td><strong>10</strong></td>
<td>&quot;IFGAIN&quot;&quot;n&quot;</td>
</tr>
</tbody>
</table>

For 2-port models, n = A, B, R1, R2

For 4-port models, n = A, B, C, D, R1, R2, R3, R4

For example: "IFGAINB"

OFF, leaving "SRC1|2 Out 1" OFF.

"RearPanel" (use Pulse IO pin 8)

"Pulse1"

"Pulse2"

"Pulse3"

"Pulse4"
This feature, when used with the N526xA test sets and external mmWave Modules, extends the frequency coverage of your VNA. The N5251A broadband system is also configured using the Millimeter Module Configuration dialog.

In this topic:

- Features and Limitations
- How to Configure Millimeter Modules
- Mixer Mode
- Spectrum Analyzer mmWave Measurements
- mmWave Module Power Level Control
- Using Cal All Wizard for Supported mmWave Measurement Classes

See Also

- N5261A / N5262A User Manual (Requires internet connection)
- N5251A Installation Guide (Requires internet connection)
- Millimeter-Wave Network Analyzers Technical Overview (Requires internet connection)
- Direct Connect: mmWave Measurements with No Test Set
- Download a macro for Configuring Frequency Extenders (Requires internet connection)

Other IF Access Topics

Note: In the VNA user interface and in this help file, the N526xA Millimeter Head Controller is referred to as a test set. The test head modules are referred to as mmWave modules.

CAUTION: Turn OFF test set power before connecting or disconnecting the DC cable to the mmWave modules.

Features
Controls N5261A and N5262A Test Sets.

Compatible with iTMSA (True Mode Stimulus).

Several methods available to provide Leveled power to the DUT Input

Compatible with Integrated Pulse Application

The following Applications are supported:

- SMC mixer measurements.
- IMSpectrum and IMSpectrum for Converters
- Spectrum Analyzer

The following configurations are supported:

- PNA-X or N522xA with options Opt 020 works with N5261A and N5262A Test Sets.
- When using the N5262A test set, a 2-port PNA-X or N522xB requires Opt S93551A/B.
- Broadband - single sweep from 10 MHz to 110 GHz.
- Banded - frequency coverage divided into bands.

Limitations

- **Power Settings** Your mmWave modules may have a variable attenuator on them. When used with an N5261A or N5262A, after performing a Source Power Cal, then the VNA power settings may be used to control the power into the DUT. See Leveled Power Capabilities.

- To protect your mmWave modules from damage, the settings on the Millimeter Module Configuration dialog can ONLY be changed manually. They can NOT be reset or changed by performing a Preset, by recalling an Instrument State, or from a remote program.

- ONLY the Applications listed above are supported.

Broadband and Banded mmWave Measurements

Broadband refers to mmWave configurations with a frequency range of 10 MHz to 110 GHz. This configuration spans the entire frequency range in a single sweep.

Banded refers to any configuration that is not a broadband configuration and is waveguide based.
Frequency extenders cover frequency ranges from 50 GHz to 1.1 THz. The supported solutions can be configured for different frequency bands with or without a test set controller depending on the measurements required and the frequency extenders being used.

The following table shows the waveguide designation equivalents.

<table>
<thead>
<tr>
<th>MIL-DTL-85/3C</th>
<th>IEEE Std 1785.1</th>
<th>Frequency Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>WR-15</td>
<td>WM-3759</td>
<td>50 GHz to 75 GHz</td>
</tr>
<tr>
<td>WR-12</td>
<td>WM-3099</td>
<td>60 GHz to 90 GHz</td>
</tr>
<tr>
<td>WR-10</td>
<td>WM-2540</td>
<td>75 GHz to 110 GHz</td>
</tr>
<tr>
<td>WR-08</td>
<td>WM-2032</td>
<td>90 GHz to 140 GHz</td>
</tr>
<tr>
<td>WR-06</td>
<td>WM-1651</td>
<td>110 GHz to 170 GHz</td>
</tr>
<tr>
<td>WR-05</td>
<td>WM-1295</td>
<td>140 GHz to 220 GHz</td>
</tr>
<tr>
<td>WR-04</td>
<td>WM-1092</td>
<td>170 GHz to 260 GHz</td>
</tr>
<tr>
<td>WR-03</td>
<td>WM-864</td>
<td>220 GHz to 330 GHz</td>
</tr>
<tr>
<td>WR-02</td>
<td>WM-570</td>
<td>330 GHz to 500 GHz</td>
</tr>
<tr>
<td>WR-1.5</td>
<td>WM-380</td>
<td>500 GHz to 750 GHz</td>
</tr>
<tr>
<td>WR-1.0</td>
<td>WM-250</td>
<td>750 GHz to 1.1 THz</td>
</tr>
</tbody>
</table>


**PNA-X Notes**

- The PNA-X or N522xB rear panel IF Inputs use 5 SMA connectors. Previous VNA models use BNC connectors. Adapters may be required.

- Frequency Offset and SMC Measurements are supported when using mmWave modules. Learn more.

- Beginning with A.10.45, Spectrum Analyzer Measurements are supported when using mmWave modules. Learn more.

**How to Configure Millimeter-Wave Modules**

**Using a mouse**

1. Click **Instrument**
2. Select **Setup**
3. Select **External Hardware**
4. Select **Millimeter Config...**
**Note:** To protect your mmWave modules from damage, settings on this dialog can ONLY be changed manually or with a remote program. They can NOT be reset or changed by performing a Preset, or by recalling an Instrument State.

**Available Configurations** Lists the Standard VNA configuration and other configurations that you have created.

- Click **New** for first-time use.
- For the N5251A, select **Broadband 10MHz - 110GHz**.
- For banded, define a frequency band.
- Select **Standard PNA** to exit the Millimeter Module Configuration dialog.

**Name** Shows the currently selected configuration. Edit this field to change the configuration name. Type a unique name using only alphanumeric characters and underscore.

**New** Click to create a new Millimeter Module configuration. A name is automatically selected. Edit the Selected Configuration field to change the configuration name.

**Remove** Deletes a Millimeter Module Configuration.
**Test Set Properties**

**Test Set**  Select a test set to use in the current configuration. The firmware does NOT check to ensure that the selected test set is connected.

**Mixer Mode**  Check to allow mixer testing using SMC. Learn more.

**Route VNA RF to rear panel "SW SRC OUT"**  Available ONLY on PNA-X with option 224 or 423 AND when using N5261A and N5262A test sets.

When checked, Port 1 source is switched to J11 and Port 3 source is switched to J8 on the VNA rear panel. Use this configuration to quickly switch the RF Output back to the VNA front panel.

**Enable Test Set RF ALC**  Available for N5261A and N5262A ONLY. When checked, power is automatically leveled at the mmWave module RF input when using the standard cables and making non-pulsed measurements. Clear this box to use non-standard cables or when making pulse measurements. When cleared, the following fields become available:

- **Max Power Limit**  The maximum mmWave module RF input is limited to this value when Test Set RF ALC is OFF. When you exit this dialog box using OK, set the power out of the VNA using the Power and Attenuator dialog.
- **Offset**  Sets the loss of the cables. The mmWave module RF input is adjusted by this amount. Positive offset increases the power.
- **Slope**  Helps compensate for cable and test fixture power losses at increased frequency. The mmWave module RF input power increases as the sweep frequency increases in dB/GHz. The slope is defined relative to the mmWave module RF input frequency. The slope starts at 0Hz and a positive slope will increase the power level. Range is +/- 2 dB/GHz.

**Note:** Changing any of the source controls after a calibration will cause the source power cal to be turned off. A source power cal will need to be performed again with the new settings.

**Frequency Settings**

- **Multiplier RF IN**  RF Frequency Range (displayed in grey fields) multiplied by this value = test port frequency range.
- **Multiplier LO IN**  LO Frequency Range (displayed in grey field) multiplied by this value equals the test port frequency. The IF frequency is:
  - 'C' Models = 8.333 MHz
  - PNA-X models = 7.605 MHz
**Test Port Frequency**  Set the Start and Stop frequencies of the selected configuration at the test ports. This becomes the displayed Start and Stop frequency of the VNA.

**Important Notes**

- To set Test Port Frequency, first set the appropriate **Multiplier** values that are specified in your mmWave module documentation.

- Ensure that the RF and LO Frequencies (highlighted below) are within the frequency range of the sources. The VNA offers no warning if they are NOT.

<table>
<thead>
<tr>
<th>Multiplier RF IN:</th>
<th>Start Frequency</th>
<th>Stop Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>12.5000000000 GHz</td>
<td>18.3333333333 GHz</td>
</tr>
<tr>
<td>Multiplier LO IN:</td>
<td>9.3750000000 GHz</td>
<td>13.7500000000 GHz</td>
</tr>
</tbody>
</table>

**Source**

Click a button to launch the External Devices dialog where you can select an internal or external source to be used for the VNA LO source or VNA RF source.

- **Cancel** Closes dialog box without saving changes.

- **OK** Saves the configuration and the **VNA is Preset** before making the appropriate settings.

- **About MM...** Displays information about the current hardware configuration.

**Mixer Mode**

Mixer measurements can be made at mmWave frequencies using **SMC**. (VMC measurements are NOT supported.)

Beginning with A.09.40, mixer measurements can be made with a 2-port test set connected to a 4-port PNA-X. This configuration yields a 2-port mmWave system. [Learn about 2-port system connections and limitations.](#)

Before A.09.40, The Mixer Mode checkbox could be enabled ONLY when the number of VNA test ports matched the number of ports on the mmWave test set. This means that an N5261A (2 port test set) could ONLY be connected to a 2-port VNA and an N5262A (4-port test set) could ONLY be connected to a 4-port VNA.

**Procedure**
1. Connect your DUT to the mmWave system as described below.

2. Configure this dialog (Millimeter Module Configuration). Check **Mixer Mode**, then press **OK**. This presets the VNA.

3. **Create an SMC measurement.**

4. **Make mixer settings.** As with standard SMC measurements, only two DUT ports can be swept in frequency. The remaining DUT port must be a fixed frequency. **See configuration used for harmonic mixers.**

5. **Increase power** for mmWave modules that are connected directly to a VNA port or external source.

6. Calibrate using the **SMC Calibration Wizard with mmWave Power Control.**

**Hardware Connections for Mixer mode**

The following image shows the standard connections from a N5261A or N5262A test set port to a mmWave module.

mmWave mixers usually require that two of the three mixer ports operate at mmWave frequencies. When **Mixer Mode** is checked on this dialog (Millimeter Module Configuration), the following restrictions apply:

- **On a 2 port mmWave system**, only **port 1** of the test set can be used as a mmWave frequency port. Port 2 can **NOT** be used.

- **On a 4 port mmWave system**, only **ports 1 and 3** of the test set can be used as mmWave frequency ports. Ports 2 and 4 can **NOT** be used.

- **The SMC parameter** being measured must be within the frequency range of the VNA or within the frequency range...
range of the banded mmWave module. Frequencies in between these ranges are allowed by the SMC mixer setup dialog, but the measurement results on the screen are NOT accurate.

Connections with a 4-port mmWave system

**Upconverters**

- DUT Input - Connect to VNA port 2 or port 4.
- DUT LO - Connect mmWave module to test set Port 3.
- DUT Output - Connect mmWave module to test set Port 1.

**Downconverters**

- DUT Input - Connect mmWave module to test set Port 1.
- DUT LO - Connect mmWave module to test set Port 3.
- DUT Output - Connect to VNA port 2 or port 4.

Connections with a 2-port mmWave system

Although supported, testing mmWave mixers with a 2-port system can be challenging for the following reasons:

- Testing mmWave mixers requires that two of the three DUT ports be at mmWave frequencies.
- Only test set port 1 is capable of adequately driving a mmWave module when used as a receiver.
- Therefore, the second DUT port that requires mmWave frequencies must have the mmWave module connected directly to an external source or a VNA second source.
- When using the mmWave module as a source, only the DC Bias and RF cable is necessary. The LO cable to the mmWave module is NOT used. This is because the RF input frequencies are multiplied in the mmWave module to provide the source frequencies. So a mmWave module used as a source can use the RF cable to connect directly to the VNA second source or an external source. About +5 dBm of RF power is required to adequately drive the mmWave module.

**Downconverters** - requires two mmWave modules as sources

- DUT Input - Connect the mmWave module to the test set port 1.
- DUT LO - Connect the RF cable of the mmWave module to an external source or the VNA (SRC2) second source.
- DUT Output - Connect to VNA port 2.

**Upconverters** - requires a mmWave module as a **source** at the DUT LO and a mmWave module as a **receiver** at the DUT Output:

- DUT Input - Connect to VNA port 2.
- DUT LO - Connect the RF cable of the mmWave module to an external source or the VNA (SRC2) second source.
- DUT Output - Connect the mmWave module to the test set port 1.

**Measuring Harmonic Mixers**

Harmonic mixers have a multiplier circuit in the LO port of the DUT. Enter the multiplier value in the numerator of the X LO port in the **SMC mixer setup dialog**. This will provide the correct LO frequencies out of the appropriate source.

**Spectrum Analyzer mmWave Measurements**

Beginning with A.10.45.xx, spectrum analyzer measurements can be made at mmWave frequencies using the Option S93090xA **Spectrum Analyzer** application plus Option S93093A/B or Option S93094A/B. Broadband and banded mmWave measurements are supported. The test set configuration is required for both.

Options S93093A/B and S93094A/B are mmWave measurement options specific to the Spectrum Analyzer application. Broadband measurements require Option S93093A/B and measure from 10 MHz to = 110 GHz. Banded measurements require Option S93094A/B and measure frequencies > 110 GHz.

**Note:** The **IF Response Adjustment** must be performed each time a new configuration is set up or if cables are changed.

The following procedure outlines the steps required to set up a spectrum analyzer mmWave measurement.

1. Configure the mmWave measurement using the **Millimeter Module Configuration** dialog.
2. On the VNA front panel, press **Meas > S-Param > Meas Class...**
3. Select **Spectrum Analysis**, then either:
   - **OK** delete the existing measurement, or
• New Channel to create the measurement in a new channel.

4. A Spectrum Analysis measurement is displayed. Learn about setting up a Spectrum Analyzer measurement.

5. Select other mmWave supported measurement classes as needed.

**mmWave Module Power Level Control**

Beginning with A.09.40, the following TWO features are integrated into Guided Cal:

- For S-parameter Cal - Use Multiple Sensors
- For SMC Cal - Power Table

The following table shows features that can be used to provide leveled power to the input of your DUT for S-parameter and SMC measurements.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Use when...</th>
<th>Use for...</th>
<th>Access the feature...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receiver leveling</td>
<td>Provides a sweep-to-sweep leveled power.</td>
<td>Works anytime.</td>
<td>S-params and SMC</td>
<td>Before or after Cal</td>
</tr>
<tr>
<td>Use Multiple Sensors</td>
<td>Allows several power sensors to be used to calibrate source power.</td>
<td>You require more than one power sensor to complete the source power calibration of the measurement frequency range.</td>
<td>S-params and SA</td>
<td>During Guided Power Cal</td>
</tr>
<tr>
<td>Power Table</td>
<td>Build or use a file that contains data of mmWave module output power vs frequency.</td>
<td>A power sensor is NOT available for calibration of the mmWave modules being used.</td>
<td>S-params</td>
<td>During Std Source Power Cal</td>
</tr>
<tr>
<td>Calibrate the source at multiple power levels</td>
<td>Source power is measured using the specified power meter/sensor or VNA receiver to construct a</td>
<td>A component is used in the source path which does not have NOT linear gain or loss over frequency.</td>
<td>S-params</td>
<td>During Std Source Power Cal</td>
</tr>
</tbody>
</table>

6801
If you have one or more power sensors that spans the frequency range of your measurement, then use the following process.

Otherwise, perform a standard Source Power Cal. Learn how.

**Using one or more power sensors**

Check ALC Enabled (if available) on the Millimeter Module Configuration dialog.

1. With an S-parameter measurement active, press Cal > Smart Cal....

2. On the following Select Ports dialog, check Calibrate source and receiver power, then click Next.

3. **Important:** In the following dialog, check Use Multiple Sensors, even if using only one sensor.

4. Complete the Guided Cal process.

**Note:** During the 'Connect a power sensor to port n'...step, the following error message may be displayed:

The default power level of 11 dBm is unachievable after calibration. Lower the power before starting calibration.
This means that a high amount of loss was measured in the path, and 11 dBm at the test ports will not be possible.

**Cancel** the calibration and lower the source power level using the **Power and Attenuators dialog**.

**Perform a standard Source Power Cal - S-parameter measurements**

When one or more power sensors that span the frequency range of your S-parameter measurement are NOT available, then use the following process.

**Note:** Perform an S-parameter calibration AFTER performing the following Source Power Cal.

1. Check **ALC Enabled** (if available) on the **Millimeter Module Configuration** dialog.

2. Press **Cal > Main > Other Cals > Source Power Cal...** then **Options...** to launch the following dialog:

   ![Source Power Calibration Options](image)

   **See the help topic for this dialog**

3. If one does not already exist, create a power table to be used to calibrate the VNA receiver. [Learn how](#).

4. Check **Use a power table and the analyzer reference receiver**.

5. Click **Power Table**, then navigate to the *.prn file.

6. Click **OK**.

7. Check **Calibrate the source at multiple power levels**.

8. Click **Power Levels...**, then enter the Max power, Min power, and Step Size at which source power should be
corrected. Be sure that the source power for your measurement is within these power levels. Otherwise, source power will NOT be accurate. Learn more about this feature.

9. Check Calibrate the analyzer reference receiver, then click OK.

10. On the Source Power Cal dialog, click Take a sweep. The output of the test set is set to Max power and a sweep is performed to calibrate the reference receiver.

11. Power is dropped for several subsequent sweeps. The calibrated reference receiver is used to fully characterize the source power.

12. The entire correction table can be saved along with the instrument state in a *.csa file. Learn how.

13. Power out of the input module should be flat and accurate.

SMC Cal

Use the following calibration process to achieve accurate, leveled power at the mmWave test ports.

1. With a configured SMC measurement active, Cal > Main > Smart Cal....

2. At the SMC Calibration Setup dialog, when a Thru standard is NOT available, check Independent power cals for input and output ports (no thru).

3. On the Select Ports dialog, check Calibrate source and receiver power, then click Next.

4. At the following Power Cal settings dialog:

   ![Power Cal Settings dialog](image)

   Learn about this dialog

   a. When you have ONE power sensor that spans the frequency range of your SMC measurement, then click Power Meter Settings to configure the power sensor. There are currently NO provisions for using multiple power sensors with SMC Calibration.

   b. Otherwise, use the following Power Table process.

      i. If one does not already exist, create a power table to be used to calibrate the VNA receiver.
Learn how.

ii. Check **Use Power Table**.

iii. Click **Power Table**, then navigate to the *.prn file. The selected *.prn file is annotated to the dialog.

iv. Click **OK**.

5. If you checked **Independent power cals for input and output ports (no thru)**, you will ALSO be prompted to select a power table for Port 2.

6. Complete the **Guided SMC Cal** process.

---

**Power Table**

**Note:** This is NOT the same table that is used for the **Calibrate the source at multiple power levels** feature.

A power table is a text file with data that describes the output power of the module as a function of frequency. This is valid when the mmWave module is driven at high levels (+11 dBm). This file may have been created for you by a third party or shipped with your mmWave Module. If not, you can create this *.prn file from the manufacturer's specification for the mmWave module.

This file can be created manually, using a text file program such as Notepad. Copy the header information, and create the file with two columns, one for frequency and one for output power.

---

![Port 3 Power.prn - No...](image)

<table>
<thead>
<tr>
<th>Frequency (Hz)</th>
<th>Output Power (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>75000000000</td>
<td>6.709234e+000</td>
</tr>
<tr>
<td>75175000000</td>
<td>6.635661e+000</td>
</tr>
<tr>
<td>75350000000</td>
<td>6.669741e+000</td>
</tr>
<tr>
<td>75525000000</td>
<td>6.737478e+000</td>
</tr>
<tr>
<td>75700000000</td>
<td>6.729851e+000</td>
</tr>
<tr>
<td>75875000000</td>
<td>6.688992e+000</td>
</tr>
<tr>
<td>76050000000</td>
<td>6.629982e+000</td>
</tr>
<tr>
<td>76225000000</td>
<td>6.676495e+000</td>
</tr>
<tr>
<td>76400000000</td>
<td>6.773778e+000</td>
</tr>
<tr>
<td>76575000000</td>
<td>6.811515e+000</td>
</tr>
<tr>
<td>76750000000</td>
<td>6.798744e+000</td>
</tr>
<tr>
<td>76925000000</td>
<td>6.740466e+000</td>
</tr>
</tbody>
</table>
**Example .prn file**

*Note:* With Rev. 09.31, the first line of the *.prn file must have the Input power at which these measurements were made. Otherwise, an error message appears with the default value that will be assumed. See above image for format.

---

**Using Cal All Wizard for Supported mmWave Measurement Classes**

Beginning with A.10.45.xx, the Cal All Wizard can be used to calibrate broadband and banded configurations.

A power table can be used instead of multiple power sensors to cover a wide frequency range. In addition, a power table can be used when the measurement frequency exceeds the frequency range of the power sensor (typically > 110 GHz). Refer to Power Table for information on how to create a power table.

The power table file name must be **powertable1.prn**, where "1" corresponds to the port number. This file must be stored in the following directory on the VNA:

C:\ProgramData\Keysight\Network Analyzer\Configurations\<configuration name>\%

where **configuration name** is a directory name that is also the name of the currently selected configuration in the Millimeter Module Configuration dialog.

Once the power table has been created and saved to the directory shown above, it will be listed in the Sensor drop down menu in the Power Cal Settings dialog where it may be selected instead of a power sensor.

1. Check ALC Enabled (if available) on the Millimeter Module Configuration dialog.

2. Ensure that the measurement classes to calibrate are active.

3. Press **Cal > Main > Other Cals > Cal All...** to launch the following dialog showing the active measurement classes:
4. Select the ports, click **Next**, then confirm or change the calibration properties in the **Measurement Class Cal Properties** dialog.

5. Click **Next** to access the **Calibration Attenuator Settings** dialog.

6. In the **Calibration Attenuator Settings** dialog, perform the following:
   
   a. Set the attenuator settings. Learn more.
   
   b. Click on the **Noise Reduction** button to improve measurement accuracy. Learn more.
   
   c. Click on the **Mechanical Devices** button to view all switches and attenuators in the VNA. Learn more.

7. Click **Next**, then select the DUT connectors and calibration kits in the **Select DUT Connectors and Cal Kits** dialog.

8. Click **Next** to access the **Power Cal Settings** dialog.

9. In the **Power Cal Settings** dialog, perform the following:
   
   a. Check **Use Multiple Sensors** if more than one power sensor is needed to cover the frequency range then select a sensor from the **Sensor** drop down menu.
   
   b. Otherwise, check **Use Multiple Sensors** then select the power table from the **Sensor** drop down menu. Learn how to create a power table.
c. Learn more about **Accuracy Tolerance** and **Max Number of Readings**.

10. Click **Next** and follow the calibration process until completed.
mmWave Measurements without a Test Set

This topic describes the basic configuration needed to connect a waveguide transmission reflection frequency extender directly to a PNA-X (without a N526x test set). This configuration will allow you to create an integrated 2-port banded waveguide S-Parameter measurement solution. In particular, it will focus on the options and configurations of the PNA-X as well as the settings required to make a complete measurement.

- Requirements and Limitations
- Hardware Connections
- VNA Settings

See Also

mmWave Configuration with a N526x Test Set

Requirements

- 4-port PNA-X, or 2-port / 2-source PNA-X
- Option S93080A FOM. Learn more about FOM.
- Firmware A.09.00 or greater
- mmWave Modules
- DC Power Supply for mmWave Modules
- mmWave Cal Kit

- The frequency extenders being used should have the following basic architecture. This is an example of the OML modules, although other frequency extenders maybe used.
The key parameters to be aware of are:

- Ensure that the PNA-X being used has sufficient drive power for the RF and LO signals. Since most frequency extenders typically operate in a saturated mode the PNA-X has sufficient RF port power to drive the OML, VDI, and Farran frequency extenders.

- The IF output of the frequency extenders to the receiver input is less than the compression level of the PNA-X.

Limitations

- This configuration is limited to S-Parameters only.

- Log Sweep is not allowed.

Hardware Connections
The following list of components applies to the example shown above:

- Dual Source PNA-X N5242 with option 400 include option S93080A FOM
- Quantity 2 of 75-110 GHz Transmission Reflection Modules N5256A.
- DC Power Supply 12V at least 3 Amp to drive 2 Frequency Extenders E3632A.
- Special Module cables N5260AK48 quantity 2, these include all the RF, LO, IF and special DC cable for the OML module used.

Connect the frequency extenders to the VNA as shown below:

- This shows how one mmWave module is configured. When used with a 2-port PNA-X, the internal second source is used instead of port 3.
- This same principal is used to configure the second mmWave module using ports 2 and 4.
Remove the jumper cables, then connect to the Receiver IN connectors.

<table>
<thead>
<tr>
<th>Color</th>
<th>VNA to 1st Module (VNA to 2nd Module)</th>
<th>mmWave Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orange</td>
<td>Rcvr Ref 1 IN (Rcvr Ref 2 IN)</td>
<td>Ref IF Out</td>
</tr>
<tr>
<td>Blue</td>
<td>Rcvr A IN (Rcvr B IN)</td>
<td>Test IF Out</td>
</tr>
<tr>
<td>Red</td>
<td>Port 1 (Port 2)</td>
<td>RF In</td>
</tr>
<tr>
<td>Green</td>
<td>Port 3 (Port 4)</td>
<td>LO In</td>
</tr>
</tbody>
</table>
VNA Settings

**Note:** A macro is available that makes the following VNA settings automatically. [Learn more.]

1. In **Power and Attenuators** dialog:

   ![Power and Attenuators](image)

   a. Set the Port 3 and Port 4 power to ON. (This allows LO power to always be ON).
   
   b. Adjust the port power to adequately drive the mmWave module (12 dBm is this example).
   
   c. Clear the **Port Powers Coupled** checkbox if the LO and RF power requirements are different.

2. In **Path Configuration** dialog, Set Port 1 Ref Switch to **External**. This is NOT necessary for rear-panel connections and on a second mmWave module connected to ports 2 and 4.

   ![Path Configuration](image)

3. In **FOM** dialog:

   **Note:** The following procedure shows mmWave frequencies from 75 GHz to 110 GHz.
To make the following Multiplier, Divisor, and Frequency settings, click in the Settings field for the corresponding Source or Receiver, then click Edit.

b. Receivers: Set Offset to 100 MHz and Multiplier to 0.
c. Source 2 (LO): Set Offset to 12.5 MHz and Divisor to 8.
d. Set all modes to Coupled.
e. Check Frequency Offset: ON
f. Set Primary: Start 75 GHz - Stop 110 GHz
g. Set X-Axis Display Annotation to Primary.

4. In Calibration Preferences dialog: To calibrate at the mmWave frequencies check the Use FOM Primary Frequency.

Using the Macro

Using the macro will eliminate all of the steps involved in setting up the power and the attenuators as well as the Frequency settings. Download the macro from the following URL http://na.support.keysight.com/pna/apps/mmwave_setup.msi.

Load the macro and enter the values for setting up the measurement.
The N5264B is a PNA-X with no Source Ports or Tunable receivers. This makes the N5264B a very fast and very sensitive IF receiver that has been designed specifically for antenna and radar cross-section (RCS) measurements. When used with external sources, the N5264B can make frequency-scan measurements of antennas, or make RCS measurements in time domain.

- **N5264B Options and Limitations**

- **Configuring the N5264B**
  - IF Receivers
    - Internal LO Source (Opt 108)
  - External Sources
  - How to Create Measurements
  - Calibrating the N5264B

- **See Also**
  - FIFO and other Antenna Features
  - Antenna Selection Guide
  - 85309 Manual updated for VNA -P/N 85310-90002
  - Antenna and RCS Measurement Configurations
  - N5264B Specs
N5264B Options and Limitations

The N5264B has the following options:

- **IF Receiver** (base model) - The A, B, C, D, and R receivers are always set at the specified IF frequency.

- **IF Receiver + LO source** (Opt 108) - In addition to the receiver-only option, this option adds a 10 MHz to 26.5 GHz LO source.

- **Fast-Sweep / FIFO Mode** (Opt S93118A/B) - These features together allow you to make very fast measurements and save the data to a remote computer. Learn more.

Limitations

The following VNA features are NOT available on the N5264B

- No Source ports

- No S-Parameters - Arbitrary ratioed parameters ARE allowed.

- No Application support

- Limited Calibrations
Configuring the N5264B

IF Receivers

- The five IF receiver inputs are on the N5264B rear panel.
- All five IF receivers are measured at the same time. However, only those measurements that are displayed are updated.
- External mixers are always required to down-convert signals to the IF frequency.
- The default IF frequency for the N5264B is 7.605634 MHz.
- Change the IF frequency to any value between +14.9999 MHz and -14.9999 MHz using SENS:IF:FREQ (SCPI) or IFFrequency (COM) commands.

Internal LO Source (Option 108) - 10 MHz to 26.5 GHz

The LO source output connector is on the N5264B rear panel.

The power level of the internal LO source is typically about +10 dBm and can NOT be changed.

To change the frequency of the LO Source:

1. Change the IF frequency of the measurement if necessary.
2. On the FOM dialog, change Receivers frequencies to the RF frequency range to be measured.
3. The LO Source frequency range is set automatically. This frequency value can not be viewed.

External Sources

Because the N5264B has no internal sources (except for the optional LO), newly created measurements are displayed with a source port = 0 (zero).

External sources can be configured so that they are controlled by the N5264B. Learn how to configure an external source.

Once configured, all existing and new measurements are changed to use the external source as the source port. When two or more sources are configured, the first configured source is displayed by default.

The external source settings can be changed from the following dialogs:

- Power and Attenuators dialog - Controls the ON | OFF state (set to Auto by default) and Power level of the external source.
• **FOM dialog** - Controls the frequency range. External sources are listed by name and uncoupled by default.

**How to Create Measurements**

Create ratioed and unratioed measurements using the standard **Receivers tab**.

S-parameters are not available.

**Calibration**

The only calibrations available are Response, Source Power and Receiver calibrations.

These are performed from the **Unguided Cal Wizard**.
Configure an External Device

Once configured (as shown in this topic), an external device will appear in, and be controlled from, relevant VNA dialogs as though it were internal to the VNA.

- **External Device** Configuration dialog

The following (separate) topics discuss how to set properties for these types of devices:

- External Source Properties
- Power Meter As Receiver (PMAR) Properties
- External Pulse Generator Properties
- Configure an SMU (Source/Measure Unit)
- Configure a DC Source/Meter
- Drivers for External Compound Sources

---

**How to access the External Device Configuration dialog**

VNA Applications have additional methods of launching this dialog.

<table>
<thead>
<tr>
<th>Using Hardkey / SoftTab / Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Setup</strong> &gt; <strong>External Hardware</strong> &gt; <strong>External Device...</strong></td>
<td>1. Click <strong>Instrument</strong></td>
</tr>
<tr>
<td>2. Select <strong>Setup</strong></td>
<td>2. Select <strong>Setup</strong></td>
</tr>
<tr>
<td>3. Select <strong>External Hardware</strong></td>
<td>3. Select <strong>External Hardware</strong></td>
</tr>
<tr>
<td>4. Select <strong>External Device...</strong></td>
<td>4. Select <strong>External Device...</strong></td>
</tr>
</tbody>
</table>

See Remotely Specifying a Source Port

---

**External Device Configuration** dialog box help
Important Notes

- This dialog is used to configure the following types of external devices:
  - External Source Properties requires FOM S9x080A, S9x082A or Option 009
  - Power Meter As Receiver (PMAR) Properties
  - External Pulse Generator Properties
  - DC Power Analyzer
  - SMU (Source/Measure Unit)

- To configure an external source using this dialog, your VNA must have FOM Option. Without this option, you must control an external source manually. See Synchronize an External Source for help with manual source control.

- By default, an external device is **de-activated** when the VNA is Preset or when a Instrument State is recalled. This behavior can be changed with a Preference setting so that it remains active through a Preset or Instrument State recall.

- External Device properties are NOT saved in an Instrument State file. However, the reference to the External Device from relevant VNA dialogs IS saved. Therefore, recalling a state file that refers to a device that is NOT present will result in a “Device configuration not found” error.

- Multiple configurations for the same physical device can be Active. However, only one configuration for the same external source can have the I/O Enabled.

- The driver for the P9336A USB I/Q Arbitrary Waveform Generator must be installed in the PNA. The driver can be downloaded from the Keysight website.

- The B2902A or N6705C driver names cannot be the same as the device name because the driver
names are reserved. For example, a device name of "B2902SMU" when using the B2902A driver is acceptable.

**External Devices**

The devices that are currently configured appear in this list. The number of devices that can be configured is limited by the specified Interface.

**New** Click to create a new device configuration. The default name is Device\(<n>\), where \(<n>\) is the next number for 'Device'.

**Remove** Click to remove the selected device from the list.

**Properties**

**Name** Enter a device name as it will appear when referring to this device in VNA dialog boxes. Edit the name at any time. Duplicate names are not allowed.

- Because External Devices can be used with FOM ranges, do NOT name an external device any of the following FOM range names: "primary", "receivers", or "source", "source1", "source2", "source3" and so forth. Learn more about FOM ranges.
- Do NOT use a parameter name such as "S11" or "R1".
- **DC Analyzer devices MUST use at least three characters in the name.**

**Device Type** Select one of the following:

(Quotes are used when specifying in a remote program.)

- "DC Meter" - Learn more
- "DC Source" - Learn more
- "Power Meter" (PMAR) - Learn more
- "Pulse Generator" - Only the Keysight 81110A Pulse Generator is supported. Learn more
- "SMU" - Learn more
- "Source" (RF) - Learn more
- "None" - returned remotely before setting Device Type.

**Driver** Select the appropriate model to be configured.
(Quotes are used when specifying in a remote program.)

For **Source** Device Type choose from:

("AG" is short for Agilent.-

- "AG81150_1" (Legacy driver with no modulation capabilities)
- "AG81150_2" (Legacy driver with no modulation capabilities)
- "AG81150_2S" (Legacy driver with no modulation capabilities)
- "AG836XX" (8360 and 8340)
- "AGESG" (ESG)
- "AGEXG" (EXG) See configuration note
- "AGGeneric" For sources that are NOT listed but can be controlled using SCPI. Click Device Properties, then Edit Commands to send commands to these sources. Learn how.
- "Agile Vector Adapter" (Driver for N5192 and N5194 Agile Vector Adapters with modulation capabilities. These sources need to be inserted in a back loop of the PNA. The PNA provides the LO.)
- "AGMXG" (MXG) The MXG must have at least firmware A.01.44 for FOM power sweep to work correctly.
- "AGPSG" (PSG)
- "HP834XX" (Driver for HP 834XX)
- "KtUXG" (Legacy driver for N5193)
- "M8190" (M8190A Arbitrary Waveform Generator with E8267D PSG Vector Signal Generator) Learn more .
- "M8190 + IQ mixer" (M8190A Arbitrary Waveform Generator with I/Q Mixer) Learn more .
- "M9336" (M9336A PXIe I/Q Arbitrary waveform Generator)
- "MXG_Vector" MXG driver that supports modulation for vector MXGs.
- "PSG_Vector" (PSG driver that allows modulation)
- "VXG" (M9383B/84B VXG Microwave Signal Generator)
- "VXT_Vector" (VXT driver with modulation capabilities)
- "M9383A" (PXIe Microwave Signal Generator)
- "P9336a + PSG " (P9336A USB I/Q Arbitrary Waveform Generator with E8267D PSG Vector Signal Generator)

For **SMU** Device Type, choose from:

- "B29xx" - B2900 Series
"N67xx" - N6700 Series

All other device types have only one driver.

Active - Show in UI  Check to make the device available for use in the relevant dialogs. An instrument state that is saved with an Active device (checked) will include the device in the state file. Otherwise, if the Active box is cleared, the device will NOT appear in the state file. Learn more about Instrument State files.

Note: The preset makes this disable. Check this again if necessary after preset.

Enable I/O  Clear this box to disable communication with the selected device. Do this to configure a device that is not yet connected to the VNA.

- Communication with devices is attempted when Enable I/O is checked, Active is checked, and OK is pressed.
- When communication is attempted, devices with Enable I/O checked are queried for selected limits, such as frequency, power, and number of points. If there are limit problems, the VNA sends an error and the affected channels are put into Hold. These limits are enforced by the dialog box in which they are set. Resolve the reported limit problem and then restore the triggering.
- If communication with a device is lost the affected channels are put into Hold.

Device Properties  Click to launch the Properties dialog for relevant Device type:

- Configure External Sources
- Configure a Power Meter As Receiver
- Configure an External Pulse Generator
- Configure a DC Meter / Source
- Configure a SMU

IO Configuration

Type VISA address that is used to connect the device to the VNA.

The "..." button opens the Select VISA Resource dialog box.
**Interactive I/O.** Open the interactive I/O of Connection Expert.

Select in Resource Finder from the following selections, then click **Find**.

- **GPIB** - Devices connected to the System Controller GPIB port.
- **USB** - Devices connected to the VNA USB ports. See Important First-time USB connection note.
- **Aliases** - Devices that are connected to ANY interface for which you created an alias. See Configure Alias and LAN devices.
- **LAN** - Devices connected to a network using a LAN connection. The VNA must also be connected to the network.

**Note:** Devices connected to LAN must first be configured in Keysight IO libraries before they will appear on the Available list. See Configure Alias and LAN devices.

**Configure Alias and LAN Devices**

Use this procedure to configure a device using a LAN interface. Also use for ANY device for which you want to set an alias (easily-recognized) name. The alias name appears in the Available field when Aliases is selected as the Interface.

1. On the VNA, minimize the VNA application (**System > Main > Minimize Application**).
2. In the system tray (lower-right corner) right-click the IO icon, then click **Keysight Connection Expert**.

To Add a LAN Device:
1. In Keysight Connection Expert, click Manual Configuration tab.

2. Select LAN Instrument.

3. Click, then enter the IP address of the external source.

4. Click Test This VISA Address to verify communication.

5. Click Accept.

To create an Alias for a connected device:

1. In the list of connected instruments, click the instrument, then click Add or Change Aliases.

2. Enter the Alias Name to be used in the External Device Configuration dialog.

Drivers for External Compound Sources

Drivers have been created to facilitate the configuration of external source devices commonly used for specific types of measurements such as Spectrum Analyzer (SA) or Modulation Distortion (MOD or MODX). The following is a list of the current drivers for these sources:

- **M8190** (M8190A Arbitrary Waveform Generator with E8267D PSG Vector Signal Generator)
- **M8190 + IQ mixer** (M8190A Arbitrary Waveform Generator with I/Q Mixer)

**M8190A Arbitrary Waveform Generator**

The M8190A configuration can be with or without an embedded CPU. However, the preferred M8190A is the configuration without an embedded CPU using a PCIe cable connection to the PNA. A special cable is required: PCIe 4-lane to PCIe 8-lane. In this configuration, the driver must be installed in the PNA for the driver to be seen by the CPU in the PNA. This makes the cable connection to the M8190A an extension of the memory domain of the PNA CPU. This configuration is several times faster than using the USB interface.

The M8190A must have at least one of the following options:

- **M8190A-14B** - 8 GSa/s with 14 bit resolution
- **M8190A-12G** - 12 GSa/s with 12 bit resolution

**Note:** Power up the M8190A first and wait until the LED on the front-panel is lit (green) before powering up the PNA.
M8190A Preferred Model (without CPU)

M8190A Outputs

The Direct Out outputs are used by the M8190 driver and the Amp Out outputs are used by the M8190 + IQ mixer driver.

M8190 Compound Source Driver
The **Driver** name is **M8190**.

**M8190 Device Properties**

In the dialog above, **M8190PSG** is the name assigned to the **M8190** driver. With **M8190PSG** highlighted in the **External Devices** window, de-select the **Active - Show in UI** check box, then click on the **Device Properties...** button to access the following dialog:

**Note:** Before changing the configuration of this driver, the **Active - Show in UI** must be de-selected. After changes are made and this dialog is closed, check the **Active - Show in UI** check box when ready to activate the driver. Otherwise, the setting changes will not be applied.
**Timeout (sec)** Sets the VISA timeout and will stop processing additional SCPI commands on the first error and will put the measurement into hold.

**Dwell per point (ms)** Applies a dwell in Hardware List triggering ONLY. Set the time (in milliseconds) the external source will wait before data acquisition.

**PSG I/O Configuration** - Sets the VISA address of the PSG.

**Amplitude I/Q** - Sets the amplitude of the I and Q differential outputs from the M8190A. Calibration affects these settings. The M8190A **Direct Out** outputs are connected to the PSG using four cables of the same length.

**DC Offset I/Q** - Sets the offset to remove any LO feedthru.

**8 GSa/s 14bits (14B option)** - Selects the Option M8190A-14B.

**12 GSa/s 12bits (12G option)** - Selects the Option M8190A-12G.

**Digital Upconverter (DUC)** - Selects the Option M8190A-DUC.

**M8190 + IQ Mixer Compound Source Driver**
The **Driver** name is **M8190 + IQ mixer**.

**M8190 + IQ mixer Device Properties**

In the dialog above, **M81_IQ** is the name assigned to the **M8190 + IQ mixer** driver. With **M81_IQ** highlighted in the **External Devices** window, de-select the **Active - Show in UI** check box, then click on the **Device Properties...** button to access the following dialog:

**Note:** Before changing the configuration of this driver, the **Active - Show in UI** must be de-selected. After changes are made and this dialog is closed, check the **Active - Show in UI** check box when ready to activate the driver. Otherwise, the setting changes will not be applied.
**Time out (sec)** Sets the VISA timeout and will stop processing additional SCPI commands on the first error and will put the measurement into hold.

**Dwell per point (ms)** Applies a dwell in Hardware List triggering ONLY. Set the time (in milliseconds) the external source will wait before data acquisition.

**RF output** - Selects the IQ mixer output. Select *Straight Out, no PNA port* to go out of the IQ mixer and not controlled by the PNA, or select one of the PNA ports. The PNA port selections will set the proper path and step attenuation in the PNA automatically. If *Straight Out, no PNA port* is selected, the IQ mixer output can still be connected to the PNA, but path and step attenuation must be set manually.

The following shows a configuration example where Port 1 (J11) is the source of the LO drive signal and the RF Output of the IQ mixer is re-injected into PNA Port 1 (J10) on the rear panel:
Source power offset - Sets the power offset applied at the source driver level. If a decent value of the RF path budget between the modulated source and the DUT port is set (negative value means attenuation, positive value means amplification: there is power booster added in the path), then (even without modulation power level calibration) the source requested power and the source obtained power at the DUT plane will be aligned.

IQ Mixer LO drive - Selects the LO signal source for the IQ mixer from any PNA port, an external source, or baseband (no LO drive signal). One of four possible M8190A baseband outputs (output of DAC) can be selected: DC or AC output of Channel 1, or DC or AC output of Channel 2.

Power - Sets the power of the LO signal.

Source I/O Configuration for external source LO drive - Sets the address of an external source when used as the LO signal source.

Amplitude I/Q - Sets the amplitude of the I and Q differential outputs from the M8190A. Calibration affects these settings. The M8190A Amp Out outputs are connected to the PSG using four cables of the same length.

DC Offset I/Q - Sets the offset to remove any LO feedthru.

8 GSa/s 14bits (14B option) - Selects the Option M8190A-14B.

12 GSa/s 12bits (12G option) - Selects the Option M8190A-12G.

Digital Upconverter (DUC) - Selects the Option M8190A-DUC.
External Source Configuration

Once configured, an external source appears in VNA dialogs as though it were an internal source. This capability requires FOM Option S93080A.

In this topic:

- How to Configure an External Source
- Important Notes
- Trigger Settings and Physical Connection diagrams
- Generic Source Commands dialog

### How to Configure an External Source

1. **Important:** Create an External Source device by name (one-time). [Learn how](Separate topic).
2. Then click **Device Properties** to **Configure the External Source**. (This topic)

VNA Applications have additional methods of launching this dialog.

#### Using Hardkey/SoftTab/Softkey

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Setup</strong> &gt; <strong>External Hardware</strong> &gt; <strong>External Device...</strong></td>
<td>1. Click <strong>Instrument</strong></td>
</tr>
<tr>
<td>2. Click <strong>Device Properties</strong> to <strong>Configure the External Source.</strong></td>
<td>2. Select <strong>Setup</strong></td>
</tr>
<tr>
<td>3. Select <strong>External Hardware</strong></td>
<td>3. Select <strong>External Hardware</strong></td>
</tr>
<tr>
<td>4. Select <strong>External Device...</strong></td>
<td>4. Select <strong>External Device...</strong></td>
</tr>
</tbody>
</table>

See [Remotely Specifying a Source Port](#)

**External Source Configuration** dialog box help
This dialog box is used to make external source settings.

**Important Notes about External Sources**

- First create an External Source (device) by name (one-time). [Learn how](#). (Separate topic)

- Once you create and activate an external source from the Configure an External Device dialog, it becomes available from the following VNA dialog boxes as well as the softkeys and entry toolbar, as if it were an internal VNA source. **Use the following dialogs to set the state, frequency, and power level of the external RF source:**
  - Power and Attenuators dialog
  - FOM dialog
  - New Trace / Receivers tab dialog

- By default, an external source is **de-activated** when the VNA is Preset or when a Instrument State is recalled. This behavior can be changed with a Preference setting.

- External Keysight sources are usually limited to 1601 points with List-sweep mode. To 'work around' this limitation, divide the measurement among multiple channels. For example, to attain a sweep of 3200 points, create two channels of 1600 points. You can also use manual source control which
supports Step-sweep mode. In this mode an external source can have up to 65,535 points. See Synchronize an External Source for help with manual source control.

- External sources should always share the same 10 MHz Reference signal as the VNA. Connect a BNC cable from the VNA 10 MHz Ref Output to the External Source Input.

- All newly-activated sources are preset, with source power OFF. Source power must be turned ON in the Power dialog. Frequency Offset must be enabled in the FOM dialog.

- The same source can NOT be used more than once in the same channel.

- The VNA automatically controls all trigger settings for the external source.

- See EXG Sources configuration note.

### Source Settings

**Timeout (sec)** Sets the VISA timeout and will stop processing additional SCPI commands on the first error and will put the measurement into hold.

**Dwell per point (ms)** Applies a dwell in Hardware List triggering ONLY. Set the time (in milliseconds) the external source will wait before data acquisition.

**Enable Modulation Control** - Enables modulation of an external modulation source to be controlled from a source dialog.

**Check Lock of Reference Clock IN** - (SA and Modulation Distortion source drivers) Checks for the VNA's reference clock at the external source. The reference clock is checked at external source driver startup. If the clock is not available, an error message is displayed. If a dual hardware source is used (for example, M8190+PSG), the reference clock inputs of both sources are tested. In the case of the USB chassis (P9336A), the reference clock is tested at the rear 10 MHz chassis input if the P9336A 10 MHz chassis reference is selected.

**Edit Commands** Provides a method to send SCPI commands to AGGeneric (not listed) sources.

### Trigger Settings and Physical Connection diagrams

**Note:** The VNA controls ALL external source trigger settings automatically (except for those on this dialog). All settings in the External Trigger dialog are ignored.

**Trigger Mode**

**Software CW (GPIB)** Slowest method.

- The external source receives each CW frequency from the VNA over GPIB, USB, or LAN. No other trigger cables are required. Although a Trigger Port selection may be available, it is
NOT used.

**Hardware List (BNC)** Fastest method.

- NOT available for AGGeneric (not listed) sources.
- The external source receives a list of CW frequencies from the VNA, then receives trigger signals though a rear-panel connector when appropriate from the VNA.
- If the number of data points used in the measurement exceeds the capability of the external source, the VNA automatically switches to Software CW (GPIB) trigger mode. This will slow the measurement significantly.

**Trigger Port** Used ONLY for Hardware List Trigger Mode. Select the VNA rear panel connector to be used for triggering. The sources must be connected as follows:

- For ONE or TWO external sources, connect each source to an AUX trigger pair. See rear panel Aux connectors.
  - Source Trig Out connects to VNA Aux Trig In
  - Source Trig In connects to VNA Aux Trig Out

The following diagram shows the connections between the PNA (AUX TRIG 1 or 2 can be used) and N519xA UXG signal generator.

- For more than TWO sources, some will need to be connected using the following daisy chain image. For example, with three external sources, two could be daisy-chained to Aux 1 while the third is connected by itself to Aux 2.
Notes

- Source 1, which receives the trigger out of the VNA, must be the first source listed on the External Devices Configuration dialog box. Devices are listed in the order in which they are created. You may have to delete, then re-create a source to move it down on the list.

- Connect **EXG sources** as follows:
  
  - EXG rear-panel label TRIG 1 connects to VNA AUX TRIG 1 OUT
  - EXG rear-panel label TRIG 2 connects to VNA AUX TRIG 1 IN

**Note:** The VNA AUX TRIG 2 can be used instead of VNA AUX TRIG 1.
Enter the SCPI commands that control the following functions on your AGGeneric (not listed) source. A field without a SCPI command entered will be ignored and that function will not be set.

To launch this dialog, click **Edit Commands** on the **External Source Properties** dialog.

**Operation Complete (**OPC)**.

**Preset**  Presets the source

**Set CW Frequency**  Sets CW Frequency

**Set CW Sweep Mode**  Sets source sweep mode

**Set Power**  Sets source power

**Set Power State**  Turns power ON or OFF
Configure DC Sources and DC Meters

Once configured, one or more DC Sources and DC Meters can be controlled by the VNA. DC Power Analyzers are also supported, but they must be configured as a separate Source and Meter.

The Keysight N6700 series and B2900 series DC Analyzers are supported with configuration files that can be loaded on the DC Meter and DC Source property page. Once loaded, the SCPI commands that control the DC device can be modified and saved. Learn how.

See Also

External DC Meter Data Conversion

How to Configure a DC Meter or DC Source

1. **Important**: Create a DC Source / Meter device by name (one-time). Learn how (separate topic).

2. On the Configure an External Device dialog, click **Device Properties**. (This topic).

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>Setup &gt; External Hardware &gt; External Device...</strong></td>
<td>1. Click <strong>Instrument</strong></td>
</tr>
<tr>
<td>2. Click <strong>Device Properties</strong> to Configure the External Source.</td>
<td>2. Select <strong>Setup</strong></td>
</tr>
<tr>
<td></td>
<td>3. Select <strong>External Hardware</strong></td>
</tr>
<tr>
<td></td>
<td>4. Select <strong>External Device...</strong></td>
</tr>
</tbody>
</table>

Once configured, set the DC source voltage and display DC meter measurements:

- **DC Sources**: DC Source control is available in Standard, Gain Compression/GCX, and FCA channels.
  
  - Set the Start and Stop voltage on the **DC Control dialog**. To access this dialog: Press **Sweep > Source Control > DC Source**.

- **DC Meters**: DC meter measurements are available in Standard, Gain Compression /GCX, Swept IMD / IMDX, and FCA channels.
  
  - In **Gain Compression /GCX, Swept IMD / IMDX, and FCA channels**, display DC parameters as you would an RF parameter, by clicking **Trace > New Trace**.
In a Standard channel, configure an unratioed measurement. See AUX tab.

**DC Source / Meter Configuration** dialog box help

The DC Source and DC Meter properties are almost identical in how they operate. Both are documented here.

### Device Settings

**Timeout** - Sets a time limit for the DC source or meter to make contact with the VNA. If this time limit is exceeded, the VNA stops the measurement and displays the following error message.

**EXECUTION ERROR; OPC QUERY TIMEOUT ERROR**

If this occurs, check the connections between your VNA and external device.

**Dwell Before Sweep**  
Wait time before making a sweep.

**Dwell After Point**

- **DC source**  
  Wait time after setting the voltage/current at each data point.

- **DC meter**  
  Wait time before measuring voltage/current at each data point.

**Type:** This setting changes the units that are displayed in the DC Source dialog, the X-axis display annotation, and the underlying data format. Use these settings with Receiver or Source Correction (Scaling and Offset) to display and scale measurements with these units. Choose from:
V (volts - default)        dBm
A (amperes)               W (watts)
F (degrees)               C (degrees)
K (kelvin)

**Note:** To change the X-axis to display the DC Meter units, click **Response**, then **Display**, then **Labels**, then **Select X-Axis**, then select the DC Meter.

**Receiver / Source Correction**

- For a **DC source**, use the correction settings to scale and offset the output voltage.
- For a **DC Meter** (receiver), use the correction settings, along with Type, to display and scale measurements with appropriate units. For example:

  Measure the voltage across a 5 ohm resistor, then display the results in A(mperes).

  Using ohms law,  \( I = \frac{V}{5 \text{ ohms}} \) or \( I = V * 0.2 \)

  For receiver correction, enter **Scaling** = 0.2; **Offset** = 0.

**ON** Check to apply the following correction factors to each measurement.

**Offset**: Enter the value to offset the DC Meter reading or set the DC Source voltage.

**Scaling**: Enter the value to scale the DC Meter reading or set the DC Source voltage.

\[
\text{Displayed Output} = \left( \frac{\text{Measured}}{\text{Set value}} - \text{Offset} \right) \times \text{Scaling value}.
\]

**Edit Commands** - Click to start the Edit Commands dialog.

**Important Note:**

The Edit Commands dialogs (see below) MUST be completed. They are used to set the SCPI commands with which the VNA communicates with the DC device.

These commands are saved, along with other configuration settings, to configuration (*.xml) files. These files can then be loaded later when communicating with the same DC Device.

Configuration files for the Keysight N67xx and B29xx Power Analyzers and the other devices are pre-loaded on the VNA. Click **Load**, then navigate to: `c:\users\public\network analyzer\documents\drivers`. 
Save - Press to save the current DC Source or DC Meter configuration to an *.xml file. The list of files is NOT filtered by "DCMeter" or "DCSource", so use a descriptive filename.

Load - Press to load an existing configuration.

**DC Meter Edit Commands dialog box help**

**Global Tab**

The Global tab includes the system settings for the DC Meter.

**ID Query** - Enter the SCPI command to return the ID string of the DC Meter. Typically *IDN?

**Error Query** - Enter the SCPI command that is used to return DC Meter errors. Typically SYST:ERR?

**Enable I/O** - Enter the SCPI commands that is used to enable the DC Meter to read voltages.

**Disable I/O** - Enter the SCPI commands that is used to disable the DC Meter from reading voltages.
Test Connection

Click to start the Test Connection dialog. You must first have entered the I/O Configuration settings and select Enable IO on the External Device dialog.

Enter a SCPI command, then click **Send** or **Send&Read** when a return value is expected.

Sweep Tab

Use the Sweep Tab to send SCPI commands to the DC Meter at the beginning or end of each sweep.

**Abort Sweep** - Enter the SCPI command that is used to Abort or reset the DC Meter. This would be necessary when the VNA sweep is aborted or terminated. The VNA will then send the command to the DC Meter.

Point Tab
**Read (commands)** - Enter the SCPI command that is used to make a DC measurement at each data point.

**Set (commands)** - Use `<%variable>` to make a DC Meter setting.

### DC Source Edit Commands dialog box help

**Global Tab**

The Global tab includes the system settings for the DC Source.

**ID Query** - Enter the SCPI command to return the ID string of the DC Source. Typically `*IDN`? This entry can be left blank.

**Error Query** - Enter the SCPI command that is used to return DC Source errors. Typically `SYST:ERR`?

**Enable I/O** - Enter the SCPI commands that is used to enable the DC Source to output voltages.

**Disable I/O** - Enter the SCPI commands that is used to disable the DC Source from outputting voltages.

### Maximum / Minimum DC Output

**Read Max / Min Using** - Select, then enter the commands used to return the output limits of the DC source.

**Define Max / Min As** - If the DC Source has no commands to return these values, or you would
rather define the limit for your DC Source, select then enter the Max and Min voltage limits.

### Test Connection

Click to start the Test Connection dialog. You must first have entered the I/O Configuration settings and select Enable IO on the External Device dialog.

![Test Connection dialog](image)

Enter a SCPI command, then click **Send** or **Send&Read** when a return value is expected.

### Sweep Tab

Use the Sweep Tab to send SCPI commands to the DC Source at the beginning or end of each sweep.

Typically, you might send the output ON at the beginning of each sweep, and output OFF at the end of each sweep as shown above.

**Abort Sweep** - Enter the SCPI command that is used to Abort or reset the DC Source. This would be necessary when the VNA sweep is aborted or terminated. The VNA will then send the command to the DC Source.

### Point Tab
Note: The DC Source output voltages are configured on the DC Source dialog.

This dialog is used to configure the commands that are used to communicate with the DC Source.

Set commands - Enter the SCPI command, enclosed in {curly brackets} to output (set) a voltage/current from the DC Source for each data point.

- {\%f} - The value is a double value. (Most common).
- {\%d} - The value is a integer. This would be used when the voltage controls a remote switch. For example, you can program the value to: "0,1,0,1,0,1,...". where "0" = OFF and "1" = ON.
Configure a Power Meter As Receiver (PMAR)

When a power meter is configured as a VNA receiver (in standard measurement channels ONLY), you can...

- Extend the number of measurement receivers.
- Use the power meter as a scalar detector.
- Monitor the power at any point in a measurement system.
- Use multiple power meters in a Guided Power Cal to cover a wide frequency range.

**Note:** Multiple configurations for the same physical device can be Active. However, only one configuration for the same external source can have the I/O Enabled

- Use the power meter to level the stimulus power at any point in a measurement system.
- Use the power sensor as a PMAR device to confirm the accuracy of a Source Power Cal. Learn how.

**Note:** PMAR is not compatible with Point Sweep mode.

Once configured, a power meter can be used like any other VNA receiver in the following dialogs:

- New Trace / Meas dialog - used in Ratioed and Unratioed measurements.
- Receiver Leveling
- Frequency Offset Mode - Extend frequencies beyond VNA

See Also

- Supported Power Meters
• Important first-time USB connection note.

How to Create and Configure a PMAR Device

1. Create a PMAR device by name (one-time).
2. Then click Device Properties to configure the Power Meter/Sensor.

VNA Applications have additional methods of launching this dialog.

<table>
<thead>
<tr>
<th>Using Hardkey/SoftTab/Softkey</th>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press Setup &gt; External Hardware &gt; External Device...</td>
<td>1. Click Instrument</td>
</tr>
<tr>
<td>2. Click Device Properties to configure the Power Meter/Sensor.</td>
<td>2. Select Setup</td>
</tr>
<tr>
<td></td>
<td>3. Select External Hardware</td>
</tr>
<tr>
<td></td>
<td>4. Select External Device...</td>
</tr>
</tbody>
</table>

External Device Configuration dialog box help

This dialog allows you to create and configure a power meter to be used as a receiver by the VNA. Once you create and configure a power meter from this dialog box, it becomes available from VNA dialog boxes as well as the softkeys and entry toolbar, as if it were an internal VNA receiver.

• This dialog is ALSO used to configure an External Source. Learn more.
To configure a single power meter for a Source Power Cal, use the **Power Meter Settings** dialog.

**Important Notes**

- By default, an external PMAR device is **de-activated** when the VNA is Preset or when an Instrument State is recalled. This behavior can be changed with a Preference setting so that it remains active through a Preset or Instrument State recall.

- PMAR configuration is NOT saved in an Instrument State file. Therefore, recalling a state file that refers to a device that has been removed, or recalling a state file on a different VNA will result in a “Device configuration not found” error.

**External Devices**

The devices that are currently configured appear in this list. The number of devices that can be configured is limited by the specified Interface.

**New** Click to create a new PMAR configuration. The default name is Device<n>, where <n> is the next number for 'Device'.

**Remove** Click to remove the selected device from the list.

**Properties**

**Name** Enter a device name as it will appear when referring to this device in all VNA dialog boxes. Edit the name at any time. Duplicate names are not allowed.

**Notes**

- Because External Devices can be used with FOM ranges, do NOT name an external device any of the following FOM range names: “primary”, “receivers”, or “source”, “source1”, source2”, "source3” and so forth. Learn more about FOM ranges.

- Do NOT use a parameter name, such as "S11, or "R1".

**Device Type** Select **Power Meter**.

**Driver** Use AGPM for all Keysight Power Meters. See Supported Power Meters

**Active** Check to make the device available for use in the FOM, New Trace, and Receiver Leveling dialogs. An instrument state that is saved with an Active device (checked) will include the device in the state file. Otherwise, if the Active box is cleared, the device will NOT appear in the state file.

**Note:** Multiple PMAR configurations for the same physical device can be Active and Enabled.
Device Properties  Click to launch the Configure Power Sensor dialog.

IO Configuration

Interface  Select the interface that is used to connect the device to the VNA. These devices will then appear in the 'Available' field. Choose from:

- **GPIB** - Devices connected to the System Controller GPIB port.
- **USB** - Devices connected to the VNA USB ports. See Important First-time USB connection note.
- **Aliases** - Devices that are connected to ANY interface for which you created an alias. See Configure Alias and LAN devices.
- **LAN** - Devices connected to a network using a LAN connection. The VNA must also be connected to the network. Note: Devices connected to LAN must first be configured in Keysight IO libraries before they will appear on the Available list. See Configure Alias and LAN devices.

Available  Shows a list of devices that are connected to the specified IO Interface.

Refresh  Click to rescan the specified interface for devices.

Selected  Enter the IO configuration or select from the available list of IO Interfaces found.

Enable I/O  Clear this box to disable communication with the selected device. You would do this to configure a device that is not yet connected to the VNA.

- Communication with devices is attempted when Enable I/O is checked, Active is checked, and OK is pressed.
- If communication with a device is lost, the affected channels are put into Hold.
- When communication is attempted, devices with Enable I/O checked are queried for limits for frequency, power, and number of points. If there are limit problems, the VNA sends an error and the affected channels are put into Hold. These limits are enforced by the dialog box in which they are set. Resolve the reported limit problem and then restore the triggering.
- Communication is also attempted when clicking the Settings button on the Configure Power Sensor dialog. You can not change any of the sensor settings unless Enable I/O and Active are checked and communication is possible with the sensor.

Configure Alias and LAN Devices

Use this procedure to configure a device using a LAN interface. Also use for ANY device for which you want to set an alias (easily-recognized) name. The alias name appears in the Available field.
when Aliases is selected as the Interface.

1. On the VNA, minimize the VNA application.

2. In the system tray (lower-right corner) right-click the IO icon, then click **Keysight Connection Expert**

To Add a LAN Device:

1. In **Keysight Connection Expert**, click **Manual Configuration** tab.

2. Select **LAN Instrument**

3. Click, then enter the IP address of the external source.

4. Click **Test This VISA Address** to verify communication.

5. Click **Accept**.

To create an Alias for a connected device:

1. In the list of connected instruments, click the instrument, then click **Add or Change Aliases**.

2. Enter the Alias Name to be used in the **External Device Configuration** dialog.

**Power Sensor Configuration** dialog box help

To launch this dialog, with the PMAR device selected in the **External Device Configuration** dialog, click **Device Properties**.
This dialog is used to configure a power meter / sensor for use as a receiver.

To configure a single power meter for a Source Power Cal, use the Power Meter Settings dialog.

**About Power Sensor Calibration**

PMAR traces are NOT calibrated using standard VNA calibrations, including response corrections.

PMAR traces are calibrated using methods that are appropriate for the selected sensor. Follow the proper guidelines for zeroing or calibrating the sensors that are in use. Check to ensure that the selected sensor is appropriate for the frequency range and the power level at which PMAR measurements occur.

The VNA does not automatically prompt you to perform a calibration.

To calibrate a power sensor, click **Settings** on this dialog box, then click **Zero/Calibrate Sensor**. Learn more.

**Note:** By default, a PMAR is de-activated when the VNA is Preset or when a Instrument State is recalled. This behavior can be changed with a Preference setting.

**Sensor**

For power sensors that are connected to a power meter, select a sensor to configure.

**Settings** Click to launch the Power Sensor Settings dialog.

When pressed, communication with the sensor is tested. Sensor settings can NOT occur unless **Enable I/O** is checked on the External Device Configuration dialog, and the sensor is properly connected and configured.

**Sensor Settling**

Each power meter reading is "settled" when either:

- two consecutive meter readings are within this Tolerance value or
- when the Max Number of Readings has been met.

The readings that were taken are averaged together to become the "settled" reading.

**Tolerance** When consecutive power meter readings are within this value of each other, then the
reading is considered settled.

**Max Number of Readings** Sets the maximum number of readings the power meter will take to achieve settling.

**Sensor Loss Compensation**

**Use Loss Table** Select this checkbox to apply loss data to Source Power calibration correction (such as for an adapter on the power sensor).

**Edit Table** Invokes the Power Loss Compensation dialog box.

**Power Meter Uncertainties**

**Uncertainty...** Select this button to set up power meter uncertainties for power uncertainty calibrations. Then, when the Use uncertainties option is selected during a guided calibration, uncertainty power values will include the uncertainty of the power meter.

Clicking on the Uncertainties... button opens a dialog to either select a specific power meter from a list or load a custom file of power meter uncertainties.

---

**Power Sensor Settings** dialog box help

This dialog appears when you click the Settings button on the Configure Power Sensor dialog.
**Note:** Be sure that the frequency range of your power sensor covers the frequency range of your measurement. This does NOT occur automatically.

**Sensor A (B)** Displays one of the following messages depending on type of sensor.

- **Not connected** The VNA is not detecting a power sensor.

- **Sensor Data** Allows the following entries for power sensor data:
  
  - **Reference Cal Factor** Specifies the Cal Factor for the 50 MHz reference signal.
  
  - **Cal Factor Table** Specifies the frequency and corresponding Cal Factor for the sensor.
  
  - **Delete Cal Factor** Deletes the indicated row in the table.
  
  - **Delete All** Deletes all data in the table.
  
  - **To Add a Row** to the table, click on a row in the table and press the down arrow on either the VNA front panel or keyboard. A row is added to the bottom of the table. The table is automatically sorted by frequency when OK is pressed.

- **Cal factors are contained within this sensor** Internal Reference Cal Factor and Cal Factor data are loaded automatically and the following dialog appears.

  ![Power Sensor Settings](image)

  - **Use Internal Cal Factors** This box only appears when internal cal factors have been detected for the sensor and by default will be checked. Clear this box to not use internal cal factors.

**Load Cal Factors** Click to load cal factors from a *.csv file that you create from the cal factors that appear on the sensor. The first line of the file MUST have the reference Cal Factor (typically...
100), followed by Freq / Cal Factor pairs as show in the following image:

**Save Cal Factors** Click to save the cal factor table to a *.csv* file.

**Limit Frequency Range**

- Check to limit the use of the power sensor to those within the Minimum and Maximum frequency values.

- Clear to use the power sensor for all measurements. If the measurement frequency is not within the Minimum and Maximum frequency values, the closest min or max correction data is used for the measurement.

**Minimum Frequency** Specifies the minimum frequency range for the sensor.

**Maximum Frequency** Specifies the maximum frequency range for the sensor.

**Zero and Calibrate the Power Sensor**

For highest accuracy, Zero AND Calibrate the power sensor before measuring data. Follow prompts that may appear.

**Zero** - If the following settings are 'greyed', Internal or External zeroing is selected automatically based on the power meter/sensor model. Otherwise, select the appropriate type of zeroing to perform, then press *Zero*.

- **Internal Zero** - A switch inside the power sensor removes the sensor from the incident power.

- **External Zero** - Requires that you physically remove the sensor from incident power.
Note for the U2000 Series USB power sensors

Calibration is NOT available. Select External Zero ONLY when the power to be measured is below the specified level. Otherwise, the U2000 series performs internal zeroing automatically when needed. See your power sensor documentation for more details.

- U200xA - below -30 dBm
- U200xH - below -20 dBm
- U200xB - below 0 dBm

If your U2000 power sensor 'hangs' when external zeroing, upgrade the power sensor firmware to Rev. A.01.02.00 or higher to fix this problem.

Calibrate - Available when the selected sensor has calibration capability. Calibration involves measuring an internal 1 mW source.

- Keysight P-Series sensors have an internal reference so you can calibrate them without connecting to the meter’s reference port.
- Keysight U2000 USB power sensors do not require calibrating.
- For other sensors, refer to the documentation to determine if it has calibration capability.

Press Calibrate, then follow the prompts.

Power Loss Compensation dialog box help

![Power Loss Compensation dialog box](image)
To Add a Row to the table, click on a row in the table and press the down arrow on either the VNA front panel or keyboard.

To Edit a value, double-click in the cell to be edited.

Compensates for losses that occur when using an adapter or coupler to connect the power sensor to the measurement port. These components will be removed when the calibration is complete. To account for components that will remain during the measurement, use the Power Offset setting.

The Frequency / Loss pairs define the amount of loss for the entire frequency range. For example, using the entries in the above dialog image:

- 0.5 dB is used to compensate power sensor measurements up to 1 GHz.
- Each data point between 1 GHz to 2 GHz is linearly interpolated between 0.5 dB and 1 dB.
- 1 dB is used above 2 GHz.
- A single frequency/loss segment is applied to the entire frequency range.

Beginning with A.09.80, enter up to 9999 segments to achieve greater accuracy. Previously the limit was 100.

Note: Large segment counts with one or more power sensors can result in long load and close times for the VNA Application.

**Frequency** Enter a frequency in Hz.

**Loss** Enter a loss as a POSITIVE value in dB. To compensate for gain, use NEGATIVE values.

**Delete Table Segment** Deletes row indicated in the field.

**Delete All** Deletes all data in the table.

The Power Loss Compensation table survives VNA Preset and Power OFF. To NOT use Loss compensation, clear the Use Loss table checkbox on the Configure Power Sensor dialog.

**Use a PMAR Device to confirm a Source Power Cal**

Learn how to create and configure PMAR device.

After a Source Power Cal has been performed, use the same sensor as a configured PMAR to analyze the accuracy of the Calibration.
1. Create a PMAR device with the power sensor that will be used for the Source Power Cal.


3. Create an unratioed measurement with the PMAR device. Learn how.

4. With the power sensor still connected to the test port, monitor the corrected source power using Min and Max markers or the Trace Statistics peak-to-peak feature.
Configure and Use External Pulse Generators

Once configured, one or more 81110A External Pulse Generators can be accessed from the VNA Integrated Pulse Application. The external pulse generators can be used without Opt. S93025A/B (internal pulse generators). However, the Integrated Pulse App is available ONLY with Opt. S9x025A/B.

Only the 81110A Keysight Pulse Generator is supported.

In this topic:

- How to Configure an External Pulse Generator
- Pulse Generator Configuration dialog box help
- Using External Pulse Generators with the Integrated Pulse App

See Also

Integrated Pulse Application

IF Path Configuration

81110A Quick Start Guide.

How to Configure an External Pulse Generator

1. **Important**: Create an External Pulse Generator device by name (one-time). Learn how (separate topic).

2. On the Configure an External Device dialog, click Device Properties (this topic).


Using **Hardkey/SoftTab/Softkey**

1. Press **Setup > External Hardware > External Device...**

Using a mouse

1. Click **Instrument**
2. Select **Setup**
3. Select **External Hardware**
4. Select **External Device...**

[Programming Commands]
Tip: In the External Device Configuration dialog, you can configure the same 81110A twice; once for each output module. For example:

- Name = "81110A-1" Output = Out1
- Name = "81110A-2" Output = Out2

Pulse Generator Configuration dialog box help

System Settings

**Time out** - Set the amount of time allowed to communicate with the external pulse generator. If communication has not been established before this amount of time has elapsed, a Timeout message will appear. Check connection settings on the External Device dialog.

**Primary Mode** - When checked, the 81110A trigger mode is set to Internal. This also causes the 81110A to appear as a selection on Integrated Pulse App, Pulse Trigger setting. When selected here and on that dialog, the timing of configured 'follower' pulse generators is controlled by the 81110A pulse generator. Although more than one configured pulse generator can have the Primary Mode setting checked, only one pulse generator can be connected to the rear-panel Pulse connections. Learn more about making physical connections.

When this setting is cleared, the 81110A trigger mode is set to External and can be configured as a 'follower' pulse generator to the VNA internal pulse generators or another external pulse generator.

Output Settings

The following are 8110A settings made by the VNA. Some settings may not be possible depending on the modules that are installed on the 81110A. Please refer to the 81110A Quick Start Guide for...
Using External Pulse Generators with the Integrated Pulse App

Once configured, an external pulse generator can be used with the Integrated Pulse App as though it were an internal pulse generator.

N1966A Pulse I/O Adapter. See an enlarged view of the IF Block diagram

An External Pulse Generators can be used for ONE OR MORE of the following pulsed functions within the Integrated Pulse Application.

- Modulate the sources
- Drive the IF (Receiver) Gates (Narrowband mode ONLY).
- Trigger the ADC to make receiver measurements (Wideband mode ONLY).

How to Modulate a Source with an External Pulse Generator

When using an external source modulator (Z5623AH81):
Connect: the 8110A to the Z5623AH81 as shown in the Narrowband Pulse topic.

Setting: On the Pulse Generator Setup dialog, disable (clear) the Internal Pulse Modulators.

When using internal source modulators, the external pulse generator can drive the internal modulators in two ways:

- 81110A directly to the internal pulse modulators.
  - Connect: 81110A to RF Pulse Mod In on the N1966A OR rear-panel Pulse I/O connector.
  - Setting: On the Pulse Generator Setup dialog, set Modulator Drive to "External".

- 81110A drives internal pulse generators, which drives the internal modulator.
  - Connect: 81110A to Pulse Sync IN, on the N1966A OR rear-panel Pulse I/O connector.
  - Settings:
    - On the Pulse Generator Configuration dialog (above) check Primary mode.
    - On the Pulse Setup dialog, set Pulse Trigger to <ext pulse gen name>.

How to Gate IF Receivers with an External Pulse Generator

(Used ONLY in Narrowband mode.)

When IF Gating is used, the external drive can be routed in two ways:

- 81110A drives gates directly at the rear-panel IF Gate inputs.
  - Connect: 81110A to the Pulse IN for one or more VNA receivers on the N1966A OR rear-panel Pulse I/O connector.
  - Setting: On the Pulse Setup dialog, under Measurement Timing, for the receivers to be gated, set Pulse Gen to <ext pulse gen name>.

- 81110A drives the internal generators, which drive the gates.
  - Connect: 81110A to Pulse Sync IN, on the N1966A OR rear-panel Pulse I/O connector.
  - Settings:
    - On the Pulse Generator Configuration dialog (above) check Primary mode.
    - On the Pulse Setup dialog, set Pulse Trigger to <ext pulse gen name>.
On the Pulse Setup dialog, under Measurement Timing, for the receivers to be gated, set Pulse Gen to the internal pulse generator (Pulse0 through Pulse4) to be used to pulse the Rcvr<n>. Set unique pulse Width and Delay for the Receiver.

**How to trigger the ADC with an External Pulse Generator**

(Used ONLY in Wideband mode).

Pulse0 may be used to trigger the ADC. The following shows how P0 may be driven by an external pulse generator.

- **Connect:** 81110A to Pulse Sync IN, on the N1966A OR rear-panel Pulse I/O connector.

- **Settings:**
  - On the Pulse Generator Configuration dialog (above) check Primary mode.
  - On the Pulse Setup dialog, set Pulse Trigger Source to <ext pulse gen name>.
  - On the Pulse Setup dialog, under Measurement Timing, for the receivers to be triggered, set Pulse Gen to Pulse Trigger. Set Delay for the Receivers.
Configure an External SMU

An external SMU Source Measure Unit is a combined DC Source and Meter. The VNA supports the Keysight B2900 series and N6700 series of SMUs. Only the B2900 series offers Hardware Triggering (learn more).

Learn more about the Keysight SMU offerings at: http://Keysight.com/find/smu

For all other External DC devices, select DC Source or Meter Device Type. Learn more about configuring those devices.

How to Configure a SMU

1. **Important:** Create an SMU by name (one-time). Learn how (separate topic).

2. On the Configure an External Device dialog, click **Device Properties.** (This topic).

### Using Hardkey/SoftTab/Softkey

<table>
<thead>
<tr>
<th>Using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Click <strong>Instrument</strong></td>
</tr>
<tr>
<td>2. Select <strong>Setup</strong></td>
</tr>
<tr>
<td>3. Select <strong>External Hardware</strong></td>
</tr>
<tr>
<td>4. Select <strong>External Device...</strong></td>
</tr>
</tbody>
</table>

Once configured, set the SMU source voltage and display Voltage / Current Meter measurements:

- **SMU Sources:** DC Source control is available in Standard and Gain Compression/GCX channels. Set the Start and Stop voltage on the **DC Control dialog.** To access this dialog:
  
  - Press **Sweep > Source Control > DC Source...**

- **SMU Measure:** DC measurements are available in Standard and Gain Compression/GCX channels.
  
  - In Gain Compression/GCX channels, display DC parameters as you would an RF parameter, by pressing **Trace > Trace Setup > Measure...**

  - In a Standard channel, configure an unratioed measurement. Press **Trace > Trace Setup > Measure... > Receivers.** For Numerator, select the external (or internal) DC meter.

  - Change the X-axis to display the DC Meter parameters by pressing **Sweep > Main > X-axis Type,** then select the DC Meter.
To start this dialog, on the External Device Configuration dialog, select SMU as the Device Type. Then click Device Properties.

SMU Information

The DC device names for six DC devices are listed. The names are made from the root device name (given in the External Device Configuration dialog) plus a suffix.

In the above image (example):

- **Device0** is the root device name.
- **"_Src1"** means voltage source of channel 1.
- "\text{VM1}\" means voltage meter of channel 1.
- "\text{AM2}\" means current meter of channel 2.
- Check \text{Chan} \langle n \rangle to enable that channel in the SMU device.
- \textbf{Module Selection} selects the SMU module in the B1500A mainframe to use for the channel.

<table>
<thead>
<tr>
<th>Module Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Module installed in slot 1</td>
</tr>
<tr>
<td>2</td>
<td>Module installed in slot 2</td>
</tr>
<tr>
<td>3</td>
<td>Module installed in slot 3</td>
</tr>
<tr>
<td>4</td>
<td>Module installed in slot 4</td>
</tr>
<tr>
<td>5</td>
<td>Module installed in slot 5</td>
</tr>
<tr>
<td>6</td>
<td>Module installed in slot 6</td>
</tr>
<tr>
<td>7</td>
<td>Module installed in slot 7</td>
</tr>
<tr>
<td>8</td>
<td>Module installed in slot 8</td>
</tr>
<tr>
<td>9</td>
<td>Module installed in slot 9</td>
</tr>
<tr>
<td>10</td>
<td>Module installed in slot 10</td>
</tr>
</tbody>
</table>

\textbf{Trigger Settings}

\textbf{Trigger Mode}: Choose from:

- \textbf{Software CW (GPIB)} Slowest method. The SMU receives each stimulus voltage from the VNA over GPIB, USB, or LAN. No other trigger cables are required.
- \textbf{Hardware List (BNC)} Available ONLY in B2900 series. Fastest method. The SMU receives a list of stimulus voltages from the VNA, then receives trigger signals though a rear-panel connector when appropriate from the VNA.
Note: The B2900 series can store no more than 2500 data measurements. Therefore, this mode is NOT allowed when the VNA data points exceed 2500 points.

Note: Hardware List trigger mode is NOT supported in GCA 2D sweeps.

- **SMU Trigger In / Out** - Available ONLY when the trigger mode is Hardware List. Select the trigger in and trigger out pins on the B2900 digital I/O connector. Connect these pins to the either pair of VNA AUX IO IN and OUT connectors. The VNA AUX trigger pair is automatically selected.

---

**External SMU - Source / Voltage / Current Meter** dialog box help

The descriptions for this dialog pertain to the Source, Voltage, and Current Meter tabs. Exceptions are highlighted.

**SMU Chan**  Select the SMU channel for which the settings on this page apply.

**Source type (Source ONLY)**  Select either a voltage source or current source.

**Device Settings**

**Timeout** - Sets a time limit for the DC source or meter to make contact with the VNA. If this time limit is exceeded, the VNA stops the measurement and displays the following error message.

**EXECUTION ERROR;OPC QUERY TIMEOUT ERROR**
If this occurs, check the connections between your VNA and external device.

**Dwell Before Sweep**  Wait time before making a sweep.

**Dwell After Point**

- **Source**  Wait time after setting the voltage/current at each data point.
- **Voltage / Current Meter**  Wait time before measuring voltage/current at each data point.

**Current / Voltage Limit** *(Source ONLY)*  Depends on the Source Type selection)  Sets the maximum current or voltage to be allowed from the source.

**Note:** To change the X-axis to display the Meter units, click **Response**, then **Display**, then **Labels**, then **Select X-Axis**, then select the SMU Meter.

**Receiver / Source Correction**

- For a **Source**, use the correction settings to scale and offset the output voltage or current.
- For a **Voltage or Current Meter** (receiver), use the correction settings to display and scale measurements.  For example:
  
  Measure the voltage across a 5 ohm resistor, then display the results in A(mperes).

  Using ohms law,  \( I = \frac{V}{5 \text{ ohms}} \) or \( I = V \times 0.2 \)

  For receiver correction, enter Scaling = 0.2; Offset = 0.

**ON**  Check to apply the following correction factors to each measurement.

**Offset:** Enter the value to offset the Meter reading or set the Source voltage.

**Scaling:** Enter the value to scale the Meter reading or set the Source voltage.

\[
\text{Displayed Output} = \left( \frac{\text{Measured}}{\text{Set value}} - \text{Offset} \right) \times \text{Scaling value.}
\]

**Edit Commands**  - Click to start the Edit Commands dialog. This feature is usually not necessary with an SMU external device.

**Save** - Press to save the edited SMU configuration to an *.xml file. The list of files is NOT filtered so use a descriptive filename.

**Load** - Press to load an existing configuration.
The SMU Source and Meter Edit commands are almost identical in how they operate. Both are documented here.

Global Tab

The Global tab includes the system settings for the SMU Source.

**ID Query** - Enter the SCPI command to return the ID string of the DC Source. Typically *IDN? This entry can be left blank.

**Error Query** - Enter the SCPI command that is used to return DC Source errors. Typically SYST:ERR?

**Enable I/O** - Enter the SCPI commands that is used to enable the DC Source to output voltages.

**Disable I/O** - Enter the SCPI commands that is used to disable the DC Source from outputting voltages.

**Maximum / Minimum DC Output (Source ONLY)**

**Read Max / Min Using** - Select, then enter the commands used to return the output limits of the SMU source.

**Define Max / Min As** - If the SMU has no commands to return these values, or you would rather define the limit for your SMu, then enter the Max and Min voltage limits.
Test Connection

Click to start the Test Connection dialog. You must first have entered the I/O Configuration settings and select Enable IO on the External Device dialog.

Enter a SCPI command, then click **Send** or **Send&Read** when a return value is expected.

Sweep Tab

Use the Sweep Tab to send SCPI commands to the SMU at the beginning or end of each sweep.

**Abort Sweep** - Enter the SCPI command that is used to Abort or reset the SMU. This would be necessary when the VNA sweep is aborted or terminated. The VNA will then send the command to the SMU.

Point Tab
| **Read (commands)** - Enter the SCPI command that is used to make a DC measurement at each data point. |
| **Set (commands)** - Use `<%variable>` to make a DC Meter setting. |
Synchronize VNA/PNA-X with an External Source(s)

The VNA External Source Control feature can be used to automatically control external sources. However, this feature requires certain VNA options. Learn more.

Many VNA measurements require the use of at least two sources. If your VNA has only one internal source, an external source is required. For example, when measuring the insertion loss of a mixer, the LO must be swept at the same time as the RF input. This requires the VNA and external source to be synchronized.

The following procedure shows how to automatically synchronize the VNA/PNA-X with an Keysight external sources.

**Hardware configuration**

- Connect the VNA and External Source(s) Time Base (VNA/PNA-X 10 MHz OUT to External Source 10 MHz IN)

Connect the AUX Trigger I/O connectors as follows (for two sources):

- VNA/PNA-X AUX Trig-1 IN to External Source-1 Trigger OUT
- VNA/PNA-X AUX Trig-1 OUT to External Source-1 Trigger IN
- VNA/PNA-X AUX Trig-2 IN to External Source-2 Trigger OUT
- VNA/PNA-X AUX Trig-2 OUT to External Source-2 Trigger IN

Learn more about the AUX Trigger capabilities.

**VNA/PNA-X Settings**

- Refer to External Source Configuration

**What is Happening?**

The following is a flow diagram showing the handshake / synchronization process between the VNA/PNA-X and an External Source.
1. VNA/PNA-X loads frequency points and source power information onto the external source(s).

2. A trigger signal from the source starts the first data point of the measurement.

3. VNA/PNA-X data acquisition (measurement) starts, and then stops AFTER the first data point acquisition.

4. The VNA/PNA-X sends a trigger signal out to the source telling it to move to the next frequency data point.

5. The external source and VNA/PNA-X move to the next data point. The source usually takes longer than the VNA.

6. The source sends the Ready for Trigger signal to the VNA/PNA-X for next data acquisition.

7. Back to step 4 until last data point.
E5092A Test Set Control

- Overview
- Connecting the E5092A
- How to make E5092A test set Control Settings
- Calibrating with the E5092A

Other System Configuration Topics

Overview

The E5092A multiport test set can be controlled with the firmware of the 4-port PNA Series, 4-port E5080, and M980xA PXI VNAs with more than 4 ports (assuming the port-1 to 4 are connected to the E5092A). The E5092A control functions of GUI settings or corresponding SCPI commands can switch the E5092A’s RF paths and the control line outputs at the beginning of each channel’s sweep. These E5092A control functions associated with the channel are available only in the standard class.

Notes:

- To control the E5092A with the M980xA firmware, the M980xA firmware must be A.15.xx.xx and above, and the S94702A AMX VNA Plugin license is required in the PXI controller.
- The E5092A can be controlled only with the single firmware instance.
- The switching of the test set paths and control line outputs occurs just before the channel is triggered. If the channel trigger state is Hold, the switching for that channel does not occur.
- In addition to the E5092A control functions associated with the channel, there are SCPI commands which switch the E5092A’s RF paths and control line outputs independently of the channel when the commands are executed (CONTrol:MULTplexer commands). These commands are effective in the measurement classes other than the standard class.

Connect and Configure the E5092A

The E5092A can be connected to any one of the VNA USB ports. When first installed, Windows will automatically launch the "Add New Hardware" wizard. Click Next to install the E5092A test set.
Note: See the power handling limitations of the VNA USB ports.

Connect the VNA test ports to the E5092A test ports. Match VNA test port 1 to E5092A test port 1, and so forth.

Selecting ID for E5092A

The VNA can control up to two E5092A test sets. Set the Instrument ID bit switch to 1 or 2. The test sets will then be identified automatically and referred to by the DIP switch setting on the E5092A rear-panel. Change the ID bit switch setting before connecting to the VNA USB.

Power ON

Immediately after power-on, all of the port connection indicator LEDs of the E5092A go ON. Then, after the VNA detects the E5092A, the four LEDs that indicate the connected test ports remain ON. If the VNA is not powered on or if the E5092A is not connected using a USB cable, all of the LEDs stay ON.

How to make E5092A test set Control Settings

Using Hardkey/SoftTab/Softkey

1. Press Setup > External Hardware > External Testset > E5092A...

Using a mouse

1. Click Instrument
2. Select Setup
3. Select External Hardware
4. Select External Testset
5. Select E5092A...
E5092A test set control dialog box help

The title of the dialog shows the test set model and ID number of the active test set.

Select ID  ID of the test set to be configured. Up to two E5092A test sets can be controlled. Click to change test set ID. Learn how to set the test set ID.

Enable Test Set Control  When cleared, port switching and control line settings are disabled. This selection affects all channels using the selected test set.

<table>
<thead>
<tr>
<th>Selection</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>E5092_13</td>
<td>Select the 13-port configuration of the E5092A</td>
</tr>
<tr>
<td>E5092_16</td>
<td>Select the 16-port configuration of the E5092A</td>
</tr>
<tr>
<td>E5092_22</td>
<td>Select the 22-port configuration of the E5092A</td>
</tr>
<tr>
<td>E5092_28</td>
<td>Select the switching independently in the E5092A</td>
</tr>
<tr>
<td>E5092_X10</td>
<td>Select the 10-port full crossbar configuration of the E5092A</td>
</tr>
</tbody>
</table>

Show Test Set Property  When checked, a second row on the status bar appears which indicates the test set that is being controlled and the current port control selection. For example, the following image shows the status bar when controlling an E5092A test set.

A. Configured channel

B. Port Control settings for E5092A
C. Test set Label. E5092A control does not use this field. It is shared between Interface Control and External test set Control. The two labels are separated by /.

Control of the second status bar is completely separate from the first status bar, which is controlled from the View, Status Bar menu.

<table>
<thead>
<tr>
<th>Port Mapping Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Port Control and Control Line settings affect the channel of the active (selected) measurement. These settings will occur as the channel is being measured.</td>
</tr>
<tr>
<td>- Correction is NOT turned OFF when port mappings are changed. However, the calibration is NO LONGER VALID!</td>
</tr>
</tbody>
</table>

Control Lines  Specifies the values of individual control lines. These general purpose control lines on the test set front-panel can be used in your test setup. Each button toggles the control line HIGH and LOW. When first opened, the selections reflect the current control lines. See your test set documentation for more information about the control lines.

OK  When clicked, the changes to the dialog box are implemented and the port selections and control values are immediately sent to the specified test sets. The Port Control and Control line settings are stored with other channel data and used when those channels are swept.

Cancel  (or Escape) Changes to the dialog are not implemented and revert to the settings before the dialog box was opened.

Calibrating with the E5092A

The following are a few changes in the way you calibrate the VNA with the E5092A connected:

1. Create the measurements for the channel and configure the Port Control (switching) on the E5092A Test Set Control dialog box. Enable Show Test Set Property.

2. To calibrate, start the Calibration wizard and select a Calibration method (ECAL, SmartCal, Basic Cal).

3. Select the DUT connectors that are used at the E5092A measurement reference plane.

4. When prompted to connect a standard to a VNA port, instead connect the standard to the E5092A port as indicated on the test set status bar. For example, when the cal wizard prompts to connect the standard to port 1, if the status bar indicates A1 in the E5092_22 configuration, connect the standard to port A1 (5A) of the E5092A.
External Multiport Test Set Control

Test sets are designed to be controlled by the VNA. There is a switch matrix that is controlled over GPIB/LAN/USB interface. They are independent instruments and do not require the PNA/PNA-X.

The VNA with a test set controlled over the test set interface can run in either multiport VNA or standalone VNA mode.

You need to run in multiport VNA mode with option S93551A/B. You cannot use any application measurement class in this mode.

If you want to use an application measurement class with a test set attached to the PNA, you need to run in standalone mode. The VNA does not know the presence of the test set in this mode, and you can use any application measurement classes you wish, but keep in mind that losses through the test set may be significant. To ensure that the test set is routing signals to the PNA ports (and not the test set ports), create the VBS script shown below and run it on the analyzer.

```vbs
Set app = CreateObject("AgilentPNA835x.Application")
Set TS = app.GetExternalTestSetIO

TS.WriteData 0, 0
TS.WriteData 16, 0
TS.WriteData 32, 0
TS.WriteData 64, 0
```

- **Supported Test Sets**
  - Option S93551A/B
  - E5092A (separate topic)

- **Procedure**
  1. Connect Test Set
  2. Restart as Multiport
3. Optional External Test Set Control Settings

- External Test Set Control and other Functions

---

### Other System Configuration Topics

**Supported Test Sets**

The list of test sets that provide integrated solutions with the VNA is constantly growing. For a current list of supported multiport test sets, see [www.Keysight.com/find/multiport](http://www.Keysight.com/find/multiport)

**Option S93551A/B**

- **With** Option S93551A/B enabled on your VNA, **any supported multiport test set** (such as the U3042A E12) can be controlled directly from the VNA to make fully integrated measurements at ALL of the available test ports. To understand what test ports are available to source and receive, see the test set documentation.

- **Without** Option S93551A/B, basic operation depends on the number of VNA test ports.
  
  - For a 2-port VNA, configure two available test ports.
  
  - For a 4-port VNA, configure four available test ports.

Option 550 is no longer available.

**Note:** By default, the system logical test ports are mapped as follows:
- **Port 1** - VNA port 1
- **Port 2** - Test Set port 2
- **Port 3** - VNA port 2
- **Port 4** - Test Set port 4

The ports can be **remapped** using the Port Control Setting.

**Block Diagram of a 2-port VNA and N44xx Test Set**

Click one of the S-parameters to see switch and path changes. Because the test set does NOT contain receivers, **measurement speed** and **calibration** can be affected.
Procedure - How to enable full Multiport Capability

1. Enable Option S93551A/B.

2. Connect the test set to the VNA using the documentation that was shipped with the test set.

3. Restart as Multiport VNA

4. Make optional External Test Set Control Settings

Connect and Configure the Test Set

Connect the test set to the VNA using the test set documentation. Most test set documentation can
Test Set I/O-controlled test sets

Test sets that are controlled using the Test Set I/O connector have NO return communication capability. The VNA sends commands out the rear panel connector. It is assumed that the test set is responding appropriately. The "Active" LED, located on the test set front panel, should light when the test set is addressed in Multiport Mode or manual operation. When the test set is not in use, the Active LED will be OFF.

GPIB-controlled test sets

Connect the test set to the GPIB using one of the following methods:

- If the VNA will NOT be controlled by a remote computer using GPIB, then the test set can be connected directly to the VNA GPIB port. The VNA is automatically switched to System Controller mode.

- If the VNA WILL be controlled by a remote computer using GPIB, then learn how to connect the test set

Restart as Multiport VNA

How to Enable Multiport capability

**Note:** If Option S93551A/B has not been enabled, the following Multiport Capability menu selection will NOT be available.

Using Hardkey/SoftTab/Softkey

1. Press **Setup > External Hardware > Multiport > Multiport Configuration...**

Using a mouse

1. Click **Instrument**
2. Select **Setup**
3. Select **External Hardware**
4. Select **Multiport**
5. Select **Multiport Configuration...**

Multiport Restart dialog box help
See Also External Test Set Control and other VNA Functions

After the test set is connected and **VNA Option S93551A/B is enabled**, the following settings are used to enable Multiport operation.

**Test Set** Select the test set file to load. Only the files that are appropriate for use on that VNA model are displayed.

**To Add a new Test Set file:**

1. On the VNA, click **File**, then **Exit** to quit the VNA application.
2. Download the Test Set file from: http://na.support.keysight.com/multiport/testsetsupport.html/
3. Save it to the VNA c:\users\public\network analyzer\testsets
4. Start the VNA application.
5. Click **Instrument**, then **Setup**, then **External Hardware**, then **Multiport Config**.
6. In the Multiport Restart dialog, click **Restart as multiport VNA with this testset**.
7. The new test set should now be visible from the **Testset**: menu.

**Address** Enter the test set address if the test set is connected to GPIB. Connections over the Test Set I/O connector are determined by their sequence.

- **Restart as a standalone VNA**. The VNA shuts down and restarts as a standard VNA. If the test set is left connected to the VNA, switch the test set OFF, then back ON to ensure that the test set routes signals to ports 1 and 2 of the VNA. In this condition, there is more loss in the test paths than without a test set connected. If the power switch is OFF, there is SIGNIFICANTLY more loss in the test paths.

- **Restart as a multiport VNA with this testset**. The VNA shuts down and restarts as a multiport VNA with
the selected test set.

Click **OK** The VNA shuts down and restarts in the selected configuration.

To learn how to change port mapping, see Port Control.

**Problems**

If the VNA cannot find the test set, the following error is displayed on the VNA:

**GPIB ERROR Address xx cannot open VISA session.**

To correct the problem, verify the following:

- The test set is connected to the VNA using one of the methods described above.
- The correct test set address is set.
- The test set is turned ON.

**Important: After the problem has been fixed:**

1. On the External Test Set Control dialog, click Enable Test Set Control.
2. Restart Triggering - press **Trigger > Main > Continuous**.
3. The VNA again tries to find the test set.

**External Test Set Control Settings**

The following External Test Set Control Settings are used to configure Multiport test sets. For the N44xx test sets, the only setting that is necessary is port control.
How to access the External Test Set Control Settings

Using **Hardkey/SoftTab/Softkey**

1. Press **Setup > External Hardware > Multiport > Other Testset Setup...**

Using a mouse

1. Click **Instrument**
2. Select **Setup**
3. Select **External Hardware**
4. Select **Multiport**
5. Select **Other Testset Setup...**

---

**External Test Set Control** dialog box help

![External Test Set Control dialog box](image)

**Important Notes:**

- With Option S93551A/B, **first** use the **Multiport Restart** dialog to **Restart as Multiport VNA with this test set.** The test set file is loaded and the test set is enabled automatically.

- When using GPIB to control an external test set, the VNA is automatically put in **System Controller mode.**

- See also **External Test Set Control and other VNA Functions**
Select ID

- For N44xx test sets: the GPIB address
- For other Multiport test sets: either GPIB address or 0 for Test Set I/O controlled test sets.

Enable Test Set Control When cleared, port switching and control line settings are disabled. This selection affects all channels using the selected test set. When checked, the 'Show Test Set Properties' checkbox is also checked automatically.

Load Test Set File For operating without Option S93551A/B.

If your Test Set is not visible, see Add a new Test Set.

The selected test set file is loaded.

1. Navigate to the folder: C:\Program Files(x86)\Keysight\Network Analyzer\testsets\  
2. Select a test set control file.

The title of the dialog shows the model of the test set file that is currently loaded.
See a list of supported test sets.

Show Test Set Properties This box becomes checked by default when the Enable Test Set Control is checked. When checked, a second row on the status bar appears which indicates the test set that is being controlled and the current port control selection. For example, the following image shows the status bar when controlling an E5091A test set and a Z5623A K64 test set:

A. Configured channel
B. Port Control settings for E5091A
C. Port Control settings for Z5623A K64
D. Test Set Label. This field is shared between Interface Control and External Test Set Control. The two labels are separated by /.

Control of the second status bar is completely separate from the first status bar, which is controlled from the View, Status Bar menu.

Test Set Label NOT available with option S93551A/B. Add text to appear on the second status bar
when **Show Test Set Properties** is checked. See image above.

**Port Control** Controls **mapping** of Physical ports to Logical ports. *(Refer to image of dialog box above.)*

- Physical ports are the port numbers that are labeled on the test set front panel.
- Logical ports are the port numbers that are referred to by most of the VNA application prompts and dialog boxes.

**Port Mapping Notes**

- Port Control and Control Line settings effect the channel of the active (selected) measurement. These settings occur as the channel is being measured.
- Correction is turned OFF when port mappings are changed.
- After the physical ports are mapped to logical ports, all VNA references to port numbers refer to LOGICAL port numbers. The only exception to this is during calibrations.

**Control Lines** For use with the U30xxA test sets. Specifies the values of individual control lines. These general purpose control lines on the test set front-panel can be used in your test setup. Each button toggles the control line HIGH and LOW. When first opened, the selections reflect the current control lines. See your test set documentation for more information about the control lines.

**OK** When clicked, the changes to the dialog box are implemented and the port selections and control values are immediately sent to the specified test set. The Port Control and Control line settings are stored with other channel data and used when those channels are swept.

**Cancel** (or Escape) Changes to the dialog are not implemented and revert to the settings before the dialog box was opened.

---

**External Test Sets and other VNA Functions**

The following features may work differently with a test set connected to the VNA.

**Remote Commands**

- See **SCPI** and **COM** commands for controlling an External Test Set.
- Use `*OPT?` (SCPI) or `NumberOfPorts` (COM) to query the number of ports for a VNA/External Test set.
- Use **logical receiver notation** to refer to double-digit ports.

### Sweep Settings

To compensate for additional cable lengths:

- Set to Stepped sweep
- Set **Dwell time** to at least 5 microseconds.

### Interface Control

When both **Interface Control** and External Test Set Control are configured, the commands on the Interface Control **Before Sweep Start** tab are sent out before any External Test Set Control commands are executed on that channel. Similarly, commands on the **After Sweep End** tab are sent after Test Set Control commands.

### Calibration

With an External Test Set connected, calibration is performed exactly like a VNA with the following exceptions:

- Correction is turned OFF when port mappings are modified. This also applies to Source Power Cal.
- Beginning with VNA Rev. A.07.50, for **TRL Cal**, **QSOLT**, or **Unknown Thru** cals with external test sets:
  - Use of a **Delta Match Calibration** from a User Cal Set is NOT required. However, for PNA-L models that require Delta match, a **Global Delta Match Cal** must be present. The Global Delta Match Cal can only be performed in stand-alone mode.
  - You can **NOT** perform any of those 3 cal types on **JUST** a pair of ports that share a test port receiver, such as Port 1 and Port 2 of a 4-port system. You would need to include an additional port in the calibration.
- With an External Test Set connected, you may be required to perform more than **3 THRU connections**.
- A test set such as the Z5623A K44 which is used with 4-port VNA models, does not terminate ports that are not currently in the source path. Because a ports load match on this system is not constant for all possible ports it can be paired with, when calibrating more than two total ports it may be necessary to make Thru measurements on more than the usual minimum number of Thru paths for a VNA calibration. The VNA will ensure that multiport calibrations use a sufficient set of Thru paths so that the calibration can correct for those variations in load match on this type of multiport system.
- As with ALL VNA calibrations, when error correction is **ON**, both forward and reverse sweeps are required for **EACH** port pair that is corrected, even if only one reflection measurement is displayed. For example, any displayed measurement with full 4-port calibration **ON** will require 12 measurement sweeps. **Learn more.**
**Source Power Cal**

*Source power calibration* involves adjusting the source so that the power at an output port is flat across a frequency range. Because of additional loss through some of the test set paths, it may NOT be possible to obtain corrected output power because of limitations on the source signal.

During a Source Power Cal, you are prompted when and where to connect the power sensor. When one of the supported test sets are connected, the prompt refers to the PHYSICAL port number, NOT the LOGICAL port number. To help with translating physical to logical port mappings, enable *Show Test Set Properties*.

**Measurements with Shared Receivers**

External test sets do not contain receivers. The VNA receivers are always used to measure signals at the external test set ports. Therefore, when a channel contains two measurements that share a VNA test port receiver, additional sweeps are necessary.

For example, to make S34 and S44 measurements in the same channel with correction OFF:

- On a 4-port VNA, only ONE sweep is required using the C (port 3), D (port 4), and R (reference for All receivers).
- On a N44xx system, TWO sweeps are required since both measurements use the B and R2 receivers. See interactive block diagram above.

**Create Ratioed and Unratioed Measurements**

When using an external test set, it IS possible to create a Ratioed measurement using two logical receivers that share the same physical VNA receiver. However, this measurement data is NOT valid. Invalid measurement traces show all data at -200 dB (in Log mag format). Learn about Logical Receiver Notation

**Logical Port References**

When an external test set is enabled, all references to VNA port numbers and test set port numbers (except during calibrations) refer to LOGICAL port numbers. Logical ports can be remapped using the *Port Control settings*. During a calibration, you are prompted to connect standards to physical port numbers.

**Balanced Port Mapping**

"Logical Ports" is a term that is used with both External Test Sets and balanced measurements. While the concept is the same, they refer to different scenarios. The two can be easily confused when making Balanced measurements with an External Test Set connected. The important principle to remember is the order in which the logical ports for each are mapped:
1. In the **External Test Set - Port Control settings dialog**, the physical VNA ports and test set ports are mapped to logical ports as noted above.

2. In the **Balanced Topology Dialog**, the new (step 1) logical ports are mapped again to become Balanced logical ports.

**Preset**

**Instrument Preset** will reset **Port Control** settings to defaults and remove the **test set label**. All other settings remain. To maintain port control settings and the test set label, create a **User Preset**.

**Instrument State Save and Recall**

**Instrument State files** include Test Set model, Enable and Status bar settings, and Port mappings and DUT control values for each channel.

If an Instrument State recall requires that a test set configuration file be loaded, recall time may be significant. For example, this would occur if a 2-port VNA with attached test set is configured as a 2-port VNA and then recalls a state file which requires 4-port operation.

**Recall Cal Sets**

If a Cal Set is saved while an external test set is enabled, when the Cal Set is recalled, then the external test set must be enabled or an error message is displayed.

**Copy Channel**

**Copy Channel** copies all relevant test set data from the source channel to the target channel.

**Applications**

No VNA applications are supported with External Test Set Control. These include FCA (opt S93083A/B), SMC (opt S93082A/B), GCA (opt S93086A/B), NFA (opt S93029A/B), Pulsed (opt H08).
Print

Port mapping information appears on the Channel Settings Table when printing.

Save sNP Files

To save sNP data with an external test set enabled, click File, Save As, then select Snp File(*.s*p), then complete the Choose ports dialog.
Display Colors

You can modify the colors that are used to draw various elements on the VNA screen and on a hardcopy print of the display.

See Also

Print Preview

How to modify DISPLAY Colors

These settings can also be accessed from the Preferences dialog box.

Using Hardkey/SoftTab/Softkey

1. Press System > System Setup > Preferences... > Colors....
2. Click Display Colors... in the Customize Display dialog box.

How to modify PRINT Colors

1. Press System > System Setup > Preferences... > Colors....
2. Click Print Colors... in the Customize Display dialog box.

Display and Print Colors dialog box help

The Display Colors and Print Colors dialog boxes function in exactly the same manner. See Print Preview procedure below.
"Pen" is a term used to describe the various elements. Each pen can have a unique color.

You can change the color of the following pens:

- **Background** - The background color of the inactive windows.
- **New Active Background** - The background color of the active window.
- **Grid** - The inner lines of all grids in all windows, and the grid frame in inactive windows.
- **Active Labels, Grid Frame** - The labels and grid frame colors in the active window. **Note**: when this pen is selected, the current window becomes inactive. Therefore, changes for this pen color will not be visible until **OK** is pressed.
- **Inactive Window Labels**
- **Failed Trace** - **Limit Line** failed traces or failure indicators (dots) and the word Fail.
- The following pens for up to 8 Traces:
  - **Data and Limits**
  - **Memory trace**
  - **Markers**
  - **Memory markers**

**About Trace Pens**

'1st Trace' is NOT always Trace1 (Tr1). For example, the first trace in a window might be Tr2 which is drawn with the "1st Trace" pen.

The first 8 traces are drawn with the defined pen colors. The next eight traces reuse the same colors, and so forth. For example, if all traces are numbered sequentially, the 9th and 17th traces are drawn using the same color as the 1st trace.

**Change Color**  Click the button or the color swatch to launch the **Change Color** dialog.

**Reset Color**  Restores the default color for the selected pen.

**Color Themes**

A theme is a complete set of pens and their colors. The current theme persists until you change it. Themes can also be saved to a file and then later recalled.
- **Save Theme**  Click to save the current set of pens to a file.
- **Recall Theme**  Click to recall and use a saved theme.
- **Reset Theme**  Click to recall the default VNA color theme.

The colors for the following Display elements can NOT be changed: toolbars, softkeys, menus, dialogs and popup messages.

---

**Change Color** dialog box help

To use a basic color, click the color from the 'Basic colors' palette, then click **OK**.

To define and use a custom color:

1. Click **Define Custom Colors>>** to open the right side of the dialog.
2. Optionally, pick a Custom color slot to replace. Otherwise, the replacement will occur at the first slot location and continue with subsequent custom color definitions.
3. Click the color pane, or drag the crosshairs, to the location of the custom color.
4. Drag the arrow to the desired saturation level of the custom color.
5. Click **Add to Custom Colors**
6. Continue to define more colors, or click **OK** to close the Color dialog.
After a custom color has been assigned to a VNA pen, the custom color can be changed. The VNA pen color remains unchanged.

**Print Preview Procedure**

Use the following procedure to preview your Print Colors on the VNA screen:

1. From the Print Colors dialog, select **Reset Theme** then **Save Theme**. Name the new theme “MyPrintTheme.colors”. This will give you a starting point equal to the default print colors.

2. Launch the Display Colors dialog, select **Recall Theme**, then select “MyPrintTheme.colors”. The display will now show the default print theme.

3. Customize the display colors. You will be previewing how the hardcopy will appear when printed.

4. Save the customized display colors to “MyPrintTheme.colors”.

5. Go to the Print Colors dialog and Recall “MyPrintTheme.colors.”
## Display Update

### How to set Display Update

**Using Hardkey/SoftTab/Softkey**

1. Press **Display > Display Settings > Display Update** (Turns ON display update).

2. Press **Display > Display Settings > click left side small button of Display Update** (Turns OFF display update).

### Notes:

1. Turn ON/OFF the Display update will result on the **Status Bar**.

2. Update State ON/OFF is part of the save/recall state.

3. Softkey(s) exist for update state ON/OFF and immediate update.

4. Disabling the display update will yield the most significant performance improvements. The performance improvement (due to disabling updates) for a single channel and window state seems negligible. However, with a large number of channels, windows and traces, it should make more of a difference (but disabling the display update will provide more performance improvement). For example, Performance Oddities, the following is looking at the INDEX line (Handler I/O) for a 1.0 GHz to 1.2 GHz with 201 point sweep (otherwise Preset condition apply). When high, the analyzer is sweeping. There is 20 ms of dead time between many of sweeps.
Mechanical Devices

- Overview
- How to access Mechanical Devices settings
- Mechanical Devices dialog

Other System Configuration Topics

Overview

**Note:** To prevent premature wear, the VNA does not allow attenuators or other mechanical switches to switch continuously.

These mechanical devices are set for the entire channel. When more than one channel is used, and a mechanical device setting is NOT the same for all channels, only the ACTIVE channel is allowed to sweep. All other channels are **Blocked** - NOT allowed to sweep. Blocked channels will resume sweeping when they are made ACTIVE, or when the conflict is resolved.

Press **Trigger > Main > Restart** to cause ALL channels to sweep once. Then the active channel will resume sweeping continuously.

The Mechanical Devices dialog shows the settings of all of the switches and attenuators in the VNA. The settings for all active channels are shown side-by-side for easy comparison. This dialog allows you to determine the settings which would cause mechanical devices to switch between states on consecutive sweeps, potentially leading to device wear-out. It also allows you to determine if the conflict can be resolved to enable continuous sweeps on all channels.

The following are the mechanical devices that are potentially shown in the dialog. These components may not appear in your VNA model:

- Port 1 through Port 4 Bypass Switches
- Port 1 through Port 4 Source Attenuator settings
- Receiver A through Receiver R Attenuator settings
- Port 1 Noise Tuner Switch and Port 2 Noise Receiver Switch
How to access Mechanical Devices settings

Using **Hardkey/SoftTab/Softkey**

1. Press **Setup > Internal Hardware > Mechanical Devices...**

Using **a mouse**

1. Click **Instrument**
2. Select **Setup**
3. Select **Internal Hardware**
4. Select **Mechanical Devices...**

Remote commands

SCPI:  `SENSe<cnm>:SWEep:BLOCked?`

COM: `IsBlocked Property`

---

**Mechanical Devices** dialog box help

See Mechanical Devices Overview (scroll up)

The devices that appear in the table depend on the VNA model and options.

> **Yellow** highlighted cell indicates the Active channel.

**Blue** highlighted cells indicate the following:

- The channel is NOT able to sweep. **Blocked** is shown in the top row.
The highlighted device settings differ from that of the sweeping channels.

To modify entries in the table, click a cell.

When a selection is changed, the new setting is applied immediately.

If Port Power is coupled, a dialog prompts if coupling should be turned OFF.

**Limitations**

- **Measurement Class** can NOT be changed from this dialog.
- The dialog does NOT report device settings for multiport test sets.
- This dialog does NOT report device settings for external sources.

**Copy Active** Available ONLY when there is a conflict which causes at least one channel to be Blocked. When clicked, the mechanical device settings of the **Active** channel are copied to the Blocked channels. A warning message appears to remind you that power to one or more channels may be increased. **Exception:** When one or more Noise channels are present, then the settings of the two Noise switches are be determined by the lowest-numbered Noise channel if none are the active channel.

**Trigger** Launches the Trigger dialog box.

**OK** Closes the dialog box.

**Cancel** Does not apply changes that were made, and closes the dialog box.
Power Limit and Power Offset

- Overview
- How to access Power Limit and Power Offset settings

Other System Topics

Overview

Power Limit (Global scope)

Global power limit sets a maximum source power level for individual test ports. This value limits port power for all channels and all applications. Power levels that attempt to exceed the power limit is clipped at the limit.

Notes

- The power limit can NOT be set for power levels which are below the power level that is required by the analyzer to achieve phase lock - approximately -30 dBm.

- Because Fast Sweep mode allows power spiking, it is NOT allowed when a power limit is enabled.

- Components that are added to the RF path are accounted for by entering their loss (negative) or gain (positive) in the Power Offset section of the dialog box.

- VNA Applications may change RF path components. For example, IMD for Converters may change the combiner path and add an amplifier for LO input. Compensation is NOT made for these changes and port power may exceed the power limit or port power may be clipped unnecessarily.

- Power limiting does NOT clip power spikes that may occur during frequency band crossings.

- External test set ports are also included for power limiting.

Power Offset (Channel scope)

Power Offset provides a method of compensating port power for added attenuation or amplification in the source path. The result is that power at the specified port, all dialogs, and annotation, reflects the added components.
How to access the Offsets and Limits settings

Also accessed through the Preferences dialog.

Using **Hardkey/SoftTab/Softkey**

1. Press **Power > Leveling & Offsets > Offsets and Limits...**

Using a mouse

1. Click **Stimulus**
2. Select **Power**
3. Select **Offsets and Limits...**

**Offsets and Limits** dialog box help

Click a WHITE cell to change values. **Shaded cells** can NOT be changed.

Remote commands can be sent to lock and unlock the dialog box (UI) settings.

**Power Limit**

Limits the source power at each test port for ALL channels. Use this feature to protect DUTs that are sensitive to overpowered at the input. **Source Power** levels that exceed the **Limit** at the specified port are clipped at the limit, the power level in the **Source Power** column turns red, and an error message is displayed on the screen.

The Power Limit settings and **PLimit** status indicator survive Instrument Preset. When an Instrument State is recalled, the current Power Limit settings are applied to the recalled state.

To learn more, see **Power Limit Overview** (scroll up).

**State / Limit**

- **ON** - Power is limited to the adjacent value at the specified source port and displays **PLimit** in the status bar indicating that a power limit is ON.
• **OFF** - Power is NOT limited to this value, but to the maximum power of the source.

For VNA models with a second internal source, the **Port 1 Src2** Power Limit setting is NEVER available. Make the setting at the standard **Port 1**.

**Power Offset**

Power Offset provides a method of compensating port power for added attenuation or amplification in the source path. The result is that power at the specified port, all dialogs, and annotation reflects the added components.

- For amplification, use positive offset.
- For attenuation, use negative offset.

**Important Note:** Power Offset is added AUTOMATICALLY when a **Source Power Calibration**, **Guided Power Cal**, or **Power Compensation** is ON with Fixture Embed/Deembed. If you are NOT seeing the correct power level at your DUT, view the power Offset column in this dialog for unexpected offsets.

Optionally change the Source Power or Port Power values so that the following equation reflects your requirement:

\[
\text{Source Power} + \text{Power Offset} = \text{Port Power}
\]

**Source Cal  ON / OFF**

**Notes**

- Power Offset can be used with **Power Sweeps**. When a power sweep is enabled, the Start and Stop power levels are reported in this dialog.

- When port power offsets are used, port powers are automatically **uncoupled**. Port powers may not be coupled again until all port offsets are zero.

- **Cal All** does not automatically use the specified power offset during a calibration. To use a power offset for one or more ports when performing a **Cal All**, you must set the power offset value in the **Cal All** wizard.

**OK**  Closes the dialog box.
Receiver Temperature

This feature allows you to read the current temperature on the receiver microcircuit.

This feature is available on ALL models except the N523x models and the 4-port N5244A and N5245A models with DSP Version 4. Also, this feature is not available on the E5080A.

- To read temperature, press System > Service > Diagnostics > Receiver Temperature....
- The temperature reading is updated with every sweep.
- Temperature is available in Celsius and Fahrenheit.
- Temperature can also be read using remote commands.
  - SCPI: SENSE:TEMPerature?
  - COM: ReceiverTemperature
Setting System Impedance

The system impedance can be changed for measuring devices with an impedance other than 50 ohms, such as waveguide devices. The VNA mathematically transforms and displays the measurement data as though the VNA ports were the specified impedance value. Physically, the test ports are always about 50 ohms.

How to change the System Impedance

Using **Hardkey/SoftTab/Softkey**

1. Press **Scale > Constants > System Z0**

**System Z0 softtab help**

Allows you to change the system impedance (default setting is 50 ohms).

**Z0** Displays the current system impedance.

**For 75 ohm devices:**

1. Change the system Z0 to 75 ohms.
2. Connect minimum loss pads (75 ohm impedance) between the analyzer and the DUT to minimize the physical mismatch.
3. Perform a calibration with 75 ohm calibration standards.

**For waveguide devices**

When selecting a Cal Kit with an impedance other than 50 ohms (Waveguide = 1 ohm), it is **NO LONGER NECESSARY** to change the System Impedance setting before performing a calibration. The impedance for the calibration is now derived from the Cal Kit 'Connector' impedance setting.
The following links require an Internet connection.

**Note:** Check out the multimedia VNA Demo presentations, including 'Network Analyzer Basics'.

### Calibrations

- **AN1287-11** Specifying Calibration Standards and Kits for Keysight Vector Network Analyzers (5989-4840EN)
- **PN8510-8A** TRL Calibration for Non-Coaxial Measurements (5091-3645E)
- **Calibrating Standards for In-Fixture Device Characterization** (White Paper) (5989-3245EN)
- **Electronic vs. Mechanical Calibration kits: Calibration methods and accuracy** (White Paper) (5988-9477EN)
- **On-Wafer Calibration Using a 4-port, 20 GHz PNA-L Network Analyzer** (N5230A Option 240/245) (5989-2287EN)

### ECal

- **Keysight Electronic vs. Mechanical Calibration Kits: Calibration Methods and Accuracy** (5988-9477EN)
- **User Characterization: Electronic Calibration Feature Allows Users to Customize to Specific Needs** (5988-9478EN)

### Embedding / De-embedding

- **De-embedding and Embedding S-Parameter Networks Using a Vector Network Analyzer** (5980-2784EN)

### Amplifier Measurements

- **AN1408-7** Amplifier Linear and Gain Measurements (5988-8644EN)
- **AN1408-8** Amplifier Swept-Harmonic Measurements (5988-9473EN)
- **AN1408-9** Amplifier and CW Swept Intermodulation-Distortion Measurements (5988-9474EN)
- **AN1408-10** High-power measurements using the PNA (5989-1349EN)
- **AN1408-16** Power-Added Efficiency (PAE) 5989-7293EN
- **AN1408-17** Making Accurate IMD Measurements with the PNA-X Network Analyzer (5989-7265EN)
- **AN1408-19** High Power Amplifier Measurements Using NVNA

### Antenna Measurements

- **Triggering PNA Microwave Network Analyzers for Antenna Measurements** (5988-9518EN)
New Network Analyzer Methodologies in Antenna/RCS Measurements (5989-1937EN)

Pulsed Antenna Measurements Using PNA Network Analyzers (5989-0221EN)

Antenna and RCS Configurations (White Paper) (5989-0220EN)

Radar Measurements (Application Note) (5989-7575EN)

**Balanced Measurements** (Although the following refer to the ENA, they are also relevant to the PNA.)

On-wafer Balanced Component Measurement with the Cascade Microtech Probing System (5988-5886EN)

Network De-embedding/Embedding and Balanced Measurement (5988-4923EN)

Backplane Differential Channel Microprobe Characterization in Time and Frequency Domains (White Paper) (5989-3248EN)

**Mixer Measurements**

AN1408-1 Mixer Transmission Measurements Using the Frequency Conversion Application (5988-8642EN)

AN1408-2 Mixer Conversion-Loss and Group Delay Measurement Techniques and Comparisons (5988-9619EN)

AN1408-3 Improving Measurement and Calibration Accuracy Using the Frequency Converter Application (5988-9642EN)

AN1408-18 Measuring Group Delay of Frequency Converters with Embedded Local Oscillators (5989-7385EN)

Comparison of Mixer Characterization using New Vector Characterization Techniques (5988-7827EN)

Novel Method for Vector Mixer Characterization and Mixer Test System Vector Error Correction (5988-7826EN)

Measuring Absolute Group Delay of Multistage Converters Using PNA Microwave Network Analyzers (5989-0219EN)

**Pulsed Measurements**

AN1408-11 Accurate Pulsed Measurements (5989-0563EN)

AN1408-12 Pulsed-RF S-Parameter Measurements Using Wideband and Narrowband Detection

AN1408-21 Active-Device Characterization in Pulsed Operation Using the PNA-x (5990-7781EN)

Pulsed Antenna Measurements Using PNA Network Analyzers (5989-0221EN)

**Materials Measurements**

Basics of Measuring the Dielectric Properties of Materials (5989-2589EN)

Split Post Dielectric Resonators for Dielectric Measurements of Substrates (5989-5384EN)
Other Measurements

AN1287-12 Time Domain Analysis Using a Network Analyzer (5989-5723EN)
AN1408-14 Using the PNA Series to Analyze Lightwave Components (5989-3385EN)
AN1408-15 Using the PNA for Banded Millimeter-Wave Measurements (5989-4098EN)
AN1408-19 High Power Amplifier Measurements Using NVNA (5990-5039EN)
AN1408-20 High-Accuracy Noise Figure Measurements Using the PNA-X
MM-Wave Network Analyzers: Analysis of Cable Length on VNA System Performance (5989-1941EN)
Ultra-Low Impedance Measurements Using 2-Port Measurements (White Paper) (5989-5935EN)

Modeling

Utilizing TDR and VNA Data to Develop 4-port Frequency Dependent Models (White Paper) (5989-0638EN)
Advanced Measurements and Modeling of Differential Devices (White Paper) (5989-4518EN)

Automation

AN 1408-13 Introduction to Application Development using the PNA (5980-2666EN)
Connectivity Advances for Component Manufacturers (5980-2782EN)
The 'Need for Speed' in Component Manufacturing Test (5980-2783EN)
Network Analyzer Basics

This self-paced webinar discusses the basic concepts of Network Analysis.

From the Internet: Network Analyzer Basics.
Protection against electrostatic discharge (ESD) is essential while removing or connecting cables to the network analyzer. Static electricity can build up on your body and can easily damage sensitive internal circuit elements when discharged. Static discharges too small to be felt can cause permanent damage. To prevent damage to the instrument:

- **Always** have a grounded, conductive table mat in front of your test equipment.
- **Always** wear a grounded wrist strap, connected to a grounded conductive table mat, having a 1 MΩ resistor in series with it, when making test setup connections.
- **Always** wear a heel strap when working in an area with a conductive floor. If you are uncertain about the conductivity of your floor, wear a heel strap.
- **Always** ground yourself before you clean, inspect, or make a connection to a static-sensitive device or test port. You can, for example, grasp the grounded outer shell of the test port or cable connector briefly.
- **Always** ground the center conductor of a test cable before making a connection to the analyzer test port or other static-sensitive device. This can be done as follows:
  1. Connect a short (from your calibration kit) to one end of the cable to short the center conductor to the outer conductor.
  2. While wearing a grounded wrist strap, grasp the outer shell of the cable connector.
  3. Connect the other end of the cable to the test port and remove the short from the cable.

See [Analyzer Accessories](#) for ESD part numbers.
Absolute Output Power

An absolute output-power measurement displays absolute power versus frequency.

- What is Absolute Output Power?
- Why Measure Absolute Output Power?
- Accuracy Considerations
- How to Measure Absolute Output Power

See other Amplifier Parameters topics

What is Absolute Output Power?

An absolute-output power measurement displays the power present at the analyzer's input port. This power is absolute—it is not referenced (ratioed) to the incident or source power. In the log mag format, values associated with the grid's vertical axis are in units of dBm, which is the power measured in reference to 1 mW.

- 0 dBm = 1 mW
- -10 dBm = 100 μW
- +10 dBm = 10 mW

In the linear mag format, values associated with the grid's vertical axis are in units of watts (W).

Why Measure Absolute Output Power?

Absolute output power is measured when the amplifier's output must be quantified as absolute power rather than a ratioed relative power measurement. For example, during a gain compression measurement, it is typical to also measure absolute output power. This shows the absolute power out of the amplifier where 1-dB compression occurs.

Accuracy Considerations

The output power of the amplifier should be sufficiently attenuated if necessary. Too much output power could:
Damage the analyzer receiver

- Exceed the input compression level of the analyzer receiver, resulting in inaccurate measurements.

Attenuation of the amplifier's output power can be accomplished using either attenuators or couplers.

The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.

How to Measure Absolute Power

Do the following to measure absolute output power:

1. Preset the analyzer.

2. Select an unratioed power measurement (receiver B). Learn how.

3. Set the analyzer's source power to 0 dBm.

4. Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port-2.

5. Connect the amplifier as shown in the following graphic, and provide the dc bias.

6. Select the analyzer settings for your amplifier under test.

7. Remove the amplifier and connect the measurement ports together. Store the data to memory. Be sure to include the attenuator and cables in the test setup if they will be used when measuring the amplifier.

8. Save the instrument state to memory.

9. Reconnect the amplifier.

10. Select the data math function Data/Memory.
11. Scale the displayed measurement for optimum viewing and use a marker to measure the absolute output-power at a desired frequency.

12. Print or save the data to a disk.
Active Probing with the VNA

You can use passive and active, single and differential probes with the VNA.

RF Probes are available from Keysight, including the U1818A/B active probe with a maximum frequency of 12 GHz. Learn more at the Keysight website.

**Note: The VNA does NOT have a probe power port.**

Therefore, order the U1818A/B with banana plugs (Opt 002) for powering with an external power supply, such as the Keysight E3620A Dual Output Power Supply or equivalent.

See Also

- App Note 5990-4387EN: High Frequency Probing Solutions for Time and Frequency Domain Applications.
- U1818A/B Technical Overview

Procedure

Use the following general procedure to make VNA measurements with ONE U1818B active probe:

1. Connect the U1818A/B banana plugs to an external power supply. Power requirements for the U1818A/B are +15V (at 42mA) and -12.6V (at 12 mA).
2. Connect the U1818A/B to the VNA port 2 directly, or using an adapter or short cable.

3. Connect your DUT input connector to the VNA port 1 directly, or using an adapter or short cable.

4. On the VNA, press **Preset**.

5. Press **Meas > S-Param > S21** (transmission measurement).

6. Press **Freq**, then select the Start and Stop frequency range of the measurement. The maximum frequency of the U1818B is 12 GHz.

7. Connect the probe tips as close as possible to the DUT input connector (point A in the above image).

   **Note:** The probe tip has two pins:
   
   - One pin is connected to the signal trace.
   - The other pin is connected to the ground trace.

8. Press **Cal > Other Cals > Smart Cal...**

   - This calibration removes the losses (from the VNA test ports to the DUT) from subsequent measurements. Learn more about VNA calibration.
   
   - When correctly calibrated, the S21 measurement should show a flat response at 0 dB across the frequency range.
   
   - Connect the probe tips anywhere in the DUT path to view the frequency response between the DUT input and the probe tips.
   
   - To view the response in Time Domain, press **Math > Time Domain > Transform**. Learn about Time Domain measurements.
The AM-PM conversion of an amplifier is a measure of the amount of undesired deviation (PM) that is caused by amplitude variations (AM) inherent in the system.

- What Is AM-PM Conversion?
- Why Measure AM-PM Conversion
- Accuracy Considerations
- How to Measure AM-PM Conversion

What Is AM-PM Conversion?

AM-to-PM conversion measures the amount of undesired phase deviation (PM) that is caused by amplitude variations (AM) of the system. For example, unwanted phase deviation (PM) in a communications system can be caused by:

**Unintentional amplitude variations (AM)**

- Power supply ripple
- Thermal drift
- Multipath fading

**Intentional modulation of signal amplitude**

- QAM
- Burst modulation

AM-to-PM conversion is usually defined as the change in output phase for a 1-dB increment in the power-sweep applied to the amplifier's input (i.e. at the 1 dB gain compression point). It is expressed in degrees-per-dB (°/dB). An ideal amplifier would have no interaction between its phase response and the power level of the input signal.
Why Measure AM-PM Conversion

AM-to-PM conversion is a critical parameter in systems where phase (angular) modulation is used, such as:

- FM
- QPSK
- 16QAM

It is a critical parameter because undesired phase deviation (PM) causes analog signal degradation, or increased bit-error rates (BER) in digital communication systems. While it is easy to measure the BER of a digital communication system, this measurement alone does not help you understand the underlying causes of bit errors. AM-to-PM conversion is one of the fundamental contributors to BER, and therefore it is important to quantify this parameter in communication systems.

Refer to the I/Q diagram below for the following discussion on how AM-to-PM conversion can cause bit errors.
The desirable state change is from the small solid vector to the large solid vector.

With AM-to-PM conversion, the large vector may actually end up as shown with the dotted line. This is due to phase shift that results from a change in the input power level.

For a 64QAM signal as shown (only one quadrant is drawn), we see that the noise circles that surround each state would actually overlap, which means that statistically, some bit errors would occur.

**Accuracy Considerations**

With this method of measuring AM-to-PM conversion, the modulation frequency is approximately the inverse of the sweep time. Even with the fastest power sweep available on most network analyzers, the modulation frequency ends up being fairly low (typically less than 10 Hz). This could cause a slight temperature change as the sweep progresses, especially if the amplifier has low thermal mass, typical of an unpackaged device. Results using this method could differ slightly if the nonlinear behavior of an amplifier is extremely sensitive to thermal changes.

- The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.

- The output power of the amplifier should be sufficiently attenuated if necessary. Too much output power could:
  - damage the analyzer receiver
  - exceed the input compression level of the analyzer receiver, resulting in inaccurate measurements

- Attenuation of the amplifier's output power can be accomplished using:
  - Attenuators
  - Couplers
The frequency-response effects of the attenuators and couplers must be accounted for during calibration since they are part of the test system. Proper error-correction techniques can reduce these effects.

The frequency response is the dominant error in an AM-to-PM conversion measurement setup. Performing a thru-response measurement calibration significantly reduces this error. For greater accuracy, perform a 2-port measurement calibration.

How to Measure AM-PM Conversion

1. Preset the analyzer.

2. Select an S21 measurement in the power-sweep mode.

3. Enter the start and stop power levels for the analyzer's power sweep. The start power level should be in the linear region of the amplifier's response (typically 10-dB below the 1-dB compression point). The stop power should be in the compression region of the amplifier's response.

4. Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port 2.

5. Connect the amplifier as shown in the following graphic, and provide the dc bias.

6. Select the analyzer settings for your amplifier under test in order to perform a swept-power gain compression measurement at a chosen frequency. See Gain Compression.

7. Remove the amplifier and perform a measurement calibration. Be sure to include the attenuator and cables in the calibration setup if they will be used when measuring the amplifier.

8. Save the instrument state to memory.

9. Reconnect the amplifier.

10. Use a reference marker to target the amplifier's input power at the 1-dB gain compression point. Select a second marker and adjust its stimulus value until its response is 1-dB below the reference marker.
11. Change the S21 measurement from a log magnitude format to a phase format (no new calibration is required).

12. Find the phase change between the markers. The value is the AM-to-PM conversion coefficient at the 1-dB gain compression point.

13. Print the data or save it to a disk.
Amplifier Parameters Reference

- Gain
- Gain Flatness
- Reverse Isolation
- Gain Drift Versus Time
- Deviation from Linear Phase
- Group Delay
- Return Loss (SWR, ρ)
- Complex Impedance
- Gain Compression
- AM-to-PM Conversion

See Also

- High-Gain Amplifiers
- High Power with PNA-X

Gain

\[ t = \frac{V_{\text{trans}}}{V_{\text{inc}}} \]

Gain (dB) = -20\log_{10}|t|

Gain (dB) = P_{\text{out}} (dBm) - P_{\text{in}} (dBm)

The ratio of the amplifier's output power (delivered to a Z0 load) to the input power (delivered from a Z0 source). Z0 is the characteristic impedance, in this case, 50Ω.

For small signal levels, the output power of the amplifier is proportional to the input power. Small signal gain is the gain in this linear region.

As the input power level increases and the amplifier approaches saturation, the output power reaches a limit and the gain drops. Large signal gain is the gain in this nonlinear region. See Gain Compression.
Gain Flatness

The variation of the gain over the frequency range of the amplifier. See Small Signal Gain and Flatness.

Reverse Isolation

The measure of transmission from output to input. Similar to the gain measurement except the signal stimulus is applied to the output of the amplifier. See Reverse Isolation.

Gain Drift versus Time (temperature, bias)

The maximum variation of gain as a function of time, with all other parameters held constant. Gain drift is also observed with respect to other parameter changes such as temperature, humidity or bias voltage.

Deviation from Linear Phase

The amount of variation from a linear phase shift. Ideally, the phase shift through an amplifier is a linear function of frequency. See Deviation from Linear Phase.

Group Delay

\[ \tau_g(\infty) = -\frac{\Delta \phi}{\Delta \omega} \]

\[ = -\frac{1}{360} \times \frac{\Delta \phi}{\Delta f} \]

The measure of the transit time through the amplifier as a function of frequency. A perfectly linear phase shift would have a constant rate of change with respect to frequency, yielding a constant group delay. See Group Delay.

Return Loss (SWR, \( \rho \))

\[ \Gamma = \frac{V_{\text{refl}}}{V_{\text{inc}}} = \rho \angle \phi \]

Reflection coefficient = \( \rho \)

Return loss (dB) = \(-20 \log_{10} \rho \)

\[ \text{SWR} = \frac{1+\rho}{1-\rho} \]

The measure of the reflection mismatch at the input or output of the amplifier relative to the system \( Z_0 \).
characteristic impedance.

**Complex Impedance**

\[
Z = \frac{1 + \Gamma}{1 - \Gamma} \ast Z_0 \\
= -R + jX
\]

Complex impedance (1+G). The amount of reflected energy from an amplifier is directly related to its impedance. Complex impedance consists of both a resistive and a reactive component. It is derived from the characteristic impedance of the system and the reflection coefficient. See [Complex Impedance](#).

**Gain Compression**

See [Gain Compression Application](#).

**AM-to-PM Conversion Coefficient**

\[
\frac{\Delta \Phi}{\Delta P} = \frac{\Delta \theta}{\Delta P}
\]

The amount of phase change generated in the output signal of an amplifier as a result of an amplitude change of the input signal.

The AM-to-PM conversion coefficient is expressed in units of degrees/dB at a given power level (usually P1dB, which is the 1 dB gain compression point). See [AM-PM Conversion](#).
Antenna Measurements

This topic describes how to setup a Keysight Vector Network Analyzer (VNA) to make S21 measurements on an array of antennas. Measurements can be made on up to 100 antenna arrays (Ports) and up to 15 discrete frequencies.

Measurement Sequence

1. The VNA is set to a start frequency.
2. As the antenna moves, the VNA responds to each external trigger signal by measuring an antenna port.
3. When all ports are measured, the VNA increments to the next frequency.
4. Again the VNA measures all ports, and so forth until all ports are measured at all frequencies in the forward direction.
5. As the antenna begins moving in the opposite direction, the same sequence occurs, except the VNA decrements in frequency until all ports are measured at all frequencies and the VNA is set back to the original start frequency.

Once setup, only external trigger signals are sent to the VNA. After each trigger, measurement data is stored in internal VNA memory.

How to set up the VNA

1. Press Preset
2. Press Trigger > Main > Trigger Source > External
3. Press Trigger > Main > Trigger
4. In the Trigger dialog under Trigger Scope, select Channel
5. Click OK

Forward Sweep

1. Press Trace > Trace N > Trace N to add a new trace.
2. Press Trace > Trace Setup > Measure....
3. Select S21 then Channel Number 1
4. Press **Trigger > Main > Trigger**

5. In the Trigger dialog under **Channel Trigger State**, set the Trigger Mode to **Point**

6. Click **OK**

7. Press **Sweep > Main > Sweep Type > Segment Sweep**

8. Click **OK**

9. Press **Sweep > Segment Table > Insert Segment**

10. Do this 15 times

11. For each Segment in the Segment table:

   1. Click **State:** and select **ON**
   2. Click both **START** and **STOP** Frequency: (each new segment ascends in frequency)
   3. Click **Points:** type Number of Ports (elements)

**Reverse sweep**

Repeat the following steps for each frequency: (up to 15)

- Increment the channel number (**X**) Starting with Channel 2
- Decrement the frequency (**F**)

1. Press **Trace > Trace N > Trace N** to add a new trace.

2. Press **Trace > Trace Setup > Measure....**

3. Click **S21** then Channel Number **X**

4. When a window contains four traces, press **Trace > Trace Setup > Add Trace > New Trace + Window.**

5. Click **OK**

6. Press **Trigger > Main > Trigger**

7. In the Trigger dialog under **Channel Trigger State**, set the Trigger Mode to **Point**

8. Click **OK**

9. Press **Sweep > Main > Sweep Type > Segment Sweep**
10. Click **OK**

11. Press **Sweep > Segment Table**

12. In the Segment table

   1. Click **State**: and select **ON**
   
   2. Click both **START** and **STOP** Frequency $F$
   
   3. Click **Points**: type Number of Ports (elements)
Balanced Measurements

- What are Balanced Devices?
- Differential and Common Modes Model
- Measuring Mixed Mode (Balanced) S-Parameters
- Measuring Imbalance Parameters
- Measuring CMRR
- Port Mapping
- Calibrating Balanced Measurements
- How the analyzer makes Balanced Measurements

Other Measurement Setup Topics

Check out the Integrated True Mode Stimulus Application (iTMSA).

What are Balanced Devices?

Standard Single-ended devices generally have one input port and one output port. Signals on the input and output ports are referenced to ground.

Balanced devices have two pins on either the input, the output, or both. The signal of interest is the difference and average of the two input or output lines, not referenced to ground.
Differential and Common Modes Model

On balanced devices, the signal of interest is the **difference** and **average** of the two input or output lines. In balanced device terminology, these signals are known as the Differential and Common modes.

The following model shows how two signals (A and B) combine to create Differential and Common mode signals:

- **Signal A** is fixed at 1V peak
- **Signal B** is selectable
- **Differential** is calculated as **A minus B**
- **Common** is calculated as the **AVERAGE of A and B**

**Note:** Click **Signal B** selections to see various Differential and Common signals.

<table>
<thead>
<tr>
<th>Signal A</th>
<th>Calculations</th>
<th>Differential (A - B)</th>
<th>Common (Avg) (A + B) / 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1V</td>
<td>1 - 0 = 1</td>
<td>(1 + 0)/2 = .5</td>
<td></td>
</tr>
<tr>
<td>180° Out of Phase</td>
<td>1V</td>
<td>1 - (-1) = 2</td>
<td>(1 + (-1))/2 = 0</td>
</tr>
<tr>
<td>180° Out of Phase</td>
<td>2V</td>
<td>1 - (-2) = 3</td>
<td>(1 + (-2))/2 = -.5</td>
</tr>
<tr>
<td>In Phase</td>
<td>1V</td>
<td>1 - 1 = 0</td>
<td>(1 + 1)/2 = 1</td>
</tr>
<tr>
<td>In Phase</td>
<td>2V</td>
<td>1 - 2 = -1</td>
<td>(1 + 2)/2 = 1.5</td>
</tr>
</tbody>
</table>

**Notes:**

- Even when Signal B is 0V, like a Single-ended signal, there is still a unique Differential and Common mode representation of the two individual signals.
- The above model does not show a DUT. The difference and average of two signals can be calculated for
both the balanced INPUT and balanced OUTPUT of a device.

**Measuring Mixed Mode (Balanced) S-Parameters**

Mixed mode S-parameters combine traditional S-parameter notation with balanced measurement terminology.

Some balanced devices are designed to amplify the differential component and reject the common component. This allows noise that is common to both inputs to be virtually eliminated from the output. For example, a balanced device may amplify the differential signal by a factor of 5, and attenuate the common signal by a factor of 5. Using traditional S-parameter notation, an S21 is a ratio measurement of the device **Output** / device **Input**. Mixing this with balanced terminology, we could view the amplifier's Differential Output signal / Differential Input signal. To see this parameter on the analyzer, we would select an Sdd21 measurement using the following balanced notation:

\[ \text{Sabxy} \]

Where

- **a** - device output mode
- **b** - device input mode

(choose from the following for both a and b:)

- **d** - differential
- **c** - common
- **s** - single ended

- **x** - device output "logical" port number
- **y** - device input "logical" port number

**See Also**

- Logical port mapping
- Port mapping with External Test Sets
- iTMSA

**Measuring Imbalance Parameters**

Imbalance is a measure of how well two physical ports that make up a balanced port are matched. With a perfectly balanced port, the same amount of energy flows to both ports and the magnitude of the ratio...
of these ports is 1.

The notation is similar to traditional S-parameters. In the following diagrams, the letters a, b, c, and d are used because any analyzer port can be assigned to any logical port using the port mapping process.

For example, in the following single-ended - balanced formula, Sba indicates the device output port is logical port b and the input port is logical port a.
Imbalance parameter when measuring a balanced - single-ended device.

\[ \text{Imbal} = \frac{\text{Scb}}{\text{Sca}} \]

Imbalance1 and Imbalance2 parameters when measuring a balanced - balanced device.

\[ \text{Imbal}_1 = \frac{\text{Sac} - \text{Sad}}{\text{Sbc} - \text{Sbd}} \]
\[ \text{Imbal}_2 = \frac{\text{Sca} - \text{Scb}}{\text{Sda} - \text{Sdb}} \]
Imbalance1 and Imbalance2 parameters when measuring a **single-ended - single-ended - balanced** device.

![Diagram of DUT ports and imbalance equations]

**Measuring CMRR (Common Mode Rejection Ratio)**

CMRR is a ratio of the transmission characteristic in differential mode over the transmission characteristic in the common mode of the balanced port as the measurement parameter. A high value indicates more rejection of common mode, which is desirable in a device that transmits information in the differential portion of the signal. The table below shows the CMRR parameter you can select when measuring each balanced device.

<table>
<thead>
<tr>
<th>Device Type</th>
<th>Port A</th>
<th>Port B</th>
<th>Port C</th>
<th>Port D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-ended - balanced device</td>
<td>Sds21</td>
<td>Scs21</td>
<td>Ssd12</td>
<td>Ssc12</td>
</tr>
<tr>
<td>Balanced - single-ended device</td>
<td>Ssd21</td>
<td>Ssc21</td>
<td>Sds12</td>
<td>Ssc12</td>
</tr>
<tr>
<td>Balanced - balanced device</td>
<td>Sdd21</td>
<td>Scc21</td>
<td>Sds12</td>
<td>Ssc12</td>
</tr>
<tr>
<td>Single-ended - single-ended - balanced device</td>
<td>Sds31</td>
<td>Scs31</td>
<td>Sds32</td>
<td>Scs32</td>
</tr>
</tbody>
</table>

**Device Topology and Port Mapping**

As we have seen on balanced inputs and outputs, the signal of interest is the difference or average of two BALANCED input or BALANCED output lines. It is also possible to have single-ended ports
AND balanced ports on the same device. The two balanced input or output lines are referred to as a single "logical" port.

When configuring a balanced measurement on the analyzer, select a device 'topology'. Then map each test port to the DUT ports. The analyzer assigns "logical ports". See how to set device topology.

The following device topologies can be measured by a 4-port analyzer.

- **Balanced / Balanced**
  (2 logical ports - 4 physical ports)

- **Single-ended / Balanced**
  (2 logical ports - 3 physical ports)
These topologies can be used in the reverse (⇐⇒) direction to measure:

- **Balanced / Single-ended** topology
- **Balanced / Single-ended - Single-ended** topology

For example, to measure a **Balanced / Single-ended** topology, measure the S12 (reverse direction) of a **Single-ended / Balanced** topology.

**Calibrating Balanced Measurements**

Balanced measurements are calibrated in the same manner as single-ended (standard) measurements. However, for highest accuracy, you must choose Thru paths so that each transmission path of the balanced measurement is represented. For a Balanced/Balanced topology, this means that FOUR Thru connections should be made.

For example (see following image):

---

![Diagram of Balanced / Single-ended topology](image)

![Diagram of Single-ended - Single-ended / Balanced topology](image)
- Balanced Port 1 is ports 1 and 3
- Balanced Port 2 is ports 2 and 4
- Thru paths to be calibrated should be: 12, 14, 32, 34.
- Paths 13, and 24 are less important.

To select Thru paths:

1. From SmartCal, on the Select DUT Connectors and Cal Kits page, check **Modify Cal**.
2. Click **Next** to see the following Cal Wizard page:

![Modify Cal](image)

**How the analyzer makes Balanced Measurements**

When using standard Balanced measurements, the analyzer does not provide true balanced measurements by stimulating both balanced inputs together and measuring both outputs relative to one another. Instead, the analyzer makes only Single-ended measurements. On a Balanced/ Balanced device, it stimulates each input and measures each output individually. From the output data, the analyzer calculates the Differential and Common outputs from the DUT using the same math formulas as the above model. However, all measurements and calculations are performed in frequency domain using complex (magnitude and phase) data. The Balanced S-parameter display data is then calculated from the Differential and Common inputs and outputs.

In iTMSA, the VNA **DOES** stimulate both balanced inputs with true balanced sources.
Complex Impedance

When making an S₁₁ or S₂₂ measurement of your device under test, you can view complex-impedance data such as series resistance and reactance as well as phase and magnitude information. Complex impedance data can be viewed using either the Smith Chart format or the Polar format.

- What Is Complex Impedance?
- Accuracy Considerations
- How to Measure Complex Impedance

What Is Complex Impedance?

Complex-impedance data is information that can be determined from an S₁₁ or S₂₂ measurement of your device under test, such as:

- Resistance
- Reactance
- Phase
- Magnitude

The amount of power reflected from a device is directly related to the impedances of both the device and the measuring system. For example, the value of the complex reflection coefficient (Γ) is equal to 0 only when the device impedance and the system impedance are exactly the same (i.e. maximum power is transferred from the source to the load). Every value for Γ corresponds uniquely to a complex device impedance (as a function of frequency), according to the equation:

\[ Z_L = \frac{(1 + \Gamma)}{(1 - \Gamma)} \times Z_0 \]

where \( Z_L \) is your test device impedance and \( Z_0 \) is the measuring system's characteristic impedance.

Complex Impedance is best viewed using either Polar or Smith Chart format.

Accuracy Considerations

- The Smith chart is most easily understood when used with a full scale value of 1.0.
- For greater accuracy when using markers in the Smith chart or polar formats, activate the discrete marker mode.

- The uncertainty of reflection measurements is affected by:
  - Directivity
  - Reflection tracking
  - Source match
  - Load match (with 2-port devices)

With a 2-port calibration, the effects of these factors are reduced. A 1-port calibration provides the same accuracy if the output of the device is well terminated. Refer to the graphic below for the following discussion.

If you connect the device between both analyzer ports, it is recommended that you use a 10 dB pad on the output of the device to improve measurement accuracy. This is not necessary if you use a 2-port calibration since it corrects for load match.

If you connect a two-port device to only one analyzer port, it is recommended that you use a high-quality load (such as a calibration standard) on the output of the device.

How to Measure Complex Impedance

1. Connect the device as shown in the previous graphic.
2. Preset the analyzer.
3. Set up, calibrate, and perform an S11 or S22 measurement.
4. View impedance data:
   a. Select the Smith Chart format.
b. Scale the displayed measurement for optimum viewing.

c. Position the marker to read the resistive and reactive components of the complex impedance at any point along the trace.
d. Print the data or save it to a disk.

5. View the magnitude and phase of the reflection coefficient:

   a. Select the Smith chart format or the Polar format.
   
   b. Select either Lin Marker or Log Marker formats.
   
   c. Scale the displayed measurement for optimum viewing.
   
   d. Position the marker to read the frequency, magnitude, and phase of the reflection coefficient ($\Gamma$) at any point along the trace.
   
   e. Print the data or save it to a disk.
The VNA has three Delay functions which are similar but are used in different ways.

1. **Group Delay format** is used to display the Group Delay of a network. Group Delay is defined as:

   \[-d(\phi)/d(\omega)\]  
   -- where \(\phi\) is radian angle, and \(\omega\) is radian frequency.

   Since it is defined by a derivative, the value must be determined from an analytic function. However, the VNA makes discrete measurements, so we approximate the group delay by taking the finite difference:

   \[-(1/360) \cdot \delta(\phi)/\delta(f)\]  
   -- where \(\phi\) is degree angle and \(f\) is frequency in Hz. The 1/360 does the proper conversion of degrees to radians and Hz frequency to radian frequency.

   From this we can see that, if the phase response of a network varies with frequency, then the Group Delay must vary as well. In fact, many filters are specified by the variation of their Group Delay.

   If we measure the phase response of a lossless cable, it should be a straight line. But, of course, nothing is perfect. The phase response will have a small amount of noise. This is due to trace noise of the VNA, and the loss with real cables or transmission lines, which causes a small amount of non-linear phase change with frequency. So, if we look at the Group Delay of a cable, we will see a small amount of variation. Also, if the frequency spacing is small enough when you make the measurement, the \(\delta(f)\) in the denominator becomes very small, so the delay can have wide swings with just a little noise.

   To overcome this issue, we sometimes add smoothing to a phase trace, which widens the effective \(\delta(f)\), called the aperture, and provides a less noisy Group Delay response. The Group Delay of a device is only valid for a given frequency aperture. [Learn more about Group Delay.](#)

2. **Electrical Delay** function. On many filters, the passband response is specified for a maximum value of "Deviation from Linear Phase". When looking at the passband of a multi-pole filter, one sees the phase changing very rapidly. This makes it difficult to determine the linearity of the phase response. The Electrical Delay function subtracts out a "LINEAR PHASE" equivalent to the delay time value computed as above. When you use this function, you dial in the Linear Delay such that a CONSTANT PHASE SLOPE is removed from the phase trace, until the phase trace is mostly flat. The remaining variation is the deviation from linear phase.

   To make this task a little less tedious, the VNA has a marker function called Marker =>> Delay. This function computes the Group Delay value at the marker position, using a 20% smoothing aperture, then changes the Electrical Delay value to this value. Obviously, if the phase trace is not perfectly linear, moving the marker and recomputing the delay will result in different values. The phase slope added by
the electrical delay function applies only to the current measurement. That is, each measurement (S11, S22, S12, S21) can have its own value of electrical delay. Learn more about Deviation from Linear Phase.

3. **Port Extension** is a function that is similar to calibration. It applies to all the traces in a given channel. It compensates for the phase response change that occurs when the calibration reference plane is not the same as the measurement plane of the device.

Let's look at an example of a DUT that is mounted on a PCB fixture with SMA connectors. We can easily calibrate at the SMA connectors. But if we add the fixture to measure the board-mounted device, the apparent phase of the DUT is changed by the phase of the PCB fixture. We use port extensions to add a LINEAR PHASE (constant delay) to the calibration routines to shift the phase reference plane to that of the DUT. This is ONLY valid if the fixture consists of a transmission line with linear phase response, and this limitation is usually met in practice. The main reason that it is NOT met is that there is mismatch at the SMA-to-PCB interface. This mismatch was not removed with the error correction because it occurs AFTER the SMA connector. Ripple can be seen on the display as signals bounce back and forth between the mismatch and the DUT. If the DUT is well matched, the ripple effect is very small. However, when we use Automatic Port Extension (APE), and we leave the fixture open (the DUT removed), the reflection is large and we see larger ripples. That is why APE uses a curve fitting process to remove the effects of the ripple. For best effect, the wider the IF Bandwidth, the better we can "smooth-out" the ripples with curve fitting. Still, we are fitting a LINEAR PHASE SLOPE to the phase response, and thus we use only a single Port Extension Delay value to represent the phase slope.

The method used by older VNAs to get this same functionality was to add a mechanical line stretcher to the reference channel, which removed a fixed delay amount from the port. Port extensions give 1x the delay for transmission at each port, and 2x the delay for reflection, so it differs somewhat from Electrical Delay above, in that the math function depends upon the measurement being made. The signal passes twice through the fixture for reflection (out and back), but only once for each port on transmission. For S21, the phase slope added is the sum of the port 1 and port 2 Port Extension Delay values.

The "User Range" APE function is used in cases where a fixture has limited bandwidth, perhaps due to tuning elements or bias elements. In this case, the model of constant delay for the fixture over the whole bandwidth is not valid, so a narrower "User Range" of frequencies can be selected to compute the delay. Since the aperture is smaller, there is more uncertainty in the delay computation for port extension. Also, for those who had been using the Marker => Delay function to estimate the delay, we added the "Active Marker" selection to APE, which works exactly the same as Marker->Delay. Learn more about Automatic Port Extensions.
Deviation from Linear Phase

Deviation from linear phase is a measure of phase distortion. The electrical delay feature of the analyzer is used to remove the linear portion of the phase shift from the measurement. This results in a high-resolution display of the non-linear portion of the phase shift (deviation from linear phase).

- What Is Linear Phase Shift?
- What Is Deviation from Linear Phase?
- Why Measure Deviation from Linear Phase?
- Using Electrical Delay
- Accuracy Considerations

See also Comparing the Analyzer Delay Functions

See other Tutorials

What Is Linear Phase Shift?

Phase shift occurs because the wavelengths that occupy the electrical length of the device get shorter as the frequency of the incident signal increases. Linear phase-shift occurs when the phase response of a device is linearly proportional to frequency. Displayed on the analyzer, the phase-versus-frequency measurement trace of this ideal linear phase shift is a straight line. The slope is proportional to the electrical length of the device. Linear phase shift is necessary (along with a flat magnitude response) for distortionless transmission of signals.

What Is Deviation from Linear Phase?

In actual practice, many electrical or electronic devices will delay some frequencies more than others, creating non-linear phase-shift (distortion in signals consisting of multiple-frequency components). Measuring deviation from linear phase is a way to quantify this non-linear phase shift.

Since it is only the deviation from linear phase which causes phase distortion, it is desirable to remove the linear portion of the phase response from the measurement. This can be accomplished by using the electrical delay feature of the analyzer to mathematically cancel the electrical length of the device under test. What remains is the deviation from linear phase, or phase distortion.
Why Measure Deviation from Linear Phase?

The deviation from linear phase measurement accomplishes the following:

- Presents data in units of phase rather than units of seconds (group delay). For devices that pass modulated signals, units of phase may be most practical.
- Provides a less noisy measurement than a group delay measurement.

Using Electrical Delay

The electrical delay feature is the electronic version of the mechanical "line stretcher" of earlier analyzers. This feature does the following:

- Simulates a variable-length lossless transmission line, which is effectively added to or removed from the reference signal path.
- Compensates for the electrical length of the device under test.
- Flattens the measurement trace on the analyzer's display. This allows the trace to be viewed at high resolution in order to see the details of the phase nonlinearity.
- Provides a convenient method to view the deviation from linear phase of the device under test. See the following graphic.

Learn how to set Electrical Delay.

Accuracy Considerations

The frequency response of the test setup is the dominant error in a deviation from linear phase measurement. To reduce this error, perform a 2-port measurement calibration.

How to Measure Deviation from Linear Phase:

1. Preset the analyzer.
2. If your device under test is an amplifier, it may be necessary to adjust the analyzer's source power:

   - Set the analyzer's source power to be in the linear region of the amplifier's output response (typically 10-dB below the 1-dB compression point).
   - Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port 2.

3. Connect the device under test as shown in the following graphic.

4. Select an S21 measurement.

5. Select the settings for your device under test, including the following:

   - **Format**: phase
   - **Scale**: autoscale

6. Remove the device and perform a calibration.

7. Reconnect the device.

8. Scale the displayed measurement for optimum viewing.

9. Create a marker in the middle of the trace.

10. Press Marker > Marker -> Functions > Marker -> Delay to invoke the Marker to Electrical Delay function. This flattens the phase trace.

11. If desired, on the **Scale** menu, click Electrical Delay to fine-tune the flatness of the phase trace.

12. Use the markers to measure the maximum peak-to-peak deviation from linear phase.

13. Print the data or save it to a disk.
20 GHz Coupler Directivity Measurement Example

The purpose of this example is to show how to use the PNA Equation Editor to create a directivity measurement trace, in order to measure a directional coupler’s directivity easily. Two different methods, using measurement trace data formatted differently are explained below. Results are displayed on the bottom left and bottom right hand side windows of the PNA screen. It’s important to note that either approach produces the same end result, as shown with markers on both equation editor traces. Measurement trace selection and formatting are critical for making this work correctly. A user would only need to use one method or the other, but both ways are explained.

1. Set up the connections between the PNA and a directional coupler as shown below.

![Diagram of PNA and directional coupler connections]

   a. Connect PNA Port 1 to coupler's input port.
   b. Connect PNA Port 2 to coupler's output port.
   c. Connect PNA Port 3 to coupler's coupled port.

2. Set up basic stimulus conditions for directional coupler's measurement.

   a. Preset the PNA.
   b. Set start frequency to 1 GHz.
   c. Set stop frequency to 20 GHz.
   d. Set power level to the default value.
   e. Set IFBW to 30 Hz.
3. **Make window setup, trace measurement, and formatting selections.**

Establish four measurement windows on the display screen. Set up windows and traces such that the left-hand side will show traces 1, 2, and 3 in upper left window, as well as trace 7 in lower-left window. Corresponding to the DUT connection diagram, traces 1, 2, and 3 will be S21, S31, and S32 measurements, respectively; and all must be formatted as real. Trace 7, an equation editor trace, will be log magnitude formatted. Trace 7 will be a S11 trace when it is created, but it will be modified to be a directivity equation trace later.

For the right-hand side of the display screen, create traces 4, 5, and 6 in upper-right window, as well as trace 8 in lower right window. Traces 4, 5, and 6, will also be S21, S31, and S32 measurements, respectively, but these traces are formatted in linear magnitude. Trace 8, an equation editor trace, will need to be log magnitude formatted, just like the lower-left window for equation Trace 7. Trace 8 will be a S11 trace when it is created, but it will be modified to be a directivity equation trace later, too.

4. **Create the directivity traces for Tr 7 and Tr 8 with the equation editor.**

Tr 7 will use real formatted data from Tr 1, Tr2, and Tr3 in the equation.

   a. Select Tr 7.
   b. From the Marker/Analysis menu, select Analysis, then Equation Editor....
   c. Enter the Equation as Dir_from_REAL_Data=S32/(S31*S21).
   d. Check the Enable Equation box.
   e. Click on OK.

Tr 8 will use linear magnitude formatted data from Tr 4, Tr 5, and Tr 6 in the equation.

   a. Select Tr 8.
   b. From the Marker/Analysis menu, select Analysis, then Equation Editor....
   c. Enter the Equation as Dir_from_LinMag_Data=Tr6/(Tr5*Tr4).
   d. Check the Enable Equation box.
   e. Click on OK.

**Note:** Window titles were added to show user comments about the traces in each window. This is strictly optional, but if desired, window titles can be added by doing the following:

   a. Select the Response menu, then choose Display, Labels, then Window Title....
   b. In the Window Title dialog box, enter a meaningful title.
   c. Check the Enable box.
   d. Click on OK.
Note: To demonstrate the equivalent results of both equation editor methods for directivity measurements, marker tables were set up for Tr 7 and Tr 8 in windows 3 and 4, respectively. Again, this is optional, but recommended.

5. Perform a 3-port calibration.

Note: An N4433A-010 ECal module was used in this example.

6. Display directivity measurement results.

a. Initiate a single sweep.
b. Observe that Tr 7 and Tr 8 are the same.

Discussion

In linear terms, the textbook equation for calculating a coupler's directivity is:

Directivity = Isolation - (Coupling + Loss)

Using log rules, this equation becomes:
Directivity = Isolation / (Coupling * Loss)

The PNA equation editor has all the underlying data structures in complex real/imag (linear) values. All the math in the textbook is also in linear terms, so the equation needs to be Dir_from_LinMag_Data=Tr6/(Tr5*Tr4). That equation was implemented in Tr 8.

The equation editor also returns the complex data structure in real/imag, so to get directivity in dB, set the format to log magnitude for Tr 7 - just like Tr 8. (And the equation is complex, so phase will be preserved for directivity too, so you can think of this as directivity_re_im.)

For Tr 8 (in dB), the correct directivity is for isolation – (coupling + loss). But since you cannot form this equation in equation editor, you have to use the complex form. So you can refer to Tr 6, but the reference is to the last complex result, not to the formatted trace.

This equation, Directivity_LogM=20*log(mag(S32/(S31*S21)), while correct (and not used in this example), will result in the complex value being formed in dB (you are formatting in the equation editor), so you would need to set the format to real to see the dB value, oddly enough. So, it is best to use Dir=S32/(S31*S21) and set the format as desired (e.g. Logmag) to get directivity in dB.
Filter Measurements

The main filter measurements discussed in this topic include:

- **Measurements**
  - Bandwidth
  - Center Frequency
  - Q Factor
  - Filter Loss (Insertion Loss)
  - Retrieving Filter Statistics
- **Ripple**
  - Retrieving Peak to Peak Ripple
- **Smoothing**
  - Performing a Smoothing Function
- **Related SCPI and COM Remote Commands**

![Diagram of bandwidth measurement](image)

**Bandwidth**

Marker search functions automatically determine the bandwidth of a band-pass filter by placing a marker on the maximum level (marker 1), a marker on the lower frequency (marker 2), and a marker on the higher frequency (marker 3). The higher and lower frequencies are determined by specifying a target bandwidth level. The default target bandwidth level is -3 dB which means that the bandwidth of a band-pass filter is determined by measuring the higher and lower frequencies at -3 dB below the peak.

To find the bandwidth of a notch filter, a positive value would be entered for the target bandwidth level (for example, 3 dB). For notch filters, the marker search functions place a marker on the minimum
level (marker 1), a marker on the lower frequency (marker 2), and a marker on the higher frequency (marker 3).

Center Frequency

In addition to being used to determine the bandwidth, markers 2 and 3 are used to determine the center frequency of a band-pass or notch filter by calculating the mathematical midpoint between them as shown in the diagram above.

Q Factor

After determining the bandwidth and center frequency using the marker search functions, the VNA also calculates the Q factor as the ratio of Center Frequency to Bandwidth (Center Frequency / Bandwidth).

The VNA can measure very high Q factors limited only by the IF bandwidth and the number of sampling points. You must ensure that a sufficient number of sampling points are utilized for accurate results.

Filter Loss (Insertion Loss)

The filter loss is the loss caused by the insertion of the filter into the transmission line. The loss is defined as follows:

\[ 20 \log_{10} |S_{21}| \]

Marker search functions retrieve the filter loss as the Y-axis value of marker 4. This is the loss of the filter at its center frequency.

Retrieving Filter Statistics

The following procedure retrieves filter bandwidth, center frequency, loss, and Q factor.

1. Press Search > Bandwidth & Notch > BW Level.

2. Enter the target bandwidth level (default is -3 dB).

Note: To find the bandwidth of a notch filter, enter a positive value for the target bandwidth level.
4. Press **Search > Bandwidth & Notch > Bandwidth Search** to retrieve filter statistics.

Ripple

Ripple refers to the amplitude deviation of the band-pass filter, which is also referred to as filter flatness. The VNA's trace statistics functions are used to find the peak to peak ripple in the band-pass as shown in **Retrieving Peak to Peak Ripple**.

![Ripple diagram](image)

**Retrieving Peak to Peak Ripple**

The following procedure sets up and retrieves the ripple of the filter.

1. Press **Math > Analysis > Statistics....**
2. In the **Trace Statistics** dialog, select **Statistics - Mean, Standard Deviation, Peak to Peak**.
3. In the pull-down menu, select a **User** number to assign a range.
4. Enter the **Start** and **Stop** frequencies for the ripple measurement. This should be the bandwidth of the band-pass filter.
5. Click **OK** to retrieve the peak to peak ripple of the filter.

Smoothing
The VNA has a smoothing function that applies an average smoothing on a trace. The smoothing function is defined as follows:

\[ Y_{n_{_\text{Smooth}}} = \frac{[Y(n-m) + Y(n-m+1) + \cdots + Y(n) + \cdots Y(n+m-1) + Y(n+m)]}{2m+1} \]

where \(2m + 1\) is called the smoothing aperture measured in points.

The smoothing function is also provided as a percentage of the span and then converted to smoothing points as follows:

\[ m = \text{int} \left[ \frac{N \cdot (\text{smoothing percent})}{2} \right] \]

When applied, each point becomes an average of the surrounding points.

**Performing a Smoothing Function**

1. Press `Avg BW > Smoothing > Smoothing ON`.
2. If entering a percentage of the span, press `Avg BW > Smoothing > Smooth Percent` which will change the number of Smoothing Points appropriately.
3. If entering smoothing points, press `Avg BW > Smoothing > Smooth Points` which will change the Smoothing Percent Span appropriately.

**Related SCPI and COM Remote Commands**
<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Determines the filter bandwidth -3 dB (default) below the band-pass filter peak</td>
<td>CALCulate:MEASure:MARKer:BWIDth:THReshold</td>
<td>BandwidthTarget</td>
</tr>
<tr>
<td>Searches measurement data using current Bandwidth Target to determine the bandwidth</td>
<td>CALCulate:MEASure:MARKer:BWIDth[:STATe]</td>
<td>SearchFilterBandwidth</td>
</tr>
<tr>
<td>Retrieves the filter bandwidth</td>
<td>CALCulate:MEASure:MARKer:BWIDth:DATA?</td>
<td>FilterBW</td>
</tr>
<tr>
<td>Retrieves the filter center frequency</td>
<td>CALCulate:MEASure:MARKer:BWIDth:DATA?</td>
<td>FilterCF</td>
</tr>
<tr>
<td>Retrieves the filter Q factor</td>
<td>CALCulate:MEASure:MARKer:BWIDth:DATA?</td>
<td>FilterQ</td>
</tr>
<tr>
<td>Retrieves the filter loss</td>
<td>CALCulate:MEASure:MARKer:BWIDth:DATA?</td>
<td>FilterLoss</td>
</tr>
<tr>
<td>Continually tracks the filter bandwidth</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TRACking</td>
<td>BandwidthTracking</td>
</tr>
<tr>
<td>Sets the start frequency for the measurement</td>
<td>CALCulate:MEASure:FUNCtion:DOMain:USER:STARt</td>
<td>UserRangeMin</td>
</tr>
<tr>
<td>Sets the stop frequency for the measurement</td>
<td>CALCulate:MEASure:FUNCtion:DOMain:USER:STOP</td>
<td>UserRangeMax</td>
</tr>
<tr>
<td>Sets the range used to calculate trace statistics</td>
<td>CALCulate:MEASure:FUNCtion:DOMain:USER[:RANGe]</td>
<td>StatisticsRange</td>
</tr>
<tr>
<td>Sets the statistic type (peak to peak)</td>
<td>CALCulate:MEASure:FUNCtion:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Retrieves the peak to peak value</td>
<td>CALCulate:MEASure:FUNCtion:DATA?</td>
<td>PeakToPeak</td>
</tr>
<tr>
<td>Turns data smoothing on/off</td>
<td>CALCulate:MEASure:SMOothing[:STATe]</td>
<td>Smoothing</td>
</tr>
<tr>
<td>Sets smoothing as a percentage of span</td>
<td>CALCulate:MEASure:SMOothing:APERture</td>
<td>SmoothingAperture</td>
</tr>
<tr>
<td>Sets the number of adjacent data points to average</td>
<td>CALCulate:MEASureSMOothing:POINts</td>
<td>None</td>
</tr>
</tbody>
</table>
Small Signal Gain and Flatness

Small signal gain is the gain in the amplifier's linear region of operation. This is typically measured at a constant input power over a swept frequency. Gain flatness is the measure of the variation of gain over a specified frequency range.

- What Is Gain?
- What Is Flatness?
- Why Measure Gain and Flatness?
- Accuracy Considerations
- How to Measure Gain and Flatness

What Is Gain?

RF amplifier gain is defined as the difference in power between the amplifier output signal and the input signal. It is assumed that both input and output impedances of the amplifier are the same as the characteristic impedance of the system.

- Gain is called S21 using S-parameter terminology
- Gain is expressed in dB—a logarithmic ratio of the output power relative to the input power.
- Gain can be calculated by subtracting the input from the output levels when both are expressed in dBm, which is power relative to 1 milliwatt.
- Amplifier gain is most commonly specified as a minimum value over a specified frequency range. Some amplifiers specify both minimum and maximum gain, to ensure that subsequent stages in a system are not under or over driven.

What Is Flatness?

Flatness specifies how much the amplifier's gain can vary over the specified frequency range. Variations in the flatness of the amplifier's gain can cause distortion of signals passing through the amplifier.
Why Measure Small-Signal Gain and Flatness?

Deviations in gain over the bandwidth of interest will induce distortion in the transmitted signal because frequency components are not amplified equally. Small-signal gain allows you to quantify the amplifier's gain at a particular frequency in a 50-ohm system. Flatness allows you to view the deviations in the amplifier's gain over a specified frequency range in a 50-ohm system.

Accuracy Considerations

- The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.
- The output power of the amplifier should be sufficiently attenuated if necessary. Too much output power could:
  - damage the analyzer receiver
  - exceed the input compression level of the analyzer receiver, resulting in inaccurate measurements.

Attenuation of the amplifier's output power can be accomplished using:

  - attenuators
  - couplers

The frequency-response effects and mismatches of the attenuators and couplers must be accounted for during calibration since they are part of the test system. Proper error-correction techniques can reduce these effects.

- The frequency response is the dominant error in a small-signal gain and flatness measurement setup. Performing a thru-response measurement calibration significantly reduces this error. For greater accuracy, perform a 2-port measurement calibration.
- Reducing IF bandwidth or using averaging improves measurement dynamic range and accuracy, at the expense of measurement speed.

How to Measure Gain and Flatness

1. Preset the analyzer.
2. Select an S21 measurement parameter.
3. Set the analyzer's source power to be in the linear region of the amplifier's output response (typically 10-dB below the 1-dB compression point).
4. Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port-2.

5. Connect the amplifier as shown in the following graphic, and provide the dc bias.

6. Select the analyzer settings for your amplifier under test.

7. Remove the amplifier and perform a measurement calibration. Be sure to include the attenuator and cables in the calibration setup if they will be used when measuring the amplifier.

8. Save the instrument-state to memory.

9. Reconnect the amplifier.

10. Scale the displayed measurement for optimum viewing and use a marker to measure the small signal gain at a desired frequency.

11. Measure the gain flatness over a frequency range by using markers to view the peak-to-peak ripple.

12. Print or save the data to a disk.
Gain compression measures the level of input power applied to an amplifier that will cause a distorted output.

The Gain Compression Application (Opt S93086A/B) makes fast and accurate compression measurements.

- What Is Gain Compression?
- Why Measure Gain Compression?
- Accuracy Considerations
- How to Measure Gain Compression

What Is Gain Compression?

Gain compression occurs when the input power of an amplifier is increased to a level that reduces the gain of the amplifier and causes a nonlinear increase in output power.

The analyzer has the ability to do power sweeps as well as frequency sweeps. Power sweeps help characterize the nonlinear performance of an amplifier. Refer to the graphic below (a plot of an amplifier's output power versus input power at a single frequency) for the following discussion.

- The amplifier has a linear region of operation where gain is constant and independent of power level. The gain in this region is commonly referred to as "small-signal gain."
- As the input power increases, the amplifier gain appears to decrease, and the amplifier goes into compression.
- The most common measurement of amplifier compression is the 1-dB compression point. This is defined as
the input power (or sometimes the output power) which results in a 1-dB decrease in amplifier gain (relative to the amplifier's small-signal gain).

Why Measure Gain Compression?

When driven with a sinusoid, the output of an amplifier is no longer sinusoidal in the compression region. Some of the amplifier output appears in harmonics, rather than occurring only at the fundamental frequency of the input signal.

As input power is increased even more, the amplifier becomes saturated, and output power remains constant. At this point, further increases in amplifier input power result in no change in output power.

In some cases (such as with TWT amplifiers), output power actually decreases with further increases in input power after saturation, which means the amplifier has negative gain.

Since gain is desired in amplifier operation, it is important to know the limit of input signal that will result in gain compression.

Accuracy Considerations

The network analyzer must provide sufficient power to drive the amplifier into saturation. If you need a higher input-power level than the source of the analyzer can provide, use a preamplifier to boost the power level prior to the amplifier under test. (See High Power PNA-X.) If using a preamplifier, you can increase measurement accuracy in the following ways:

- Use a coupler on the output of the preamplifier so that a portion of the boosted input signal can be used for the analyzer's reference channel. This configuration removes the preamplifier's frequency response and drift errors from the measurement (by ratioing).
- Perform a thru-response calibration including the preamplifier, couplers, and attenuators in the test setup.

The output power of the amplifier should be sufficiently attenuated if necessary. Too much output power could:

- Damage the analyzer receiver
- Exceed the input compression level of the analyzer receiver

Attenuation of the amplifier's output power can be accomplished using:

- Attenuators
- Couplers
The frequency-response effects of the attenuators and couplers must be considered during calibration since they are part of the test system. Proper error-correction techniques can reduce these effects.

- The frequency response is the dominant error in a gain compression measurement setup. Performing a thru-response measurement calibration significantly reduces this error.
- The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.
- Reducing IF bandwidth or using measurement averages improves accuracy, at the expense of measurement speed.

How to Measure Gain Compression

This procedure shows you how to make the following three measurements used to determine amplifier gain compression:

1. A Swept-Frequency Gain Compression measurement locates the lowest frequency at which the 1-dB gain compression first occurs.

2. A Swept-Power Gain Compression measurement shows the input power at which a in a 1-dB drop in gain occurs as a power ramp is applied to the amplifier at a particular frequency point (found in measurement 1).

3. An Absolute Power measurement shows the absolute power out (in dBm) at compression.

Swept-Frequency Gain Compression Measurement

A measurement of swept frequency gain compression locates the frequency point where 1-dB compression first occurs.

1. Preset the analyzer.

2. Select an S21 measurement parameter.

3. Set the analyzer's source power to be in the linear region of the amplifier's output response (typically 10-dB below the 1-dB compression point).

4. Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port-2.

5. Connect the amplifier as shown in the following graphic, and provide the dc bias.

6. Select the analyzer settings for your amplifier under test. To reduce the effects of noise, you may want to specify a narrower IF bandwidth.
7. Remove the amplifier and perform a thru-response calibration. Be sure to include the attenuator and cables in the calibration setup if they will be used when measuring the amplifier.

8. Save the instrument-state to memory.

9. Reconnect the amplifier.

10. Position a marker at approximately mid-span.

11. Adjust the analyzer's scale to 1 dB per division.

12. Store the trace in memory and display Data/Mem.

13. Gradually increase the source power until a 1-dB decrease in gain is observed at the first frequency over some portion of the trace.

14. Use markers to locate the frequency where the 1-dB decrease in gain first occurs. Note this frequency for use in the following measurement.

15. Print the data or save it to a disk.

---

**Swept-Power Gain Compression Measurement**

A swept-power gain compression measurement shows the input power resulting in a 1-dB drop in gain as a power ramp at a particular frequency (found in step 13 of the previous measurement) is applied to the amplifier.

1. If not already done, perform the previous measurement of swept-frequency gain compression.

2. Setup an S21 measurement in the power-sweep mode. Include the following settings:

   - Set the CW frequency to the frequency noted in step 14 of the previous measurement of swept-frequency gain compression.
Enter the start and stop power levels for the sweep. The start power should be in the linear region of the amplifier’s response (typically 10 dB below the 1-dB compression point). The stop power should be in the compression region of the amplifier’s response.

3. Adjust the scale to 1-dB per division.

4. Use markers (including reference marker) to find the input power where the 1-dB decrease in gain occurs.

5. Print the data or save it to a disk.

**Absolute Output Power Measurement**

An absolute-power measurement shows the absolute power-out (in dBm) of the amplifier at compression.

1. Select an unratioed (absolute) power measurement. Choose the B input if using the test setup in the previous graphic.

2. Retain the CW frequency used in the previous measurement of swept-power gain compression.

3. Set a marker to the input power level where the 1-dB decrease in gain occurs (found in step 4 of the previous measurement).

4. Scale the displayed measurement for optimum viewing.

5. Read the marker value to find the absolute output power of the amplifier (in dBm) where the 1-dB decrease in gain occurs.

6. Print the data or save it to a disk.

**Note:** The measurement calibration does not apply to absolute power. Therefore, if there is any attenuation external to the analyzer, you will have to correct for it manually.
Gated Measurement

- Measurement Description
- Quick Setup
- Complete Measurement Setup

See All Spectrum Analyzer Examples

Measurement Description

Gated spectrum analyzer (SA) measurements (requires option S93090xA) use triggering to capture data during the active trigger state only and thus ensure that the response more closely represents the device response. Learn more.

This example shows how to set up a gated SA measurement using an external or internal pulse to trigger the receiver measurement and an internal pulse to trigger the RF source. In order to pulse the VNA RF source with an internal pulse, option S93025A/B is required. This option is not required in the case of an external RF source with pulsed capabilities.

Note: Internal Pulse0 pulse generator is not suitable for gated SA because the pulse width cannot be set. Instead, use Pulse3 as shown in this topic (pulse 1 to pulse 4 would work too, but without the direct Trigger source to Pulse3 internal connection).

Quick Setup

Many of the setup steps can be performed automatically if option S93026A/B is installed. The quick setup will configure a gated SA measurement using internal pulse generators and modulators. The RBW will be set to the maximum setting that is compatible with the pulse width. If minor adjustments are required after performing the quick setup, refer to the Complete Measurement Setup.

1. Press Meas > S-Param > Meas Class....
2. Select Spectrum Analyzer, then either:
   - OK delete the existing measurement, or
   - New Channel to create the measurement in a new channel.
3. Press Freq > Main > SA Setup... and set the frequency of the VNA source.
4. Set the source Power to the desired level.
5. Click on the Pulse Setup... button, select Standard Pulse, then ensure that the Pulse Width is much
smaller than the Pulse Period then click OK.

![Image of Pulse Setup](image)

Complete Measurement Setup

1. If using an external pulse generator, connect a cable from the pulse generator output to the **Meas Trig In BNC** on the rear panel of the VNA.
2. On the VNA front panel, press **Meas > S-Param > Meas Class**....
3. Select **Spectrum Analysis**, then either:
   - OK delete the existing measurement, or
   - New Channel to create the measurement in a new channel.
4. Press **Freq > Main > SA Setup...** and set the frequency of the VNA source.
5. Click on the Advanced tab then click on the Advanced >> button.
6. Click on the Trigger tab then click on the Trigger button.
7. Click on the Meas Trigger tab and select **Meas Trig In BNC** (if using an external pulse generator) or **Pulse3** (if using internal pulse3 to Meas Trig In bypass) under **Source** and **High Level** under **Level/Edge**.

Selecting **High Level** (or **Low Level**) under **Level/Edge** triggers LO acquisitions and continues to acquire data while the pulse trigger remains high (or low).

**Note:** Edge triggering does not perform gated measurements.
7. Click on the **Setup** tab and select **External (uses Pulse3)** (if using internal pulse3 to Meas Trig In bypass) or **External (uses MEAS TRIG IN)** (if using an external pulse generator),

8. Select **Channel** under **Trigger Scope, Point** for the **Trigger Mode**, then click **OK**.

For gated SA measurements, **Point** does not refer to a data point or display point. Instead, it refers to the next LO acquisition. For SA, each time the LO is shifted an acquisition is captured. The time for each LO acquisition is based on the **ADC Record Size** times the ADC Sampling Frequency (10 nsec). The number of LO acquisitions is determined by the **Image Reject** setting. This information is displayed in the SA Setup dialog in the Advanced tab.
9. Click on the **Source** tab and set the source **Power** to the desired level.

10. Click on the **Power and Attenuator...** button, set the **Leveling Mode** to **Open Loop**, then click **OK**.

**Open Loop** leveling is used during pulse conditions with the internal source modulator. No leveling is used in setting the source power. Learn more.

11. In the **Trigger** tab, click on the **Pulse Gen Config** button.
11. If using an internal pulse generator and an internal RF source, perform the following steps:
   a. Set the pulse **Frequency**.
   b. Enable **Pulse1** and set the **Width** greater than the acquisition time (displayed in the Advanced tab).
      Pulse1 is the RF source trigger.
   c. Ensure that **Enable Source 1 Modulator** is checked, and that **Pulse1** is enabled and selected as the **Modulator Drive**.
   d. If using the internal Pulse3 bypass, enable **Pulse3** and set the **Width** greater than the acquisition time per LO (displayed in the Advanced tab).
   e. Click **OK**.

12. If using an external pulsed generator as the measurement trigger, perform the following steps:
   a. Ensure that the pulse width is greater than the acquisition time per LO (displayed in the Advanced tab).
   b. Connect the RF source pulse trigger output to the VNA **Meas Trig In BNC** connector on the rear panel.
Group delay is a measure of phase distortion. Group delay is the actual transit time of a signal through a device under test as a function of frequency. When specifying group delay, it is important to specify the aperture used for the measurement.

- What is Group Delay?
- Group Delay versus Deviation from Linear Phase
- What Is Aperture?
- Accuracy Considerations
- How to Measure Group Delay

See also Comparing the Delay Functions.

**See other Amplifier Parameter topics**

**What Is Group Delay?**

Group delay is:

- A measure of device phase distortion.
- The transit time of a signal through a device versus frequency.
- The derivative of the device’s phase characteristic with respect to frequency.

Refer to the graphic below for the following discussion:
The phase characteristic of a device typically consists of both linear and higher order (deviations from linear) phase-shift components.

<table>
<thead>
<tr>
<th>Linear phase-shift component:</th>
<th>Higher-order phase-shift component:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Represents average signal transit time.</td>
<td>Represents variations in transit time for different frequencies.</td>
</tr>
<tr>
<td>Attributed to electrical length of test device.</td>
<td>Source of signal distortion.</td>
</tr>
</tbody>
</table>

Refer to the graphic below for the following discussion:

In a group delay measurement:

- The linear phase shift component is converted to a constant value (representing the average delay).
- The higher order phase shift component is transformed into deviations from constant group delay (or group delay ripple).
- The deviations in group delay cause signal distortion, just as deviations from linear phase cause distortion.
- The measurement trace depicts the amount of time it takes for each frequency to travel through the device under test.

Refer to the following equation for this discussion on how group delay is calculated:
Phase data is used to find the phase change \((-d\phi)\).

- A specified frequency aperture is used to find the frequency change \((d\omega)\).
- Using the two values above, an approximation is calculated for the rate of change of phase with frequency.
- This approximation represents group delay in seconds (assuming linear phase change over the specified frequency aperture).

**Group Delay versus Deviation from Linear Phase**

Group delay is often a more accurate indication of phase distortion than **Deviation from Linear Phase**.

**Deviation from linear phase** results are shown in the upper region of the following graphic: Device 1 and device 2 have the same value, despite different appearances.

**Group Delay** results are shown in the lower region: Device 1 and device 2 have different values of group delay. This is because in determining group delay, the analyzer calculates slope of phase ripple, which is dependent on number of ripples which occur per unit of frequency.

**What Is Aperture?**

During a group delay measurement, phase is measured at two closely spaced frequencies and then computes the phase slope. The frequency interval (frequency delta) between the two phase measurement points is called the aperture. Changing the aperture can result in different values of group delay.
delay. The computed slope (-delta phase / delta frequency) varies as the aperture is increased. This is why when you are comparing group delay data, you must know the aperture that was used to make the measurements.

Refer to the graphic below for the following discussion:

<table>
<thead>
<tr>
<th>Narrow aperture:</th>
<th>Wide aperture:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provides more detail in phase linearity.</td>
<td>Provides less detail in phase linearity because some phase response averaged-out or not measured.</td>
</tr>
<tr>
<td>Makes measurement susceptible to noise (smaller signal-to-noise ratio) and phase detector resolution.</td>
<td>Makes measurement less susceptible to noise (larger signal-to-noise ratio).</td>
</tr>
</tbody>
</table>

Group delay measurements can be made using the following sweep types:

- Linear frequency

- List frequency sweep segment - The group delay aperture varies depending on the frequency spacing and point density. Therefore the aperture is not constant in segment sweep. In segment sweep, extra frequency points can be defined to ensure the desired aperture.

**How to set Group Delay Aperture**

**Using Hardkey/SoftTab/Softkey**

1. Press **Format > Format 1 > Group Delay Aperture...**

**Using a mouse**

1. Click **Response**
2. Select **Format**
3. Select **Group Delay Aperture**
Group Delay Aperture dialog box help

Although the Group Delay Aperture is defined as the difference in frequency between two data points (see What Is Aperture?), the group delay calculation can be averaged over many adjacent data points, similar to the smoothing feature. The number of adjacent data points can be set using any of the following methods:

**Note:** You can change the default Group Delay Aperture to two points using a Preference. Learn how.

**Points**  
Number of adjacent data points to average. Default setting is 11 points. Choose a value between 2 and the current number of points in the channel.

**Percent of Span**  
The data points within this percentage of the current frequency span are averaged. Choose a value between (2 points / current number of points) and 100 percent. The span must contain at least two data points.

**Frequency**  
The data points within this frequency range are averaged. The frequency range must contain at least two data points.

When the frequency span or number of points is reduced so that the current Group Delay Aperture is NOT attainable, the Aperture is adjusted to the new frequency span or number of points.

**OK**  
Applies setting changes and closes the dialog box.

**Cancel**  
Closes the dialog. Setting changes are NOT applied.

Accuracy Considerations

It is important to keep the phase difference between two adjacent measurement points less than 180° (see the following graphic). Otherwise, incorrect phase and delay information may result. Undersampling may occur when measuring devices with long electrical length. You can verify that the phase difference measured between two adjacent points is less than 180° by adjusting the following settings until the measurement trace no longer changes:

- Increase the number of points
- Narrow the frequency span

Electrical delay may also be used to compensate for this effect.
The frequency response is the dominant error in a group delay test setup. Performing a thru-response measurement calibration significantly reduces this error. For greater accuracy, perform a 2-port measurement calibration.

Particularly for an amplifier, the response may vary differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.

**How to Measure Group Delay**

1. Preset the analyzer.

2. If your DUT is an amplifier, it may be necessary to adjust the source power:
   - Set the source power to be in the linear region of the amplifier’s output response, typically 10 dB below the 1 dB compression point.
   - If needed, use an external attenuator so the amplifier output power will be sufficiently attenuated to avoid causing receiver compression or damage to test port 2.

3. Connect the DUT as shown in the following graphic.
4. Select an S21 measurement.

5. Select the settings for your DUT:
   - frequency range
   - number of measurement points.
   - format: delay
   - scale: autoscale

6. Remove the DUT and perform a measurement calibration.

7. Reconnect the DUT.

8. Scale the displayed measurement for optimum viewing.

9. Use the Group Delay Aperture setting to increase the aperture, reducing noise on the trace while maintaining meaningful detail.

10. Use the markers to measure group delay (expressed in seconds) at a particular frequency of interest.

11. Print the data or save it to a disk.
High-Gain Amplifier Measurements

When measuring High-Gain Amplifiers, errors in measuring any of the S-parameters during calibration can result in error in the S21 measurement. This is because all the S-parameters are used in the error correction math.

A particular problem occurs with high gain amplifiers because the source power is set very low. Thus, when making reverse measurements (S22, S12) the signal-to-noise is poor and the raw measurements can be dominated by noise. This noise in the raw measurements will result in a noisy trace appearing for corrected S21 or S11.

If you are using a large attenuator on port 2 (which improves output match), perform an Enhanced Response Calibration as follows. This corrects for the same errors as the full 2-port correction EXCEPT the interaction between the raw load match and the DUT output match.

1. There is NO need to Uncouple the port powers.
2. Set port powers to an acceptable level. Do NOT overpower the test port.
3. Perform Enhanced Response Cal. Learn how. (Does not measure or correct for S12 or S22 port match).

If you want to do a full correction (for example, when your amplifier output match is poor so the Enhanced Response Cal above is not adequate), then...

1. Uncouple the port powers. Learn how.
2. Set input (port 1) power to approximately the output power of the amplifier up to 0 dBm
3. Set reverse (port 2) power to the same power (for measuring isolation and S22)
4. Perform a Full 2-port Cal.
5. Re-set the input power (port 1) to a lower power level appropriate for driving the amplifier.
Additional Error due to Mismatch of DUT Output Match and Raw Load Match
High-Power Amplifier Measurements with the VNA

The following is a block diagram of all models of the PNA-X Opt 423. However, all of the models listed below have the same or fewer components.

The configuration displayed here is used to make high power amplifier measurements using a preamplifier at the rear panel. The preamplifier can then be switched (SW1) as needed using the RF Configurator.

Damage Levels
<table>
<thead>
<tr>
<th>Model</th>
<th>Bridges/Combiners</th>
<th>Couplers</th>
<th>Bias-tees</th>
</tr>
</thead>
<tbody>
<tr>
<td>N5241B, N5242B, N5249B</td>
<td>+33 dBm</td>
<td>+43 dBm</td>
<td>+30 dBm</td>
</tr>
<tr>
<td>N5244B, N5245B</td>
<td>+27 dBm</td>
<td>+43 dBm</td>
<td>+30 dBm</td>
</tr>
<tr>
<td>N5247B</td>
<td>+27 dBm</td>
<td>+30 dBm</td>
<td>+30 dBm</td>
</tr>
<tr>
<td>N5221B, N5222B</td>
<td>N/A</td>
<td>+43 dBm</td>
<td>+30 dBm</td>
</tr>
<tr>
<td>N5224B, N5225B</td>
<td>N/A</td>
<td>+43 dBm</td>
<td>+30 dBm</td>
</tr>
<tr>
<td>N5227B</td>
<td>N/A</td>
<td>+30 dBm</td>
<td>+30 dBm</td>
</tr>
</tbody>
</table>

**ALL Switches:** Damage level = 1 Watt (+30 dBm) while switching.

**ALL Step Attenuators:** Damage level = 1 Watt (+30 dBm) while switching.

**Notes**

**At J11** (rear-panel), max power is 4 dB to 11 dB higher than Source 1 Out at front panel jumper due to loss of the coupler thru arms, bias-tees, and cables.

**At J10** max power +33 dBm, which is the damage level of the bridge. With +30 dBm into J10, there will be about +15 dBm at R1, assuming 15 dB coupling factor for the R1 bridge. +15 dBm is the damage level of that receiver. Therefore, it may be necessary to add attenuation in place of the R1 loop, not only to protect the receiver, but to bring it out of compression. The 0.1 dB compression level spec for the R1 receiver is between -3 and -18 dBm, depending on the frequency and option configuration.

**At Test Port 2** (DUT output): With the bias-tees (orange), only +30 dBm is allowed into the test port. With Opt 222/422 (bias-tees removed), +43 dBm is allowed. Add appropriate attenuation to not damage other components.

**About the PNA-X Option 222/422**

Option 222 and 422 are designed to permit insertion of high power amplifiers and other signal conditioning equipment to allow high power network measurements at RF levels up to 20 Watts (+43 dBm) from 10 MHz to 26.5 GHz. These options supply extended power range attenuators without bias tees. This is similar to the PNA-X -219 (add extended power range and bias-tees to 2-Port analyzer) or PNA-X -419 (add extended power range and bias-tees to 4-Port analyzer) but deletes the bias tees from the test set.

**See Also**
- High-Gain Amplifier Measurements
- RF Path Configurator
- IF Path Configurator
- High-power measurements using the PNA (5989-1349EN) Application Note 1408-10 at Keysight.com.
Noise Power Ratio (NPR) Measurement

In this topic:

- Requirements
- Measurement Description
- Physical Setup
- NPR Measurement Procedure
- Calibrate Modulation

Requirements

- Spectrum Analyzer Option S93090xA/B
- 2- or 4-Port PNA with ability to connect an external modulated source
- Supported external sources:
  - E8267D PSG Vector Signal Generator
  - M8190A with E8267D PSG Vector Signal Generator
  - M9383A MCS (wideband vector, 44 GHz, with enhanced phase noise)
  - N5182B MXG RF Vector Signal Generator
  - N5192A and N5194A UXG Vector Adapter

See All Spectrum Analyzer Examples

Measurement Description

NPR measures the nonlinear behavior of an RF microwave amplifier under a modulated signal stimulus.

In this example, a setup file consisting of a multi-tone signal with a "notch" is created, uploaded to a signal generator, then fed into the VNA to test the behavior of the device. The "notch" refers to a small part of the frequency axis (up to 10% of the frequency span) that is forced to 0 (no signal) creating a notch in the power spectral density.
NPR is the sum of all intermodulation products across the pass band ratioed with the sum of all intermodulation products in the notch.

**Physical Setup**

The following diagram shows a typical hardware setup using an M8190A with a E8267D PSG with the signal connected to the VNA rear-panel Port 1 J10 input connector.

**NPR Measurement Procedure**

1. Connect the equipment as shown above.
2. On the PNA front panel, press **Preset**.
3. On the VNA front panel, press **Meas > S-Param > Meas Class...**.
4. Select **Spectrum Analysis**, then either:
   - **OK** delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.
5. The **SA Setup** dialog will be displayed.
6. Click on the **Source** tab. The **Source** dialog will be displayed.

7. If you need to add an external source, click on the **External Devices...** button. Learn more. After adding an external source, it should be listed in the **Name** column. The following is an example showing a modulation...
source called MXG. If the external source does not have modulation capabilities, the IQMod column will be grayed out.

<table>
<thead>
<tr>
<th>Name</th>
<th>State</th>
<th>Type</th>
<th>Frequency</th>
<th>Power</th>
<th>Phase</th>
<th>Pulse</th>
<th>IQMod</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td>OFF</td>
<td>CW</td>
<td>CW Freq 1.0000000000000000 GHz</td>
<td>-5.00 dBm</td>
<td>+0.0</td>
<td>OFF</td>
<td></td>
</tr>
<tr>
<td>Port 2</td>
<td>OFF</td>
<td>CW</td>
<td>CW Freq 1.0000000000000000 GHz</td>
<td>-5.00 dBm</td>
<td>+0.0</td>
<td>OFF</td>
<td></td>
</tr>
<tr>
<td>Port 3</td>
<td>OFF</td>
<td>CW</td>
<td>CW Freq 1.0000000000000000 GHz</td>
<td>-5.00 dBm</td>
<td>+0.0</td>
<td>OFF</td>
<td></td>
</tr>
<tr>
<td>Port 4</td>
<td>OFF</td>
<td>CW</td>
<td>CW Freq 1.0000000000000000 GHz</td>
<td>-5.00 dBm</td>
<td>+0.0</td>
<td>OFF</td>
<td></td>
</tr>
<tr>
<td>Port 1 Src2</td>
<td>OFF</td>
<td>CW</td>
<td>CW Freq 1.0000000000000000 GHz</td>
<td>-5.00 dBm</td>
<td>+0.0</td>
<td>OFF</td>
<td></td>
</tr>
</tbody>
</table>

8. Click on the Path Configuration... button and ensure that the correct RF path is selected according to the Physical Setup then click OK. For this example, the Combiner path for Port 1 is selected since the modulated signal is connected to the rear-panel J10 connector.

9. Click in the State field of your external source to change the state to ON.

10. Click in the Frequency field of your external source to change the frequency.

11. Click in the Power field of your external source to change the power.

12. Ensure that the appropriate reference receiver (signal at input of DUT) or test port receiver (signal at output of DUT) is selected based on the Physical Setup.
13. Click in the **IQMod** column for your external source and select **Edit** (refer to step 7 dialog example above) to access the **Modulation settings** dialog.

14. Click on the **Create/Edit Mod File...** button to access the **Create Modulation** dialog.
15. Click on the **Source Name** drop-down menu to select the source connected to the VNA that will be used to generate the modulated test signal.

16. Enter the span of the modulation signal in the **Signal Span** field.

17. Enter the tone spacing for the multi-tone signal in the **Tone Spacing** field.

18. The **Number of Tones** corresponds to the span and tone spacing. This value will be calculated when you click on the **Calculate** button.

19. Enter the span of the notch in the **Notch1 Span** field. The notch can be up to 10% of the span.

20. In the **Notch Location** drop-down menu, select **Symmetric** to position the notch in the center.

21. Click on the **Calculate** button to view the signal.

22. When the signal setup is complete, click on the **Save...** button to save this setup to a *.mdx file.

23. Click on the **OK** button to return to the **Modulation settings** dialog. The file path and name are displayed in the **Modulation Filename** field.
24. Check the **Enable Modulation** check box.

25. The **Autoset Frequencies and Coherence for current modulation** and **Autoset Markers** for the current modulation are checked by default. These options automatically adjust the spectrum analyzer frequency and spectrum analyzer markers each time adjustments are made to the modulation settings.

26. Click on the **OK** button to close the dialog. The following is an example signal:
Markers

The \( ? \) marker displays the notch frequency relative to the center frequency. In this example the notch is in the center so the frequency is 0 Hz. Also displayed is the notch marker power level.

The \( > R \) marker displays the center frequency (16 GHz) and the signal span (1.5 GHz).

The Band Power displays the total power within the signal span.

The Tone(1.40 GHz) displays the average tone power across the 1.5 GHz signal span minus the average tone power across the notch (1.5 GHz - 100 MHz = 1.4 GHz).

To learn more about these markers, refer to SA Analysis Markers.
27. To view the signal at the output of the DUT, add a new trace and select the B test port receiver since the output of the DUT is connected to port 2.

![Image of New Trace window]

28. The following is an example. The blue trace is the signal at the output of the DUT.
Calibrate Modulation

The Modulation Cal procedure optimizes the signal across the pass band and the depth of the notch.

1. Click on the **SA Setup...** button to open the **SA Setup** dialog then select the **Source** tab.

2. Click in the **IQMod** column for your external source and select **Edit** to access the **Modulation settings** dialog.
3. Click on the **Calibrate Modulation...** button. The **Modulation Cal - Setup** dialog is displayed.

4. Select **Power** to calibrate power across the entire pass band.

5. Select **Flatness** to calibrate flatness across the entire pass band.

6. Select **NPR Notch** to create as deep a notch as possible.

7. Under **Calibrate At**, select **Rcvr R1** for **Power Cal**, **Flatness Cal**, and **NPR Notch Cal**. This will perform a calibration at the input of the DUT.

8. The default values for **Max Nmbr Iterations** and **Desired Tolerance** will be used.

9. Click **Next**. The following dialog is displayed.
10. Click **Calibrate** to begin the calibration. When finished, the corrected modulated signal is displayed.
11. Click on the **Finish** button then click on the **Save & Exit** button to save the file. If one or more of the default tolerance values could not be achieved, a warning message is displayed. You have the option to change the tolerance values and re-calibrate or save the current calibration.

12. In the **Modulation Cal - Measure** dialog, note that the failed calibration is indicated in the text box.
13. In the **Modulation settings** dialog, **Enable Modulation Correction** is automatically checked if the user exits the wizard with **Exit & Save Cal**.

14. Click **OK**.

15. The following shows an optimized input signal.
We can now see the Power Amplifier DUT output NPR (blue trace -14.90 dB) and still monitor the NPR at the DUT input (yellow trace -27.61 dB here).
Phase Measurements

Knowledge of both magnitude and phase characteristics is needed for successful higher-level component integration.

- What are Phase Measurements?
- Why Measure Phase?
- Using the Analyzer's Phase Format
- Types of Phase Measurements

What are Phase Measurements?

Phase measurements are made using S-parameters, just like amplitude measurements. A phase measurement is a relative (ratio) measurement and not an absolute measurement. Phase measurements compare the phase of the signal going into a device (the incident signal) to the phase of the device's response signal. The response signal can be either reflected or transmitted. Assuming an accurate calibration has been performed, the difference in phase between the two signals (known as phase shift) is a result of the electrical characteristics of the device under test.

The following graphic shows the phase shift (in time or degrees) between an incident signal and a transmitted signal (as might be seen on an oscilloscope display).

Why Measure Phase?

Measuring phase is a critical element of network analysis. The following graphic lists five reasons for measuring both magnitude and phase.
When used in communications systems to pass signals, components or circuits must not cause excessive signal distortion. This distortion can be:

- Linear, where flat magnitude and linear phase shift versus frequency is not maintained over the bandwidth of interest.
- Nonlinear, such as AM-to-PM conversion.

It is important to measure how reflective a component or circuit is, to ensure that it transmits or absorbs energy efficiently. Measuring the complex impedance of an antenna is a good example.

**Using the Analyzer's Phase Format**

The analyzer's phase format displays a phase-versus-frequency or phase-versus-power measurement. The analyzer does not display more than ±180 degrees phase difference between the reference and test signals. As the phase value varies between +180 degrees and -180 degrees, the analyzer display creates the sawtooth pattern as shown in the following graphic.

The sawtooth pattern does not always reach +180 degrees and -180 degrees. This is because the measurement is made at discrete frequencies, and the data point at +180 degrees and -180 degrees may not be measured for the selected sweep.
Types of Phase Measurements

- **Complex impedance** data is information such as resistance, reactance, phase, and magnitude that can be determined from an S11 or S22 measurement. Complex impedance data can be viewed using either the Smith Chart format or the Polar format.

- **AM-to-PM conversion** is a measure of the amount of undesired phase deviation (PM) that is caused by amplitude variations (AM) of the system. AM-to-PM conversion is usually defined as the change in output phase for a 1-dB increment in the input power to an amplifier (i.e. at the 1 dB gain compression point). This is expressed in degrees-per-dB (°/dB).

- **Deviation from linear phase** is a measure of phase distortion caused by a device. Ideally, the phase shift through a device is a linear function of frequency. The amount of variation from this theoretical phase shift is known as its deviation from linear phase (also called phase linearity).

- **Group delay** is another way to look at phase distortion caused by a device. Group delay is a measure of transit time through a device at a particular frequency. The analyzer computes group delay from the derivative of the measured phase response.

**Deviation from Linear Phase Versus Group Delay**

Although deviation from linear phase and group delay are similar measurements, they each have their purpose.

The following are the advantages of deviation from linear phase measurements:

- Less noisy than group delay.
- Able to characterize devices that pass phase modulated signals, and show units of phase rather than units of seconds.

The following are the advantages of group delay measurements:

- More easily interpreted indication of phase distortion than deviation from linear phase.
- Able to most accurately characterize a device under test. This is because in determining group delay, the analyzer calculates the slope of the phase ripple, which is dependent on the number of ripples which occur per unit of frequency. Comparing two phase responses with equal peak-to-peak phase ripple, the response with the larger phase slope results in:
  - More group delay variation.
  - More signal distortion.
Reverse isolation is a measure of how well a signal applied to the device output is "isolated" from its input.

- What is Reverse Isolation
- Why Measure Reverse Isolation?
- Accuracy Considerations
- How to Measure Reverse Isolation

What is Reverse Isolation?

Reverse isolation is a measure of how well a signal applied to the device output is "isolated" from its input.

The measurement of reverse isolation is similar to that of forward gain, except:

- The stimulus signal is applied to the amplifier's output port.
- The response is measured at the amplifier's input port.

The equivalent S-parameter is S12.

Why Measure Reverse Isolation?

An ideal amplifier would have infinite reverse isolation-no signal would be transmitted from the output back to the input. However, reflected signals can pass through the amplifier in the reverse direction. This unwanted reverse transmission can cause the reflected signals to interfere with the desired fundamental signal flowing in the forward direction. Therefore, reverse isolation is important to quantify.

Accuracy Considerations

Since amplifiers often exhibit high loss in the reverse direction, generally there is no need for any attenuation that may have been used to protect the port 2 receiver during forward transmission measurements. Removing the attenuation will:
Increase the dynamic range, resulting in improved measurement accuracy.

Require a new calibration for maximum accuracy.

The RF source power can be increased to provide more dynamic range and accuracy.

**Note:** With the attenuation removed and the RF source power increased, a forward sweep could damage the analyzer's port 2 receiver. Do not perform a forward sweep or use 2-port calibration unless the forward power is set low enough to avoid causing port 2 receiver compression or damage.

If the isolation of the amplifier under test is very large, the transmitted signal level may be near the noise floor or crosstalk level of the receiver. To lower the noise floor:

- Use or increase measurement averages.
- Reduce the IF bandwidth of the analyzer.

**Note:** Reducing IF bandwidth or using averaging improves measurement dynamic range and accuracy, at the expense of reduced measurement speed.

- When crosstalk levels affect the measurement accuracy, reduce the crosstalk error term by performing a response and isolation calibration. When performing the isolation part of the calibration it is important to use the same average factor and IF bandwidth during the calibration and measurement.
- The frequency response of the test setup is the dominant error in a reverse isolation measurement. Performing a thru-response measurement calibration significantly reduces this error. This calibration can be done as part of the response and isolation calibration.
- The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.

**How to Measure Reverse Isolation**

1. Connect the amplifier as shown in the following graphic.
2. Preset the analyzer.

3. Select an S12 measurement.

4. Select the settings for your amplifier under test.

5. Remove the amplifier and perform a thru-response calibration or a response and isolation calibration.

6. Scale the displayed measurement for optimum viewing and use a marker to measure the reverse isolation at a desired frequency.

7. Print or save the data to a disk.
Reflection Measurements

Reflection measurements are an important part of network analysis.

- What are Reflection Measurements?
- Why Make Reflection Measurements?
- Expressing Reflected Waves
  - Return Loss
  - VSWR
  - Reflection Coefficient
  - Impedance
  - Summary of Expressions

What are Reflection Measurements?

To understand reflection measurements, it is helpful to think of traveling waves along a transmission line in terms of a lightwave analogy. We can imagine incident light striking some optical component like a clear lens. Some of the light is reflected off the surface of the lens, but most of the light continues on through the lens. If the lens had mirrored surfaces, then most of the light would be reflected and little or none would be transmitted.

1. Incident  2. Reflected  3. Transmitted

With RF energy, reflections occur when the impedance of two mated devices are not the same. A reflection measurement is the ratio of the reflected signal to the incident signal. Network analyzers measure the incident wave with the R (for reference) channel and the reflected wave with the A channel. Therefore, reflection is often shown as the ratio of A over R (A/R). We can completely quantify the reflection characteristics of our device under test (DUT) with the amplitude and phase
information available at both the A and R channel. In S-parameter terminology, S11 is a reflection measurement of port 1 of the device (the input port); S22 is a reflection measurement of the port 2 (the output port).

### Why Make Reflection Measurements?

One reason we make reflection measurements to assure efficient transfer of RF power. We do this because:

1. RF energy is not cheap. When energy is reflected, that means less energy is transmitted to where it is intended to go.

2. If the reflected energy is large, it can damage components, like amplifiers.

For example, in the following graphic, the radio station on the left is not operating at peak efficiency. The amplifier impedance is not the same as the transmission line, and the transmission line impedance is not the same as the antenna. Both of these conditions cause high reflected power. This condition results in less transmitted power, and the high reflected power could damage the amplifier.

![Reflection Measurement Diagram](image)

The radio station on the right installed properly "matched" transmission line and antenna. Very little of the transmitted signal is reflected, resulting in increased broadcast power, more listeners, more advertising revenue, and more profit. The amplifier, transmission, and antenna all need to be measured to ensure that reflected power is minimized.

### Expressing Reflected Waves

After making a reflection measurement, the reflection data can be expressed in a number of ways, depending on what you are trying to learn. The various expressions are all calculated by the analyzer from the same reflection measurement data. Each method of expressing reflection data can be graphically displayed in one or more formats. For more information, see display formats.

#### Return Loss

The easiest way to convey reflection data is return loss. Return loss is expressed in dB, and is a scalar (amplitude only) quantity. Return loss can be thought of as the absolute value or dB that the reflected signal is below the incident signal. Return loss varies between infinity for a perfect impedance match.
and 0 dB for an open or short circuit, or a lossless reactance. For example, using the log magnitude format on the analyzer, the measured reflection value on the screen may be -18dB. The minus sign is ignored when expressing return loss, so the component is said to have 18dB of return loss.

**VSWR**

Two waves traveling in opposite directions on the same transmission line cause a "standing wave". This condition can be measured in terms of the voltage standing wave ratio (VSWR or SWR for short). VSWR is defined as the maximum reflected voltage over the minimum reflected voltage at a given frequency. VSWR is a scalar (amplitude only) quantity. VSWR varies between one for a perfect match, and infinity for an open or short circuit or lossless reactance.

**Reflection Coefficient**

Another way of expressing reflection measurements is reflection coefficient gamma (\(\Gamma\)). Gamma includes both magnitude and phase.

The magnitude portion of gamma is called rho (\(\rho\)). Reflection coefficient is the ratio of the reflected signal voltage to the incident signal voltage. The range of possible values for \(\rho\) is between zero and one. A transmission line terminated in its characteristic impedance will have all energy transferred to the load; zero energy will be reflected and \(\rho = 0\). When a transmission line terminated in a short or open circuit, all energy is reflected and \(\rho = 1\). The value of rho is unitless.

Now for the phase information. At high frequencies, where the wavelength of the signal is smaller than the length of conductors, reflections are best thought of as waves moving in the opposite direction of the incident waves. The incident and reflected waves combine to produce a single "standing" wave with voltage that varies with position along the transmission line.

When a transmission line is terminated in its characteristic impedance (Zo) there is no reflected signal. All of the incident signal is transferred to the load, as shown in the following graphic. There is energy flowing in one direction along the transmission line.

When a transmission line is terminated in a short circuit termination, all of the energy is reflected back to the source. The reflected wave is equal in magnitude to the incident wave (\(\rho = 1\)). The voltage across any short circuit is zero volts. Therefore, the voltage of the reflected wave will be 180 degrees out of phase with the incident wave, canceling the voltage at the load.
When a transmission line is terminated in an open circuit termination, all of the energy is reflected back to the source. The reflected wave is equal in magnitude to the incident wave ($\rho = 1$). However, no current can flow in an open circuit. Therefore, the voltage of the reflected wave will be in phase with the voltage of the incident wave.

When a transmission line is terminated in a 25 ohm resistor, some but not all of the incident energy will be absorbed, and some will be reflected back towards the source. The reflected wave will have an amplitude $1/3$ that of the incident wave and the voltage of the two waves will be out of phase by 180 degrees at the load. The phase relationship will change as a function of distance along the transmission line from the load. The valleys of the standing wave pattern will no longer go to zero, and the peaks will be less than that of the open / short circuit.

For more information, see Phase Measurements.
Impedance is another way of expressing reflection data. For more information on Impedance, see Smith Charts.

Summary of the Expressions of Reflection Measurements:
SA Amplifier Harmonics Measurement

- Physical Setup
- Measurement Description
- Step 1. Set Up Amplifier Harmonics Measurement
- Step 2. Validate Receiver
- Step 3. Validate Source

See All Spectrum Analyzer Examples

Physical Setup

Measurement Description

This measurement example helps you understand the basic operation of the Spectrum...
Analyzer application and how to optimize the measurement setup to avoid instrument-generated spurious responses. This example measures the harmonics at maximum gain of an amplifier and confirms that the harmonics are solely caused by the amplifier, not by the PNA source or receivers.

Step 1. Set Up Amplifier Harmonics Measurement

1. On the PNA front panel, press **Preset**.

2. Press **Power > Port Power > Select** and select Port 1.

3. Press **Power > Port Power > Power Level** and set the Port 1 source power to be in a linear region of the amplifier's output response.

4. Press **Freq** and set the start/stop frequency to measure the gain across the amplifier's frequency range.

5. Press **Meas > S-Param > S21**. You should see the amplifier's frequency response as shown in the following example.

6. Press **Search > Main > Max Search** to place a marker on the maximum gain.

7. Press **Marker > Marker -> Functions > Marker -> SA** to view the amplifier's spectrum at the maximum gain. Note that a second channel with Spectrum Analyzer mode is created as shown in the following example.
8. Press Setup > Main > SA Setup... then Resolution Bandwidth and adjust (narrow) the resolution bandwidth to lower the noise floor. The following example shows the lowered noise floor after changing the resolution bandwidth from 1 MHz to 100 kHz.

9. Press Marker > Marker 1-7 > Marker 2 and place the marker on the second harmonic level as shown in the following example.
Step 2. Validate Receiver

The receiver will be validated by monitoring the amplitude of the 2\textsuperscript{nd} harmonic while increasing the receiver B attenuator. The resolution bandwidth will be adjusted if necessary to lower the noise floor. The receiver B attenuator will only affect the power input to the receiver thus affecting the receiver harmonics. However, it does not affect the DUT’s 2\textsuperscript{nd} harmonic. Therefore, if the measured 2\textsuperscript{nd} harmonic does not change when the receiver B attenuator is changed, the receiver B is not contributing the 2\textsuperscript{nd} harmonic measurement.

1. Press Setup > Main > SA Setup....

2. Increase the Receiver B Attenuator then click Apply.

3. Monitor the 2\textsuperscript{nd} harmonic amplitude for any changes.

Step 3. Validate Source

To validate the source, an R1 trace is added and source attenuation is increased to verify if a 2\textsuperscript{nd} harmonic signal is being generated from the Port 1 source.

When source attenuation increases but the set Port 1 output power remains the same, the source outputs more power to maintain the set Port 1 output power. The R1 coupler is before the source attenuator thus it sees higher power from the source than that with the lower source attenuator setting. At a certain point, R1 is compressed and starts generating the 2\textsuperscript{nd} harmonic.

As the R1 receiver has no step attenuators, to verify if it is the source or the R1 receiver generating the
harmonic, measure the same harmonic signal with receiver B but without the DUT (using a Thru connection from Port 1 to Port 2). Attenuators may be added to the receiver B to avoid compression and the resolution bandwidth can be adjusted to lower the noise floor. If receiver B sees the harmonic signal, it is from the source (or both); If the R1 receiver sees the harmonic but receiver B does not, the harmonic is due to the R1 receiver compression.

1. From the menus across the top of the display, press Trace > Trace Setup > Measure..., select R1 to add an R1 trace, then click OK. Refer to the following example.

![New Trace dialog box](image1)

2. Press Freq > Main > SA Setup..., Source tab, then Power and Attenuator... button.

3. Deselect Auto Range for Port 1, increase Source Atten to 20 dB, click OK, then click OK in the Source tab. Refer to the following example.

![Power and Attenuator dialog box](image2)

4. Monitor the 2nd harmonic on the R1 trace for any changes.
5. Press **Freq > Main > SA Setup...**, **Source** tab, then **Power and Attenuator...** button.

6. Increase **Source Atten** to 30 dB, click **OK**, then click **OK** in the **Source Setup** tab. Refer to the following example.

7. Monitor the 2\textsuperscript{nd} harmonic on the R1 trace for any changes.
SA Converter Spurious Measurement

- Physical Setup
- Measurement Description
  - Step 1. Set Up the Converter SC21 Measurement
  - Step 2. Analyze Spurious Signals at IF Output
  - Step 3. Analyze Signals Using Markers

See All Spectrum Analyzer Examples

Physical Setup

Measurement Description

This measurement example sets up a converter Scalar Mixer/Converter (SMC) + Phase measurement, monitors the spurious response on SC21 at specific frequencies, then shows how to use the Spectrum
Analyzer application for analysis.

Step 1. Set Up the Converter SC21 Measurement

1. On the VNA front panel, press **Preset**.

2. Press **Meas > S-Param > Measurement Class**, select **Scalar Mixer/Converter + Phase**, then click **OK**.

3. In the **SMC Setup** dialog that appears, select the **Mixer Setup** tab to configure the SMC setup.

4. Select **Port 3** for LO1. Refer to the following example.

5. Select the **Mixer Power** tab, set **LO1 Power** to 0 dBm, then click **Apply**.

6. Select the **Mixer Frequency** tab, set converter frequencies, then click **Apply**. Refer to the following example.
7. Select the **Power** tab, set Port 1 **Power Level**, check **Power On (All Channels)**, then click **OK**.

8. Place a marker on any unknown signal of the SC21 trace as shown in the following example. (Note that the DUT used in this example has an amplifier stage and a filter.)

---

**Step 2. Analyze Spurious Signals at IF Output**

The following Spectrum Analyzer measurement example will show the IF signal at 1.1 GHz at the output of the converter when the RF is at CW 3.3 GHz. Typical symptoms include an unstable IF
response or intermittent spurious on converter output. To find the cause, the IF, LO, and RF frequency ranges must be analyzed.

1. Press **Marker > Marker -> Functions > Marker -> SA**. The following shows an RF and IF response and is an example of a spurious signal.

![RF and IF response example](image)

2. Press **Freq > Main > SA Setup...** then select the **Source** tab.

3. Change the **Port 1 Type** from CW to Linear (the source will sweep from start to stop frequency).

4. Change the **Port 1 Frequency** range to be the same as the SMC channel, set **Source Number of Steps** to 51, set **SA Sweeps per Source Step** to 1, then click **OK**. Refer to the following example.
5. Ensure that Port 3 is set to CW 2.2 GHz then click OK.

6. Observe the IF response while sweeping the input (RF) signal.

7. Select Freq > Main > SA Setup..., set Start frequency to 500 MHz, and set Stop frequency to 3.5 GHz.

8. Deselect Resolution Bandwidth Auto, change bandwidth to 100 kHz, then click OK. This allows the IF, LO, and RF signals to be viewed on the spectrum as shown in the following example. Note that the signals from the IF port are sweeping as the RF signal sweeps.
Step 3. Analyze Signals Using Markers

1. Press **Trigger** > **Main** > **Hold** to freeze the sweep, then apply markers to signals as shown in the following example.

2. Observe the signals shown in the example above:
   - Marker 4: RF, 3.1 GHz
   - Marker 3: LO, 2.2 GHz
   - Marker 2: IF, 902 MHz
   - Marker 5: spur, 1.3 GHz

   The spur frequency can be derived from the marker information:

   \[ LO - IF = LO - (RF - LO) = 2LO - RF \]

3. Press **Freq** > **Main** > **SA Setup...**, set the stop frequency to 7.5 GHz then click **OK** to observe higher-order components.

4. Press **Trigger** > **Main** > **Continuous** to sweep the channel. The frequency components are sweeping in accordance with the RF sweep.

5. Press **Trigger** > **Main** > **Hold** to investigate the signals. Note that in the following example, the marker 5 frequency is 2LO.
5. Press **Trace** > **Trace Setup** > **Measure**..., select **R3**, select **C**, then click **OK**. This adds an R3 and C receiver measurement to determine if an LO harmonic is from the Port 3 LO source or from the converter circuit. If the signal appears in the R3 reference receiver measurement, then the spur is generated by the Port 3 LO source. If the signal appears in the C test receiver measurement, then the spur is generated by the DUT. Refer to the following examples.
Avg BW

- Standard / Differential IQ / IM Spectrum / Scalar Mixer/Converter + Phase and Vector Mixer/Converter
- Active Hot Parameters
- Gain Compression and Gain Compression Converters
- Swept IMD / IMDX
- Noise Figure Cold Source and Noise Figure Converters
- Modulation Distortion and Modulation Distortion Converters
- Phase Noise
- Spectrum Analyzer
Cal

Standard Cal

Active Hot Parameters

Gain Compression and Gain Compression Converters Cal

Differential I/Q Cal

IM Spectrum and IM Spectrum Converters Cal

Swept IMD and Swept IMD Converters Cal

Modulation Distortion and Modulation Distortion Converters Cal

Noise Figure Cold Source and Noise Figure Converters Cal

Phase Noise

Scalar Mixer/Converter + Phase and Vector Mixer/Converter Cal

Spectrum Analyzer Cal
Channel

Channel 1-8

- Channel 1-8

Channel Setup

- Select
- Meas Class...
- Add Channel
- Copy Channel
- Delete Channel
Display

Window 1-8

- Windows 1-8

Window Setup

- Select: *Select the desired window.*
- Window Title...
- Add Window
- Delete Window
- Move Window...
- Window Layout

Sheet Setup

- Select
- Sheet Title...
- Add Sheet
- Delete Sheet
- Sheet Layout

Display Setup

- Trace Maximize
- Window Max
- Show Table
- Customize Display...
- Touch screen: *Enable/disable touch screen operation*
- Display Update
Format

Format 1

- Log Mag
- Lin Mag
- Phase
- Delay
- Smith
- Polar
- SWR
- Group Delay Aperture...

Format 2

- Real
- Imaginary
- Unwrapped Phase
- Positive Phase
- Inverted Smith
- Complex
- Temperature
Freq

Standard
Active Hot Parameters
Gain Compression
Differential I/Q
IM Spectrum and IM Spectrum Converters
Swept IMD and Swept IMD Converters
Noise Figure Cold Source
Noise Figure Converters
Phase Noise
Modulation Distortion and Modulation Distortion Converters
Gain Compression Converters
Scalar Mixer/Converter + Phase and Vector Mixer/Converter
Spectrum Analyzer
Macro

Favorite1-3

- Macro Setup...

To Add a Favorite, press and hold any softkey for three seconds and select the desired Favorite number (Favorite 1 to 3).

Macro1-3

- Port Config
- Src Power Alignment
- SA Multi-module Cal
- 2nd Src Config
- Factory Cal Adjust
- Multiport Cal Assist
- MM Phase Cal
- Ref. In Config
- Dummy DUT Tool

To Add a Macro, refer to Using Macros.

Key Setup

- Macro Setup...
- Clear Favorites: *Clear all favorites items*
Marker

Marker 1-7

- Marker 1 to 7
- Reference

Marker 8-15

- Market 8 to 15

Marker Setup

- Delta
- Discrete
- Type
- Format
- Coupled
- Marker Display...
- Marker Table
- All Off

Marker -> Functions

- Marker -> Start
- Marker -> Stop
- Marker -> Center
- Marker -> Span
- Marker -> Ref Level
- Marker -> Delay
- Marker -> CW Freq
Marker -> IM Spectrum
Marker -> SA

SA Analysis

- Band Power
- Band Density
- Band Span
- Equivalent Span
- Density BW
- Occupied BW
- OBW Percent

- Band Function (Modulation Distortion Only)
- Band Span (Modulation Distortion Only)
## Math

### Memory

- Data -> Memory
- Normalize
- Data Math
- Display
- 8510 Mode
- Interpolate

### Analysis

- Conversions
- Equation Editor...
- Statistics...
- AM Distortion
- Trace Deviation
- Uncertainty Analysis...
- Limits...
- Limit Table
- Compression Analysis...

### Time Domain

- Transform
- Start Time
- Stop Time
- Center Time
- Span Time
• TD Mode
• TD Toolbar
• Time Domain Setup...

Time Gating

• Gating
• Gate Start
• Gate Stop
• Gate Center
• Gate Span
• Gate Type
• Gate Shape
• Gating Setup...
Meas

Standard
Active Hot Parameters
Gain Compression
Differential I/Q
IM Spectrum
IM Spectrum Converters
Swept IMD and Swept IMD Converters
Modulation Distortion and Modulation Distortion Converters
Phase Noise
Noise Figure Cold Source
Noise Figure Converters
Gain Compression Converters
Scalar Mixer/Converter + Phase
Vector Mixer/Converter
Spectrum Analyzer
Power

Standard

Active Hot Parameters

Gain Compression and Gain Compression Converters

Differential IQ

IM Spectrum and IM Spectrum Converters

Swept IMD and Swept IMD Converters

Modulation Distortion and Modulation Distortion Converters

Noise Figure Cold Source and Noise Figure Converters

Phase Noise

Scalar Mixer/Converter + Phase and Vector Mixer/Converter

Spectrum Analyzer
<table>
<thead>
<tr>
<th>Preset</th>
<th>Main</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Preset</td>
</tr>
<tr>
<td></td>
<td>• User Preset ...</td>
</tr>
<tr>
<td></td>
<td>• Confirm Preset</td>
</tr>
</tbody>
</table>
### Save Recall

**Recall**

- Recall State
- Recall State
- Recall State...
- Recall Register
- Recall Calset...
- Recall Data...
- Recall Order

**Save State**

- Save State
- Auto Save
- Save State As...
- Save Register
- Save Type
- Delete State

**Save Other**

- Save Calset...
- Save Data...
- Save Screen...
- Save User Preset...
- Manage Files...
Scale

Main

- Autoscale
- Autoscale All
- Scale
- Reference Level
- Reference Position
- Y-Axis Spacing
- Scale Coupling
- Top
- Bottom
- Ref Y Level
- Ref Y Position
- Ref X Level
- Ref X Position

Electrical Delay

- Delay Time
- Delay Distance
- Distance Units
- Velocity Factor
- Media
- Wavegd Cutoff

Constants

- System Z0
- Phase Offset
- Mag Offset
- Mag Slope
Search

Main

- Max Search
- Min Search
- Search Range
- User Start
- User Stop
- Tracking
- Search...

Peak

- Peak Search
- Peak Right >> Search
- << Peak Left Search
- Next Peak Search
- Threshold
- Excursion
- Peak Polarity
- Tracking

Target

- Target Search
- Target Right >> Search
- << Target Left Search
- Target Value
- Transition
• Tracking

**Multi Peak & Target**

• Multi Peak Search
• Peak Threshold
• Peak Excursion
• Peak Polarity
• Multi Target Search
• Target Value
• Transition
• Tracking

**Bandwidth & Notch**

• Bandwidth Search
• BW Ref To
• BW Level
• Notch Search
• Notch Ref To
• Notch Level
• Tracking

**Comp & Sat**

• Compression Search
• Comp Level
• Saturation Search
• Pmax Backoff
• Tracking
Normal Op Pt

- Normal OP Search
- Backoff
- Pin Offset
- Tracking

Distortion (Option S9x070B Modulation Distortion only)

- ACPR Search
- NPR Search
- Tracking

Spurious (Option S93031xB Phase Noise only)

- Spurious Search
- Spurious Right >> Search
- << Spurious Left Search
- Multi Spurious Search
- Spur Sensibility
- Tracking
Setup

Standard
Active Hot Parameters
Gain Compression and Gain Compression Converters
Differential IQ
IM Spectrum and IM Spectrum Converters
Swept IMD and Swept IMD Converters
Modulation Distortion and Modulation Distortion Converters
Noise Figure Cold Source and Noise Figure Converters
Phase Noise
Scalar Mixer/Converter + Phase and Vector Mixer/Converter
Spectrum Analyzer
Sweep

Standard
Active Hot Parameters
Gain Compression and Gain Compression Converters
Differential IQ
IM Spectrum and IM Spectrum Converters
Swept IMD and Swept IMD Converters
Modulation Distortion and Modulation Distortion Converters
Noise Figure Cold Source and Noise Figure Converters
Phase Noise
Scalar Mixer/Converter + Phase and Vector Mixer/Converter
Spectrum Analyzer
System

Main

- Show Taskbar: Show the Windows Taskbar
- Move App to Back
- Minimize Application
- Exit: Exit the VNA application
- Security...
- Control Panel... : Open Windows Control Panel
- Manage Files... : Open Windows File Explore

System Setup

- Next/Prev Keys : Select Next/Previous window/channel/trace for the selected one
- Preferences...
- Sound: beep sound volume
- Remote Interface...
- LAN Status...
- Code Emulation

Print

- Print...
- Print to File...
- Page Setup...
- Print Colors...

Help
• NA Help...

• On The Web... : Connect to the Web on the keysight.com.

• Error Display...

• View Error Log...

• Tech Support
  
  o My Software Support...
  
  o My Support ID...
  
  o Licensed Features

• About NA...

Service

• Update Firmware

• Verification
  
  o Operator's Check
  
  o Performance Tests
  
  o System Verification
  
  o CalPod Check

• Adjustment Routines

• Diagnostics
  
  o Display Test
  
  o Front Panel: Check the front panel functionality.
  
  o Receiver Display
  
  o Receiver Temperature
  
  o EEPROM Headers
  
  o Mechanical Counter

  o SmartCal Save
- **Option Enable**

- **VNA Cal Status**: Opens the PathWave Calibration Advisor Status dialog:

  For information on how to set up the PathWave Calibration Advisor, click on ? in the top-right part of the dialog to access the PathWave Calibration Advisor Help.
TDR

- Setup
- TDR/TDT
- Eye/Mask
- More Functions tab
- Average tab
- Adv Waveform tab
- Hot TDR tab
- Trace Control
- Scale/Zooming
- Trigger Control
- Marker/Marker Search
- Data and Memory
- Gating
Trace

Trace 1-7/8-15

- Trace 1-15
- New Traces...

Trace Setup

- Select
- Measure
- Trace Title...
- Add Trace
- Delete Trace
- Trace Manager (Move Trace...)
- Trace Hold
Trigger

Main

- Hold
- Single
- Groups
- Continuous
- Manual Trigger
- Restart
- Trigger Source
- Trigger...
### Glossary

| A          | B          | C          | D          | E          | F          | G          | H          | I          | J          | K          | L          | M          | N          | O          | P          | Q          | R          | S          | T          | U          | V          | W          | X          | Y          | Z          |
|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|

12-Term Error Correction See Error Correction, 12-Term.

1-Port Device A device with a single connector or path to the device's circuitry. Examples include an oscillator and a load.

2-Port Calibration, Full See Error Correction, 12-Term.

2-Port Device A device with two connectors or other paths to the device's circuitry. Examples include filters, SAW devices, attenuators, matching pads, and amplifiers.

3-Term Error Correction See Error Correction, 3-Term.

A

**Active Channel** The highlighted channel affected by front panel functions.

**Active Function Readout** The area of a display screen where the active function and its state are displayed. The active function is the one that was completed by the last key selection or remote programming command.

**Active Hot Parameters (Option S9311xA/B)** Measures the optimum Gamma that provides the maximum delivered power for nonlinear devices.

**Active Marker** The marker on a trace that can be repositioned either by front panel controls or by programming commands.

**Active Trace** A trace that is being swept (updated) with incoming signal information.

**ADC** Analog to Digital Converter

**Address** The identification (represented by a name, label, or number) for a register, location in storage, or any other data source or destination. Examples are the location of a station in a communications network, or a device on the GP-IB.

**ADM** Add-Drop Multiplexer

**Admittance (Y)** The inverse of an impedance (i.e. the ratio of current to voltage). Complex
admittances take the form \( Y = G + jB(t) \).

**ALC** Automatic Level Control. See **Automatic Gain Control**.

**AM** Amplitude Modulation

**AM Group Delay** A technique for the measurement of group delay through a device which utilizes an amplitude modulated (AM) source. Note: The actual delay of the modulation envelope is measured directly with an external scalar detector. Devices that distort the amplitude of a signal cannot be measured. These include amplifiers with automatic gain control (AGC) and devices subject to saturation or power limiting.

**Amplitude Modulation** The process, or result of the process, of varying the amplitude of a carrier signal. The resulting modulated carrier contains information that can be recovered by demodulation. See also **Modulation**.

**Analog** The general class of devices or circuits in which the output varies as a continuous function of the input.

**Annotation** The labeling of specific information on the display (such as frequency or power).

**ANSI** American National Standards Institute: A national membership organization (open to manufacturers, organizations, users, and communications carriers) that approves standards, accredits standards development groups and certificate programs, and represents and coordinates US interests in non-treaty and non-government standards bodies.

**Aperture** The frequency span of the network analyzer used for calculating group delay. The narrower the aperture, the finer the resolution of the group delay variations, but noise is reduced by increasing the aperture.

**Array** A set of numbers or characters that represents any given function.

**ASCII** American Standard Code for Information Interchange

**Attenuation** Denotes a reduction in signal amplitude. The difference between transmitted and received power due to loss through equipment, lines, or other transmission devices; usually expressed in decibels.

**Attenuator** An RF or microwave device used to reduce the power level of a signal by precise, incremental amounts over its entire frequency range.

**Automatic Calibration System** AutoCal: Feature offered on Rohde&Schwarz network analyzers.

**Automatic Gain Control (AGC)** A circuit used in amplifiers and other active devices to keep its RF power level constant as other parameters change, such as frequency. Synonym: Automatic Leveling Control (ALC)
Autoscale An analyzer feature that evaluates waveforms and adjusts controls to stable and enhance the display.

AUX Auxiliary; refers to rear-panel input connector.

Averaging A noise reduction technique that computes each data point based on consecutive sweeps and weighted by a user-specified averaging factor. Each new sweep is averaged into the trace until the total number of sweeps is equal to the averaging factor.

B

B/R The ratio of data sampled at B to the data sampled at R.

Band Pass A range of frequencies that are passed through a device, such as a filter. Frequencies not within the band pass are limited or attenuated. See also Cutoff Frequency.

Bandwidth (BW) The difference between the frequencies of a continuous frequency band within which performance of a device falls within specifications.

Bandwidth Limit The condition prevailing when the system bandwidth is exceeded and signal distortion occurs beyond specifications.

Bandwidth Selectivity A measure of a filter's ability to resolve signals unequal in amplitude. It is the ratio of the 60 dB bandwidth to the 3 dB bandwidth for a given resolution filter (IF). Bandwidth selectivity tells us how steep the filter skirts are. Bandwidth selectivity is sometimes called shape factor.

Binary A method of representing numbers in a scale of two (on or off, high-level or low-level, one or zero). A compact, fast format used to transfer information to and from the analyzer.

BMP Bit-Mapped

Brightness See Color Brightness.

Broadband Device A device that operates over a very wide frequency range and exhibits only small variations in response over that range.

Buffer A storage device used when transmitting information to compensate for a difference in the rate of flow of information between two devices.

Burst Carrier A carrier that is periodically turned off and on. A burst carrier may or may not be modulated.

BUS Basic Utility System
**Bus** One or more conductors used as a path to deliver transmitted information from any of several sources to any of several destinations.

**BW** Bandwidth

**Byte** Eight bits of data representing one character processed as a unit.

---

**C**

**CAD** Computer Aided Design

**CAE** Computer Aided Engineering

**Calibration** In HP instrumentation, the process of periodically (usually annually) verifying an instrument is performing to specifications. A calibration certificate is awarded after verification.

In network analyzers, the process of removing systematic errors from measurements. See Error Correction.

**Calibration Kit** Hardware and software required to perform error correction on a network analyzer for a specific measurement and/or test set.

**Calibration, 2-Port** See Error Correction, 12-Term.

**Calibration, Blackburn** Calibrations of transmission path with corrected source match involving 15 calibration terms. Synonym: 15-term error correction

**Calibration, Frequency Response** The simplest error correction procedure to perform, but only corrects for a few of the twelve possible systematic error terms. Frequency response corrections can be made for reflection measurements, transmission measurements, and isolation measurements.

**Calibration, Interpolation** A user selectable network analyzer feature that calculates (interpolates) new error correction terms from existing terms when there is a change in network analyzer parameters, such as IF bandwidth, power, or sweep time. The resulting error correction is not as accurate as completing a full 2-port calibration.

**Calibration, Port Extension** See Port Extension.

**Calibration, Reference Plane** See Reference Plane.

**Calibration, Set Z** Sets the system impedance, usually 50 or 75 ohms.

**Calibration, SOLT** A calibration using four known standards: Short-Open-Load-Through. Also known as a full two-port calibration and 12-term error correction. See also Error Correction.
Calibration, TRL and LRM A calibration used in environments where the DUT cannot be connected directly to the network analyzer ports, (MMIC, microstrip, beam-lead diodes etc.). Thru-Reflect-Line (TRL) and M (Match) standards are fabricated and used because known high-quality standards are not readily available. The requirements for characterizing these standards are less stringent, but the calibration is not as accurate as the traditional full two-port calibration using S-O-L-T standards. The terms are used interchangeably (TRL, LRL, LRM etc.) but they all refer to the same basic calibration method.

Characteristic Impedance The impedance looking into the end of an infinitely long lossless transmission line.

Color Brightness A measure of the intensity (brightness) of a color.

Command A set of instructions that are translated into instrument actions. The actions are usually made up of individual steps that together can execute an operation.

Continuous Sweep Mode The analyzer condition where traces are automatically updated each time trigger conditions are met.

Controller A device capable of specifying the talker and listeners for an information transfer. An external computer connected to an instrument to control its operation.

Corrected Measurements made after performing error correction.

Coupler See Directional Coupler.

CPU Central Processing Unit

Crosstalk The occurrence of a signal at one port of a device being affected by a signal in any other path. Isolation is the measurement of crosstalk.

Cursor An electronically generated pointer that moves across the display to manipulate controls.

Cutoff Frequency In filters, the frequency at which attenuation is 3dB below the band pass signal level, known as the 3dB points.

CW Continuous wave: A single frequency (rather than a swept frequency).

D

DAC Digital to Analog Converter

dB Decibel: a relative unit of measure. The ratio in dB is given by: $10 \log_{10} (P_1/P_2)$ where $P_1$ and $P_2$ are the measured powers. The dB is preferred instead of arithmetic ratios or percentages because when components are connected in series, their effect on power, expressed in dB, may be arithmetically
added and subtracted. For example, if a 3dB attenuator is connected to a 10dB amplifier, the net gain of the two components is (-3dB + 10dB = +7dB).

dBm Absolute unit of measure in decibels: 0dBm = 1 mW. The conventions of the dB (adding and subtracting) continue to apply.

DBMS Database Management System

DC Direct Current

Default A known set of conditions used in the absence of user-defined conditions.

Delay See Group Delay.

Demodulation The process of recovering from a modulated carrier, information in the form of a signal having essentially the same characteristics as the original modulating signal. Recovery of the modulating signal accomplished by signal detection.

Detection The process of demodulating signal carriers. There are two basic ways of providing signal detection in network analyzers: Diode detectors (used in broadband applications) and heterodyning, (used in narrowband applications).

Detector, Diode A device used to convert a RF signal to a proportional DC level. If the signal is amplitude modulated, the diode strips the RF carrier signal from the modulation. Many sources used with scalar analyzers are amplitude modulated with a 27.778 kHz signal and then detected in the network analyzer. Phase information on the signal carrier is lost in diode detection.

Deviation from Linear Phase Linear phase refers to the nature of the phase shift of a signal through a device. The phase is linear if a plot of phase shift versus frequency is a straight line using linear scales. Deviation from linear phase causes signal distortion.

Digital Pertaining to the class of devices or circuits in which the output varies in discrete steps.

Digital Demodulation Describes a technique of extracting the information used to modulate a signal. Digital signal processing algorithms are used on the signal after it has been converted from an analog to a digital form (digitized).

Dimension To specify the size of an array. The number of array rows or columns.

Directivity In a 3-port directional coupler, the ratio of the power present at the auxiliary port when the signal is traveling in the forward direction to the power present at the auxiliary port when the same signal is traveling in the reverse direction.

Directional Coupler A 3-port device typically used for separately sampling the backward (reflected) wave in a transmission line.
**Disk** A circular, magnetic storage medium.

**Display** Noun: See Screen.

Verb: To show annotation and measurement data on the display.

**Display Detector Mode** The manner in which analog, video information is processed prior to being digitized and stored in memory.

**Display Dynamic Accuracy** The amplitude uncertainty, usually in dB, over the display dynamic range.

**Display Dynamic Range** The amplitude range, in dB, over which the display dynamic accuracy applies.

**Display Formats** Graphical formats for displaying measurement data. These include single channel, overlay (multiple traces on one graticule), split (each trace on separate graticules).

**Display Modes** The ways in which measurement data can be presented graphically. On a network analyzer, the choices are Cartesian/rectilinear (XY plot with log or linear magnitude, phase, group delay, SWR, real and imaginary, and dBV, dBmV and dBuV), polar (magnitude and angle), magnitude and phase, and Smith chart. Not all display modes are available on all network analyzers. In addition, displays can present this information in various combinations of traces. Common modes are dual, (the ability to display more than one trace, usually over the same frequency range), and alternate, (the ability to display more than one trace, each with different frequency range and type).

**Display Phase Dynamic Accuracy** The phase measurement uncertainty, usually in degrees, for measurements whose units are in degrees.

**Display Points** The total number of measurement points made in a single measurement. The points can be in units of frequency, power, or time. The number of points often dictates measurement speed, resolution, and aperture.

**Display Trace Noise, Magnitude** The amplitude uncertainty of the trace, in dB, due to random noise in the test system.

**Display Trace Noise, Phase** The phase uncertainty of the trace, in degrees, due to random noise in the test system.

**Display Type** The type of display screen built into the analyzer. Data can be displayed as a raster drawing (a computer-like dot map) or as a vector drawing (lines drawn on the display). Color and display standard can also be specified as monochrome (single color), or color (two or more colors). The format standard may also be specified, such as VGA or SVGA, for IBM-compatible personal computers.

**Distortion** Deterioration of a signal's quality due to the nonlinear characteristics of a device or system.
transfer function. Distortion is measured as a combination of the changes in amplitude, frequency and phase of signal at the output of a device or system as compared to the signal at the input.

**Drift** The slow change in signal frequency.

**DSP** Digital Signal Processing

**DUT** Device Under Test

**DVM** Digital Volt Meter

**Dynamic Range** In a receiver, the range of signal levels, from minimum to maximum, that can be reliably measured simultaneously. Dynamic range allows small signals to be measured in the presence of large signals. Source power and receiver compression usually limits the maximum boundary to dynamic range. Receiver residual responses and noise floor usually limit the minimum power boundary.

---

**E**

**ECal** See **Electronic Calibration**.

**Electrical Delay** A simulated variable length of lossless transmission line, added to or subtracted from a receiver input, to compensate for interconnecting cables. The firmware equivalent of mechanical or analog "line stretchers" in other network analyzers.

**Electronic Calibration (ECal)** A calibration system for electronic calibration of RF and microwave vector network analyzers. The electronic calibration system creates a twelve-term, two-port error model and then provides a confidence check of the calibration. The Ecal system consists of a repeatable, variable-impedance, solid-state calibration standard and a mainframe control unit which interfaces with the 8510, 8720 series, and the 8753 network analyzers or a USB module which interfaces with the PNA series network analyzers.

**EMC** Electro-Magnetic Compatibility

**EMI** Electro-Magnetic Interference: Unintentional interfering signals generated within or external to electronic equipment. Typical sources could be power-line transients, noise from switching-type power supplies and/or spurious radiation from oscillators. EMI is suppressed with power-line filtering, shielding, etc.

**Engage** To activate a function.

**Enter** The process of inputting information.

**EPROM** Electronically Programmable, Read-Only Memory
**Error Correction** In network analyzers, a process that removes or reduces systematic (repeatable) measurement errors by measuring known standards from a calibration kit. Synonym: measurement calibration

**Error Correction, 3-Term** Used to remove systematic measurement errors on a device with one port, such as a load.

**Error Correction, 12-Term** Correction for a two port device using six parameters:
- Directivity
- Source match
- Load match
- Reflection frequency response
- Transmission frequency response
- Isolation

To completely characterize a two-port device, these six parameters must be characterized in the forward and reverse directions, making a total of 12 terms. The user usually has the option of omitting isolation from the correction process. Synonym: Full two-port error correction

**Error Correction, 1-Port** Corrects a test set for port 1 or port 2 directivity, frequency response, and source match errors. The process requires three known standard terminations, for example, open, short, and load.

**Error Message** A message on a display that indicates an error condition. Missing or failed hardware, improper user operation, or other conditions that require additional attention can cause an error condition. Generally, the requested action or operation cannot be completed until the condition is resolved.

**ESD** Electro Static Discharge

**Ethernet** A network that adheres to the IEEE 802.3 Local Area Network standard.

**Ethernet address** A hexadecimal number which is used to identify a machine on a network. Each analyzer is assigned a unique Ethernet address at the factory and it is stored in the analyzer's ROM.

**External trigger signal** A TTL signal that is input to an analyzer and initiates a measurement sweep or similar event, making the measurements synchronous with the external triggering source.

**F**

**Filter** A passive device that allows some frequencies to pass and attenuates others, depending on the type and specifications. A high-pass filter passes frequencies above the cutoff frequency, a low-pass filter passes frequencies below the cutoff frequency, and a band-pass filter passes frequencies between
two specific frequencies.

**Firmware** An assembly made up of hardware and instruction code. The hardware and instruction code is integrated and forms a functional set that cannot be altered during normal operation. The instruction code, permanently installed in the circuitry of the instrument, is classified as ROM (read only memory). The firmware determines the operating characteristics of the instrument or equipment.

**Flatness** The amplitude and phase response of a device under test (DUT), a signal source, a receiver, or a combination of these. See also Frequency Response.

**FM** Frequency Modulation

**Frequency** The number of periodic oscillations, vibrations, or waves per unit of time, usually expressed in cycles per second, or Hertz (Hz).

**Frequency Accuracy** The uncertainty with which the frequency of a signal or spectral component is indicated, either in an absolute sense or relative to another signal or spectral component. Absolute and relative frequency accuracies are specified independently.

**Frequency Range** The range of frequencies over which a device or instrument performance is specified.

**Frequency Resolution** The ability of a network analyzer to measure device characteristics at closely spaced frequencies and display them separately. Resolution of equal amplitude responses is determined by IF bandwidth. Resolution of unequal amplitude responses is determined by IF bandwidth and bandwidth selectivity.

**Frequency Response** The peak-to-peak variation in the displayed amplitude response over a specified center frequency range. Frequency response is typically specified in terms of dB, relative to the value midway between the extremes.

**Frequency Span** The magnitude of the displayed frequency component. Span is represented by the horizontal axis of the display. Generally, frequency span is given as the total span across the full display. Some analyzers represent frequency span (scan width) as a per-division value.

**Frequency Stability** The ability of a frequency component to remain unchanged in frequency or amplitude over short and long-term periods of time. Stability refers to an oscillator's ability to remain fixed at a particular frequency over time.

**Front Panel Key** Keys that are located on the front panel of an instrument. The key labels identify the function the key activities. Numeric keys and step keys are two examples of front panel keys.

**Full 2-Port Calibration** See Error Correction, 12-Term.

**Function** The action or purpose that a specific item is intended to perform or serve. The network
analyzer contains functions that can be executed via front panel key selections, or through programming commands. The characteristics of these functions are determined by the firmware in the instrument. In some cases, a DLP (downloadable program) execution of a function allows you to execute the function from front panel key selections.

**Fundamental Frequency** In any waveform, the lowest frequency component; all other components are harmonics. A pure sinusoid has only one component, the fundamental.

---

**G**

Gb Gigabit

GB Gigabyte

GHz Gigahertz

GIF Graphics Interchange Format - Standard graphic format to store bitmapped graphics files.

Giga Prefix for one billion.

GP I/O General Purpose Input / Output; a connector usually on the back of an instrument that allows communication with other test equipment, external test sets, switches, and computers that enable the instrument to be triggered or to trigger external equipment. An example is a foot switch that continues or cycles a measurement, allowing the operator to use both hands on the test hardware.

GPIB General Purpose Interface Bus - IEEE 488 bus is interconnect bus and protocol, allows linking of instruments and computer.

Graticule (or Grid) Enclosed area where waveform is displayed on instrument. Tick marks, on frame or axis, are a scaling aid for making visual measurements.

Group Delay A measure of the transit time of a signal through a DUT versus frequency. Group delay can be calculated by differentiating the DUT's insertion-phase response with respect to frequency. See also **AM Group Delay** and **Deviation from Linear Phase**.

GUI Graphical User Interface

---

**H**

Hardcopy Paper copy of data.

Hardkey A front-panel key, which engages a single analyzer function or presents a single menu of softkeys.
Horizontal Reference  See Reference Level.

Horizontal Resolution The analyzer's ability to take closely spaced horizontal data points over the full sweep.

Host Computer  A computer or device on a network that provides end users with services such as computation and database access and that usually performs network control functions.

Host Name  A unique name that is used to identify each host machine on a network. The host name is directly linked to, and can usually be used in place of, the IP address. The user or the system administrator usually creates the host name.

HP  Hewlett-Packard Company

HPGL  Hewlett-Packard Graphics Language

HP-IB  Hewlett-Packard Interface Bus. A parallel interface that allows "daisy chaining" of more than one device to a port on a computer or instrument. Interface protocol is defined in IEEE 488.2; equivalent to the industry standard GPIB.

HTTP  HyperText Transfer Protocol: Used to carry World Wide Web (WWW) traffic.

Hue  The dimension of color referred to a scale of perceptions ranging from red through yellow, green, and blue, and back to red. A particular gradation of color, tint, shade.

I

I/O Input/Output

I/O Path  Input/Output Path

IEEE  Institute of Electrical and Electronic Engineers

IF  Intermediate Frequency: the frequency at which a signal is processed after mixing.

Impedance  The ratio of voltage to current at a port of a circuit, expressed in ohms.

Initialize  The process that assigns information locations to a disk to prepare the magnetic media to accept files.

Input  A path intended for putting a signal into an instrument.

Most network analyzers have either 3 (labeled A, B, and R) or 4 inputs (labeled A, B, R1, and R2). Inputs are not the same as channels.

Input Attenuator  An attenuator between the input connector and the first mixer of a spectrum
analyzer (also called an RF attenuator). The input attenuator is used to adjust the signal level incident to
the first mixer, and to prevent gain compression due to high-level or broadband signals. It is also used
to set the dynamic range by controlling the degree of internally-generated distortion. For some
analyzers, changing the input attenuator settings changes the vertical position of the signal on the
display, which then changes the reference level accordingly. In Keysight microprocessor-controlled
analyzers, the IF gain is changed to compensate for changes in input attenuator settings. Because of
this, the signals remain stationary on the display, and the reference level is not changed.

**Insertion Loss** The difference between the power measured before and after the insertion of a device.
The attenuation between the input and output of a device.

**Intensity** Brightness; emitting or reflecting light; luminosity.

**Interface** A connection that allows a common communication link between two or more instruments.

**Intermodulation Distortion** Undesired frequency components resulting from the interaction of two or
more spectral components passing through a device having nonlinear behavior, such as a mixer or an
amplifier. The undesired components are related to the fundamental components by sums and
differences of the fundamentals and various harmonics. The algorithm is: \( f_1 \pm f_2, 2xf_1 \pm f_2, 2xf_2\pm f_1, \)
\(3xf_1 \pm 2x f_2, \) and so on.

**Internet** The connection of two or more distinct networks. Often a gateway or router is used to make
the connection.

**Interpolate** To determine a value of a signal between two adjacent points by a procedure or algorithm.

**IP** Internet Protocol

**IP Address** Internet protocol address: a unique number that is assigned to each device which is to be
connected to a TCP/IP network. Before using an analyzer on a network, your network administrator
will need to assign an IP address. An IP address consists of a 32-bit value presented in decimal dot
notation: 4 octets (bytes) separated by a dot.

**ISDN** Integrated Services Digital Network: A standard digital service capability that features one or
more circuit-switched communication channels capable of carrying digital voice, data, or image
signals, a packet-switched channel for out-of-band signaling and control. In addition, ISDN provides a
collection of standard and optional features that support information productivity for the user,
providing higher-speed Internet access than analog systems.

**ISO** International Standards Organization

**Isolation** A specification or measure of the immunity that one signal has to being affected by another
adjacent signal. The occurrence is known as crosstalk.

**Isolator** An RF device used for providing isolation between paths and components. Made from a 3-port
circulator, the third port being terminated in a 50ohm load.

---

J

---

K

Kilo Prefix for one thousand.

KB Kilobyte

Kb/s Kilobytes per second

---

L

LAN Local Area Network

LANS Local Area Network System

LCD Liquid Crystal Display

LED Light Emitting Diode

LFE Low Frequency Extension

LIF Logical Interchange Format (used for older HP disk drives/computers)

Limit Lines Lines input by the user that overlay the analyzer's measurement data to allow automatic detection of data that is out of the acceptable range. Pass/Fail annotation, audio alarms, or electronic output can be triggered to notify the operator or on-line computer program of the over-limit condition.

Limit-Line File The user-memory file that contains the limit-line table entries.

Limit-Line Table The line segments of a limit line are stored in the limit-line table. The table can be recalled to edit the line segments, then restored in the limit-line file.

Linear Device A device in which the output is continuously proportional to the input.

LO Local Oscillator. In a superheterodyne system, the LO is mixed with the received signal to produce a sum or difference equal to the intermediate frequency (IF) of the receiver.

LO Feedthrough The response that in a superheterodyne system when the first local oscillator frequency is equal to the first IF.

Load A one port microwave device used to terminate a path in its characteristic impedance.
**Load Match** A measure of how close the device's terminating load impedance is to the ideal transmission line impedance. Match is usually measured as return loss or standing wave ratio (SWR) of the load.

**Local Lock Out** A condition or command that prevents analyzer front-panel entries (and disables the Local key).

**Local Operation** To operate manually from the front panel.

**Log** Logarithm

**Log Display** The display mode in which vertical deflection is a logarithmic function of the input signal amplitude. Log display is also called logarithmic display. The display calibration is set by selecting the value of the reference level position and scale factor in dB per division.

**LRM** Line-Reflect-Match. See Calibration, TRL, and LRM.

---

**M**

**Magnitude** The amplitude of a signal measured in its characteristic impedance without regard to phase. See also Scalar.

**Marker** A graphical symbol along a display trace that is annotated with measurement characteristics of that specific data point.

**Marker Functions** Mathematical or statistical computation on the data of one or more markers to provide the operator more information. For example, the marker delta function calculates and displays the difference between two markers.

**Maximum Input Level** The maximum signal power that may be safely applied to the input of an analyzer. The maximum input level is typically 1 W (+30 dBm) for Keysight spectrum analyzers.

**MB** Megabyte

**Measurement Uncertainty** The quantified amount of error in a measurement situation. Calibrations are intended to reduce the amount of uncertainty. The following are sources of measurement errors that lead to uncertainty:

- Systematic errors (imperfections in calibration standards, connectors, cables, and instrumentation)
- Random errors (noise, connector repeatability)
- Drift (source and instrumentation)
**Mega** Prefix for one million.

**Memory** A storage medium, device, or recording medium into which data can be stored and held until some later time, and from which the entire original data may be retrieved.

**Memory Card** A small memory device shaped like a credit card that can store data or programs.

**Menu** The analyzer functions that appear on the display and are selected by pressing front panel keys. These selections may invoke a series of other related functions that establish groups called menus.

**MHz** Megahertz

**milli** Prefix for one-thousandth.

**Modem** Modulator/Demodulator

**Modulation** The process, or the result of the process, of varying a characteristic of a carrier signal with an information-bearing signal, causing the carrier to contain the information. See AM and FM.

**Modulation Distortion (Option S93070xB)** Measures the nonlinear behavior of an RF microwave amplifier under a modulated signal.

**Monitor** Any external display.

**Monochrome** Having only one color (chromaticity).

**ms** Millisecond

**mW** Milliwatt: one thousandth of a watt

**Multisync** A type of monitor that can synchronize its horizontal sweep to various frequencies within a specified range.

---

**N**

**Narrowband** In network analysis, the frequency resolution of the analyzer's receiver that is sufficiently narrow to resolve the magnitude and phase characteristics of narrowband devices. The reduced receiver bandwidth usually decreases the noise floor of the receiver, providing more measurement amplitude range.

**Narrowband Device** A device whose transfer characteristics are intended to operate over a very narrow frequency range and are designed to provide well-defined amplitude responses in that range, such as a band pass filter.

**Network Analysis** The characterization of a device, circuit, or system derived by comparing a signal
input going into the device to a signal or signals coming out from the device.

**NIST National Institute of Standards and Technology**

**Nit** The unit of luminance (photometric brightness) equal to one candela per square meter.

**Noise** Random variations of unwanted or disturbing energy in a communications system from man-made and natural sources that affects or distorts the information carried by the signal. See also Signal-to-Noise Ratio.

**Noise Figure (F):** For a two-port device, a measure of how the noise generated inside the device degrades the signal-to-noise ratio of a signal passing through the device at 290 degrees, usually expressed in dB.

**Noise Floor** The analyzer's internal displayed noise. The noise level often limits how small a signal magnitude can be measured. In network analysis, noise floor is measured with the test ports terminated in loads, full two-port error correction, 10 Hz IF bandwidth, maximum test port power, and no averaging during the test.

**Non-Insertable Devices** In measurement calibration, a device that cannot be substituted for a Zero-Length Through Path. It has the same type and sex connectors on each port, or a different type of connector on each port.

**Nonvolatile Memory** Memory data that is retained in the absence of an ac power source. This memory is typically retained with a battery. Refer also to battery-backed RAM.

**Normalize** To subtract one trace from another to eliminate calibration data errors or to obtain relative information.

---

**O**

**Offset** To move or set off a determined amount. Used in instruments for offsetting frequencies, limits, delay, loss, impedance, etc.

**Output Attenuation** The ability to attenuate the signal, the source, in order to control its power level.

---

**P**

**PC Personal Computer**

**PDF Portable Document Format (used on the Web)**

**Parser, Command** Reads program messages from the input queue of a device in the order they were
received from the controller. The parser determines what actions the analyzer should take. One of the most important functions of the command parser is to determine the position of a program message in the analyzer SCPI command tree. When the command parser is reset, the next element it receives is expected to arise from the base of the analyzer command tree.

**Peak Search** A function on an analyzer that searches for the largest response and places a marker on it.

**Phase** The fractional part of a cycle through which an oscillation has advanced, measured from an arbitrary starting point; usually measured in radians or degrees. In network analysis, the phase response of the device under test is the change in phase as a function of frequency between the input stimulus and the measured response.

**Port** The physical input or output connection of an instrument or device.

**Port Extension** Redefining the reference plane to other than that established at calibration. A new reference plane is defined in seconds of delay from the test set port.

**Positive Peak** The maximum, instantaneous value of an incoming signal.

**Postscript (.ps files)** Stores bitmapped graphics files in an encapsulated format for direct use by postscript printers.

**Power, Max Input** The upper limit to input power for which the specifications apply. Some specifications may have different levels of maximum inputs. For example, compression power maximum is usually higher than the harmonic distortion maximum.

**Power, Safe Input** The input power, usually in dBm, allowed without damaging the instrument.

**Preset** A pre-defined instrument state (that also runs an analyzer self-test). The action of pushing the Preset key.

**Protocol** A set of conventions that specify how information will be formatted and transmitted on a network, and how machines on a network will communicate.

---

**Q**

**Q or Q Factor** The ratio of energy stored to energy lost in a resonant circuit. High Q indicates a sharp resonance response over frequency.

**Query** Any analyzer programming command having the distinct function of returning a response. These commands may end with a question mark (?). Queried commands return information to the computer.
Expression for complex impedance, where $r$ represents the resistive portion and $x$ represents the reactive portion.

**R Channel** Reference Channel

**RAM** Random Access Memory, or read-write memory: A storage area allowing access to any of its storage locations. Data can be written to or retrieved from RAM, but data storage is only temporary. When the power is removed, the information disappears. User-generated information appearing on a display is RAM data.

**ROM** Read Only Memory: A storage area that can be read only; it cannot be written to or altered by the user. In instruments, the storage area that contains the "brains" or operational programming; the firmware.

**Receiver** A circuit or system designed for the reception and/or measurement of signals in a specified frequency spectrum.

**Receiver Dynamic Range** See Dynamic Range.

**Reference Level** An instrument function that allows the user to set the amplitude value at the reference position. On network analyzers, the reference position is also selectable. On some spectrum analyzers, the reference position is fixed at the top of the display.

**Reference Plane** The electrical location at which a network analyzer assumes the system connectors and fixturing ends and the DUT begins. The reference plane is set by using calibration standards with known electrical length. The closer the reference plane is to the device under test (DUT), the better the characterization of the device because of the elimination of test system uncertainties.

**Reference Receiver** In a network analyzer, the receiver that measures signals as they come out of the source, before they are incident on the test port and DUT. Typically, these signals are used to compare with the signal at the Test Port Receiver, to determine the affect that the DUT has on the signal. In a 2-port network analyzer, these are typically named 'R1' (port 1) and 'R2' (port 2). See a block diagram of the receivers in your PNA.

**Reflection** The phenomenon in which a traveling wave strikes a discontinuity and returns to the original medium.

**Reflection Coefficient** The ratio of the reflected voltage to the incident voltage into a transmission line or circuit. If a transmission line is terminated in its characteristic impedance, the reflection coefficient is zero. If the line is shorted or open the coefficient is 1. See also Return Loss and SWR.

**Reflection Measurements** Measurements that characterize the input and/or output behavior of the device under test (DUT). Measured as the ratio of the reflected signal to the incident signal as a
function of frequency. Parameters are called return loss, reflection coefficient, impedance, and standing wave ratio (SWR), all as a function of frequency. See also S-Parameters.

Remote A mode of operation where another device (or computer) controls an instrument via the HP-IB. In this mode, the instrument front panel keys are disabled. Front panel operation is called local operation.

Remote Programming The automatic operation of an instrument by a computer, usually through a HP-IB, LAN, or RS-232 link.

Resolution The ability of a receiver to resolve two signals.

Resolution Bandwidth The ability of a spectrum analyzer to display adjacent responses discretely (Hertz, Hertz decibel down). This term is used to identify the width of the resolution bandwidth filter of a spectrum analyzer at some level below the minimum insertion loss point (maximum deflection' point on the display). Typically, it is the 3 dB resolution bandwidth that is specified, but in some cases the 6 dB resolution bandwidth is specified.

Return Loss The amount of dB that the reflected signal is below the incident signal. If zero signal is reflected, the impedance of the device is equal to the characteristic impedance of the transmission system, and return loss is infinite. If the entire incident signal is reflected, the return loss is zero. See also S-Parameters, Reflection Coefficient, and SWR.

Reverse Measurement The measurement of a device from output to input.

RF Radio Frequency (from approximately 50 kHz to approximately 3 GHz). Usually referred to whenever a signal is radiated through the air.

ROM Read Only Memory

S

S/N Signal-to-Noise Ratio

Sampler An electronic component that captures the signal level and phase across a known impedance at a uniform rate. In Network Analyzers, this sampling rate must be sufficiently high and precisely timed to make accurate measurements. Network analyzers typically have three or four samplers or mixers.

Sampler Bounce The leakage or crosstalk between a network analyzer's samplers. Delay in this crosstalk caused by leakage transmission propagation, give the interference its "bounce" appearance. Sampler bounce causes an increase in the noise level of the affected channel, reducing the sensitivity of the analyzer.
**Saturation** The degree of color purity, on a scale from white to pure color.

**Scalar** A quantity that has magnitude but no phase. A network analyzer capable of measuring only magnitude.

**Scale Factor** The display vertical axis calibration in terms of units per division.

**SCPI** Standard Commands for Programmable Instruments

**Screen** The physical surface of the CRT or flat panel upon which the measurement results, setup information, softkey definitions, and other instrument communication is presented.

**Self-Test** A group of tests performed at power-up (or at preset) that verify proper instrument operation.

**Sensitivity** The minimum input signal required to produce a specified output signal having a specified signal-to-noise ratio, or other specified criteria.

On a spectrum analyzer, the level of the smallest sinusoid that can be observed, usually under optimized conditions of minimum resolution bandwidth, 0 dB input attenuation, and minimum video bandwidth.

The normalized change in YIG component's center frequency resulting from a change in tuning coil current, specified in MHz/mA.

**Serial Prefix** The five-character prefix that begins an instrument serial number; used to represent versions of firmware or hardware changes that have occurred.

**Server** A device that is configured to provide a service to other devices on a network, such as shared access to a file system or printer.

**Signal-to-Noise Ratio** SNR: The ratio of the amplitude of the desired signal to the amplitude of noise signals, usually expressed in dB and in terms of peak values for impulse noise and root-mean-square values for random noise.

**Single Sweep Mode** The spectrum analyzer sweeps once when trigger conditions are met. Each sweep is initiated by pressing an appropriate front panel key, or by sending a programming command.

**Small Signal Gain Compression** A situation when the input signal's measured amplitude is less than its actual level due to overloading of the network analyzer's input mixer; the analyzer is operating nonlinearly. For broadband analyzer detectors, a signal other than the one under test can put the analyzer into this gain compressed mode, thereby making even lower level signals appear at a lower level than actual. The broadband mode measures all the power incident to the analyzer, not just the signals at the frequency of interest.

**Smith Chart** A graphical mapping of the complex reflection coefficient into normalized complex impedance. Circles on the chart represent constant resistance and radiating lines orthogonal to the
circles represent constant reactance. The center of the chart represents the characteristic impedance of the transmission system. Any point on the chart defines a single complex impedance. A line on the chart represents changing impedance over frequency.

SOLT Short-Open-Load-Through calibration. See also Calibration, SOLT.

Source A device that supplies signal power; a sweep oscillator or synthesized sweeper.

Source Amplitude Accuracy The amplitude uncertainty, in dB, of the source power readout.

Source Amplitude Flatness The amplitude flatness, in dB, of the source power over the frequency range specified.

Source Frequency Resolution The smallest unit of frequency which can be set and/or measured, in Hz.

Source Frequency Time Base Accuracy A measure of the analyzer's frequency stability measured in parts per million (ppm. or 1 part in 10E6). For example, a stability of ±5.0 ppm means that an analyzer will measure 1 MHz to an accuracy of ±5 x 10-6 x 10E6 Hz = ±5 Hz.

Source Frequency Time Base Stability A measure of the analyzer's time base accuracy over time and temperature. Typically the time base accuracy will be specified for 1 year. A typical temperature frequency stability is ±10 ppm for 25° C ± 50 C.

Source Harmonics The level of harmonics generated by the analyzer's signal source, in dBc from the fundamental.

Source Match A measure of how close the signal source impedance is to the ideal transmission line impedance of the test system. Match is usually measured as return loss or standing wave ratio (SWR) of the source.

Span The stop frequency minus the start frequency. The span setting determines the horizontal-axis scale of the analyzer display.

Span Accuracy The uncertainty of the indicated frequency separation of any two signals on the display.

S-Parameters (Scattering Parameters) A convention used to characterize the way a device modifies signal flow using a network analyzer. A two port device has four S-parameters: forward transmission (S21), reverse transmission (S12), forward reflection (S11), and reverse reflection (S22).

Spectrum Analyzer (Option Sx090A/B) Spectrum Analyzer function for component measurements.

Stop/Start Frequency Terms used in association with the stop and start points of the frequency measurement range. Together they determine the span of the measurement range.
Storage States The number of settings, programs, traces, and other parameters available to be saved, cataloged, and recalled at any one time.

Storage, Disk An internal or external digital storage disk for saving test data, instrument settings, IBASIC programs, and other measurement parameters. Storage formats include MS-DOS (R) and HPs standard LIF with binary, PCX, HP-GL, or ASCII data formats.

Structural Return Loss Poor return loss in cable due to a periodic fault such as a periodic dent caused by dropping the cable spool or by the cable pulling process during manufacture.

Supplemental Characteristics Typical but non-warranted performance parameters, denoted as "typical", "nominal" or "approximate".

Sweep The ability of the source to provide a specified signal level over a specified frequency range in a specified time period. Also see Sweep Mode and Sweep Type.

In data processing mode, a series of consecutive data point measurements, taken over a sequence of stimulus values.

Sweep Mode The way in which a sweep is initiated or selected, e.g., single, continuous, alternate, or chopped.

Sweep Type The method of sweeping the source, e.g., linear, log, or frequency step.

Sweeper A signal source that outputs a signal that varies continuously in frequency.

SWR Standing Wave Ratio, calculated as \((1 + \pi) / (1 - \pi)\) where \(\pi\) is the reflection coefficient.

Sync Synchronization, or Synchronized

Syntax The grammar rules that specify how commands must be structured for an operating system, programming language, or applications.

System Dynamic Range The difference between the maximum receiver input level and the receiver's noise floor. System dynamic range applies to transmission measurements only, since reflection measurements are limited by directivity.

TDR/TDT (Option S93011A/B) Time Domain Reflectometry/Time Domain Transmission.

T/R See Transmission/Reflection.

Termination A load connected to a transmission line or other device.
**Test Limit** The acceptable result levels for any given measurement.

**Test Port** See Port.

**Test Port Receiver** In a network analyzer, the receiver directly behind the test ports, used to measure the signal as it is reflected off, or transmitted through, the DUT. This signal is typically compared with the signal at the Reference Receiver to determine how the DUT affects a signal. In a 2-port network analyzer, these are typically named 'A' (port 1) and 'B' (port 2). See a block diagram of the receivers in your PNA.

**Test Set** The arrangement of hardware (switches, couplers, connectors and cables) that connect a test device input and output to the network analyzer's source and receiver to make s-parameter measurements.

**Third Order Intercept** TOI: The power input to a non-linear device that would cause third order distortion at the same power level. TOI is a measurement to determine the distortion characteristics of a mixer or receiver. The higher the value, the more immune the receiver to internal distortion.

**Thru** Through line: A calibration standard. See Calibration, SOLT.

**Tint** A shade of color; hue.

**Toggle** To switch states, usually to change a function from on to off, or off to on.

**TOM** Thru-Open-Match: A Rohde&Schwarz term to describe a calibration method.

**Trace** A series of data points containing frequency and response information. The series of data points is often called an array. The number of traces is specific to the instrument.

**Tracking** The ability of the analyzer's receiver to tune to the source frequency over the measurement frequency range. Poor tracking results in amplitude and phase errors due to the receiver IF circuits attenuating and delaying the device under test output.

**Transfer Function** The ratio of the output signal to the stimulus signal, both as a function of frequency.

**Transmission** See Transmission Measurements.

**Transmission Intermodulation Spurious** A measure of the capability of the transmitter to inhibit the generation of intermodulation distortion products. Intermodulation spurious is sometimes called intermodulation attenuation.

**Transmission Measurements** The characterization of the transfer function of a device, that is, the ratio of the output signal to the incident signal. Most common measurements include gain, insertion loss, transmission coefficient, insertion phase, and group delay, all measured over frequency. See also S-Parameters.
**Transmission/Reflection (T/R)** Refers to the suite of measurements made by a scalar or vector network analyzer to characterize a device's behavior over frequency. See also S-Parameters.

**Transparent** Something that is not visible to the user. Usually a procedure that occurs without the user's initiation or knowledge.

**Trigger** A signal that causes the instrument to make a measurement. The user can select several options for triggering, such as manual, continuous, or external (for synchronizing measurements to an external source).

**TRL** Through-Reflect-Line. See Calibration, TRL and LRM.

**TTL** Transistor-Transistor Logic

**Two-Port Error Correction** See Error Correction, 12-Term.

**U**

**Uncorrected** Measurements made without performing error correction.

**Uncoupled Channels** Stimulus or receiver settings allowed to be set independently for each channel.

**UNI** User-Network Interface: The point at which users connect to the network.

**Units** Dimensions on the measured quantities. Units usually refer to amplitude quantities because they can be changed. In analyzers with microprocessors, available units are dBm (dB relative to 1 mW dissipated in the nominal input impedance), dBmV (dB relative to 1 mV), dBW (dB relative to 1 1W), V (volts), W (watts).

**V**

**Variable** A symbol, the value of which changes either from one iteration of a program to the next, or within each iteration of a program.

**Vector** A quantity that has both magnitude and phase.

A network analyzer capable of measuring both magnitude and phase.

**VEE** Visual Engineering Environment (Keysight software product)

**Velocity Factor** A numerical value related the speed of energy through transmission lines with different dielectrics (.66 for polyethylene). Used in making time domain measurements.

**Vertical Resolution** The degree to which an instrument can differentiate amplitude between two
signals.

**Video** An electrical signal containing timing, intensity, and often color information that, when displayed, gives a visual image.

**Video Bandwidth** In spectrum analyzers, the cutoff frequency (3 dB point) of an adjustable low-pass filter in the video circuit. When the video bandwidth is equal to or less than the resolution bandwidth, the video circuit cannot fully respond to the more rapid fluctuations of the output of the envelope detector. The result is a smoothing of the trace, or a reduction in the peak-to-peak excursion, of broadband signals such as noise and pulsed RF when viewed in broadband mode. The degree of averaging or smoothing is a function of the ratio of the video bandwidth to the resolution bandwidth.

**Video Filter** In spectrum analyzers, a post-detection, low-pass filter that determines the bandwidth of the video amplifier. It is used to average or smooth a trace. Refer also to Video Bandwidth.

**VNA** Vector Network Analyzer

---

**W**

**Waveform** A representation of a signal plotting amplitude versus time.

**Wireless** A term that refers to a broad range of technologies that provide mobile communications for home or office, and "in-building wireless" for extended mobility around the work area, campus, or business complex. It is also used to mean "cellular" for in-or out-of-building mobility services.

**WWW** World Wide Web

---

**X**

---

**Y**

---

**Z**

**Zero-Length Through Path** In a measurement calibration, when the two test cables mate together directly without using adapters or a thru-line. See also Non-Insertable Devices.
An internet connection is required to view ALL specifications documents online at [www.keysight.com](http://www.keysight.com).

<table>
<thead>
<tr>
<th>Doc Number</th>
<th>Model</th>
<th>PNA-X Models</th>
<th>N5242-90027</th>
<th>N5241B (13.5 GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>N5242B (26.5 GHz)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>N5249B (8.5 GHz)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>N5244B (43.5 GHz)</td>
<td></td>
</tr>
<tr>
<td>N5245-90028</td>
<td></td>
<td></td>
<td>N5245B (50 GHz)</td>
<td></td>
</tr>
<tr>
<td>N5247-90029</td>
<td></td>
<td></td>
<td>N5247B (67 GHz)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>N523x Models</th>
<th>N5234B (10 MHz to 43.5 GHz - 2-port ONLY)</th>
</tr>
</thead>
<tbody>
<tr>
<td>N5235B (10 MHz to 50.0 GHz - 2-port ONLY)</td>
<td></td>
</tr>
<tr>
<td>N5239B (300 kHz to 8.5 GHz - 2-port ONLY)</td>
<td></td>
</tr>
<tr>
<td>N5231B (300 kHz to 13.5 GHz - 2-port / 4-port)</td>
<td></td>
</tr>
<tr>
<td>N5232B (300 kHz to 20.0 GHz - 2-port / 4-port)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>N522xB Models</th>
<th>N5221B (13.5 GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>N5222B (26.5 GHz)</td>
<td></td>
</tr>
<tr>
<td>N5224B (43.5 GHz)</td>
<td></td>
</tr>
<tr>
<td>N5225B (50 GHz)</td>
<td></td>
</tr>
<tr>
<td>N5227B (67 GHz)</td>
<td></td>
</tr>
</tbody>
</table>
### N522xB "Metrology" Models

<table>
<thead>
<tr>
<th>Opt 210/410</th>
<th>Model and Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>N5221-90004</td>
<td>N5221B (13.5 GHz)</td>
</tr>
<tr>
<td></td>
<td>N5222B (26.5 GHz)</td>
</tr>
<tr>
<td>N5224-90004</td>
<td>N5224B (43.5 GHz)</td>
</tr>
<tr>
<td></td>
<td>N5225B (50 GHz)</td>
</tr>
<tr>
<td>N5227-90006</td>
<td>N5227B (67 GHz)</td>
</tr>
</tbody>
</table>

### Others

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N5290-90001</td>
<td>N5290A Broadband Network Analyzer (900 Hz to 110 GHz)</td>
</tr>
<tr>
<td>N5291-90002</td>
<td>N5291A Broadband Network Analyzer (900 Hz to 120 GHz)</td>
</tr>
<tr>
<td>N5264-90030</td>
<td>N5264B (Measurement Receiver)</td>
</tr>
<tr>
<td></td>
<td>N522xB/N523xB/N524xB with Option S93090xA</td>
</tr>
<tr>
<td></td>
<td>(Spectrum Analyzer)</td>
</tr>
</tbody>
</table>

See the equations that are used to generate uncertainty curves.

**Block Diagrams** for the following models are available in VNA Help:

- VNA-X Models with Opt 224 and 423
- N522xB Models with Opt 219 and 419

The following are specifications for discontinued models:

- E8356A, E8357A, E8358A
- E8801A, E8802A, E8803A
- N3381A, N3382A, N3383A
- E8362A, E8363A, E8364A
Addenda
## Applications

<table>
<thead>
<tr>
<th>Option</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>S9x010A/B</td>
<td>Time Domain</td>
</tr>
<tr>
<td>S9x011A/B</td>
<td>Enhanced Time Domain Analysis</td>
</tr>
<tr>
<td>S9x015A/B</td>
<td>Dynamic Uncertainty</td>
</tr>
<tr>
<td>S9x029A/B</td>
<td>Noise Figure and Noise Figure on Converters</td>
</tr>
<tr>
<td>H29</td>
<td></td>
</tr>
<tr>
<td>S9x031B</td>
<td>Phase Noise Application, up to 70 GHz</td>
</tr>
<tr>
<td>S9x032B</td>
<td>Phase Noise Application, up to 125 GHz</td>
</tr>
<tr>
<td>S9x070A/B</td>
<td>Modulation Distortion</td>
</tr>
<tr>
<td>S930700B</td>
<td>Modulation Distortion up to 8.5 GHz</td>
</tr>
<tr>
<td>S930701B</td>
<td>Modulation Distortion up to 13.5 GHz</td>
</tr>
<tr>
<td>S930702B</td>
<td>Modulation Distortion up to 26.5 GHz</td>
</tr>
<tr>
<td>S930704B</td>
<td>Modulation Distortion up to 43.5 GHz</td>
</tr>
<tr>
<td>S930705B</td>
<td>Modulation Distortion up to 50 GHz</td>
</tr>
<tr>
<td>S930707B</td>
<td>Modulation Distortion up to 67 GHz</td>
</tr>
<tr>
<td>S9x080A/B</td>
<td>Frequency Offset</td>
</tr>
<tr>
<td>S9x082A/B</td>
<td></td>
</tr>
<tr>
<td>080</td>
<td></td>
</tr>
<tr>
<td>S9x083A/B</td>
<td>Frequency Converter (FCA)</td>
</tr>
<tr>
<td>S9x082A/B</td>
<td>SMC Only</td>
</tr>
<tr>
<td>080</td>
<td></td>
</tr>
<tr>
<td>S9x086A/B</td>
<td>Gain Compression</td>
</tr>
<tr>
<td>S9x082A/B</td>
<td>Gain Compression on Converters</td>
</tr>
<tr>
<td>080</td>
<td></td>
</tr>
<tr>
<td>S9x087A/B</td>
<td>Swept IMD</td>
</tr>
<tr>
<td>S9x082A/B</td>
<td>Swept IMD on Converters</td>
</tr>
<tr>
<td>080</td>
<td>IM Spectrum</td>
</tr>
<tr>
<td>S9x088A/B</td>
<td>Source Phase Control</td>
</tr>
<tr>
<td>S9x089A/B</td>
<td>Differential IQ</td>
</tr>
<tr>
<td>S9x090A/B</td>
<td>Spectrum Analysis, up to 8.5 GHz</td>
</tr>
<tr>
<td>Application Code</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>S9x0901A/B</td>
<td>Spectrum Analysis, up to 13.5 GHz</td>
</tr>
<tr>
<td>S9x0902A/B</td>
<td>Spectrum Analysis, up to 26.5 GHz</td>
</tr>
<tr>
<td>S9x0904A/B</td>
<td>Spectrum Analysis, up to 43.5 GHz</td>
</tr>
<tr>
<td>S9x0905A/B</td>
<td>Spectrum Analysis, up to 50 GHz</td>
</tr>
<tr>
<td>S9x0907A/B</td>
<td>Spectrum Analysis, up to 67 GHz</td>
</tr>
<tr>
<td>S9x0909A/B</td>
<td>Spectrum Analysis, up to 90 GHz</td>
</tr>
<tr>
<td>S9x093A/B</td>
<td>Extend spectrum analysis to 110 GHz</td>
</tr>
<tr>
<td>S9x094A/B</td>
<td>Extend spectrum analysis beyond 110 GHz</td>
</tr>
<tr>
<td>S9x110A/B</td>
<td>Active (Hot) Parameters</td>
</tr>
<tr>
<td>S9x111A/B</td>
<td>Active (Hot) Parameters Restricted to 50 GHz</td>
</tr>
<tr>
<td>S9x460A/B</td>
<td>iTMSA</td>
</tr>
<tr>
<td>S9x026A/B H08</td>
<td>Integrated Pulse</td>
</tr>
<tr>
<td>S9x790A 790</td>
<td>Measurement Wizard Assistant</td>
</tr>
</tbody>
</table>

* These applications are selected from the Measurement Class dialog and can NOT coexist in a channel with other measurement classes.

Applications WITHOUT * CAN exist in all measurement classes unless they are explicitly restricted.

**See Also**

- All VNA Models and Options
- Diagnostic Tools and Adjustments
- Embedded LO
- External (Banded) Millimeter Modules
- External Multiport Testset Control
- FIFO and other Antenna Features
- IF Path Configuration
- Interface Control
Differential Amplifier Harmonic Distortion and THD

Physical Setup

Measurement Description

Step 1. Create 5 Differential IQ Channels
Step 2. Define Frequency Ranges
Step 3. Add External Source
Step 4. Define Sources
Step 5. Define and Select Trace Parameters
Step 6. Define X-Axis for each Parameter
Step 7. Calibrate using Cal All Channels

Physical Setup

The two VNA internal sources are used as the differential amplifier inputs.

Connect the inputs to ports 1 and 3 using the same frequency range but with a phase offset of 180°.

Connect the outputs to ports 2 and 4.

Measurement Description

The two VNA internal sources are used as the differential amplifier inputs.
Ch1: Sweep differential-input frequency, measure input and output powers, gain, 2nd and 3rd harmonics (dBc), input match, and total harmonic distortion (THD).

Ch2: Sweep differential-input phase, measure input and output powers, gain, and 2nd harmonic (dBc).

Ch3: Sweep differential-input power to measure gain compression, harmonic powers, and THD.

**Step 1. Create 3 Differential IQ Channels**

On the VNA front panel, press **Meas > S-Param > Meas Class....**

Select Differential I/Q, then either:

- OK delete the existing measurement, or
- New Channel to create the measurement in a new channel.

A Differential I/Q measurement is displayed.

**TIP:** Set up one channel, then use **Copy Channel** to create additional channels (Trace/Chan, Channel, Copy Channel)

**Step 2. Define Frequency Ranges**

All channels will have five ranges.

**Ch 1 (Select to make active)**

Swept Input and harmonic frequencies.

Click Stimulus, then DIQ Setup...

Click New four times for a total of 5 ranges.
Click Edit in each range, then enter the following:

**F1:** 10 MHz to 1.0 GHz; IFBW 1 kHz; Uncheck Couple
**F2:** IFBW 1 kHz; Check Couple; Couple to: F1; Check UP; Multiplier: 2
**F3:** IFBW 1 kHz; Check Couple; Couple to: F1; Check UP; Multiplier: 3
**F4:** IFBW 1 kHz; Check Couple; Couple to: F1; Check UP; Multiplier: 4
**F5:** IFBW 1 kHz; Check Couple; Couple to: F1; Check UP; Multiplier: 5

**Ch 2 and Ch 3**

Fixed Frequencies.

TIP: Set up Ch 2, then use Copy Channel to overwrite channel 3.

Click Stimulus, then DIQ Setup...
Click New four times for a total of 5 ranges.
Click Edit in each range, then enter the following:

**F1:** 50 MHz to 50 MHz (CW); IFBW 1 kHz; Uncheck Couple
**F2:** IFBW 1 kHz; Check Couple; Couple to: F1; Check UP; Multiplier: 2
**F3:** IFBW 1 kHz; Check Couple; Couple to: F1; Check UP; Multiplier: 3
**F4:** IFBW 1 kHz; Check Couple; Couple to: F1; Check UP; Multiplier: 4
**F5:** IFBW 1 kHz; Check Couple; Couple to: F1; Check UP; Multiplier: 5

**Step 3. Add External Sources**

External sources are NOT required for this configuration.

**Step 4. Define Sources**

Click Stimulus, then DIQ Setup...
Learn more about these settings.

Make the following Source settings:

**Ch 1 (frequency sweep)**

<table>
<thead>
<tr>
<th>Source Name</th>
<th>State</th>
<th>Frequency</th>
<th>Power</th>
<th>Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td>Auto On</td>
<td>F1</td>
<td>-25.00dBm</td>
<td>0.00°</td>
</tr>
<tr>
<td>Port 2</td>
<td>Off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 3</td>
<td>Auto On</td>
<td>F1</td>
<td>-25.00dBm</td>
<td>180.00°</td>
</tr>
<tr>
<td>Port 4</td>
<td>Off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 1 Src2</td>
<td>Off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Ch2 (phase sweep)**
Note: Do not attempt to control the phase of port 1. It will automatically get set to zero degrees when configured as the reference for port 3.

**Optional Source Configuration Settings**

**Match Correction ON to improve Power Accuracy**

For ALL channels, on the Port 2 dialog:

- Check Match Correction ON.
- Then apply Match Correction for F1, F2, F3, F4.
- "Off+Match" is annotated on the Source setting dialog, although this is NOT shown in the images above.
Decrease Mag/Phase Tolerance and Increase Iterations for Improved Accuracy

On any source in which a Phase setting is made, click Phase Control Setup. Decrease the Tolerance and increase the Max Iterations to improve phase accuracy. Learn more.

Step 5 Define and Select Trace Parameters

Note: Some of the parameters above are NOT displayed. These parameters can be used as diagnostic or troubleshooting parameters. For example, the "M_" terms are created when match correction is applied, but they are not usually displayed directly.

Click Instrument, then Trace, then Add Trace, New Trace..., then Edit Parameters.

Click Load, then navigate to the following VNA folder location to load these parameters:

C:\Users\Public\Documents\Network Analyzer\Samples\Setups\DiffIQ\Diff_Harmonics_THD.xml

Once these parameters are defined, select (check) the parameters for each channel as follows:

Ch1: Ipwr_F1, Opwr_F1, Gain, 2nd_dBc, 3rd_dBc, Sdd11_F1, THD
Ch2: Ipwr_F1, Opwr_F1, Gain, 2nd_dBc
Ch3: Ipwr_F1, Opwr_F1, Opwr_F2, Opwr_F3, Opwr_F4, Opwr_F5, Gain, THD

Step 6. Define X-Axis For Each Parameter

For each channel, click Stimulus, then Sweep, then X-axis Type...

Make the following selections:

<table>
<thead>
<tr>
<th>Channel</th>
<th>X-Axis Domain</th>
<th>X-Axis Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Frequency</td>
<td>F1</td>
</tr>
<tr>
<td>2</td>
<td>Phase</td>
<td>Port 3</td>
</tr>
<tr>
<td>3</td>
<td>Power</td>
<td>Port 1</td>
</tr>
</tbody>
</table>

Step 7. Calibrate using Cal All Channels

Click Response, then Cal, then Cal All...

Select the ports (in circle) for ALL channels.
Differential Amplifier IMD

Physical Setup

Measurement Description
Step 1. Create Differential IQ Channel
Step 2. Define Frequency Ranges
Step 3. Add External Sources
Step 4. Define Sources
Step 5. Define and Select Trace Parameters
Step 6. Define X-Axis for each Parameter
Step 7. Calibrate using Cal All Channels

See All DIQ Examples

Physical Setup

The two PNA internal sources are combined to create one differential amplifier input.

Two external sources are combined to create the other differential amplifier input.
Connect the inputs to ports 1 and 3.
Connect the outputs to ports 2 and 4.
Connect the two external sources to the PNA rear panel as shown above. One external combiner is required.

**Measurement Description**

Ch1: Sweep input frequencies of four signals (two pair of differential signals), measure input and output main-tone powers, third-order lower and upper IM products, and gain of DUT.

**Step 1. Create a Differential IQ Channel**

On the PNA front panel, press Meas, then [Measurement Class]
Select Differential I/Q, then either:
   - OK delete the existing measurement, or
   - New Channel to create the measurement in a new channel.

A Differential I/Q measurement is displayed.

**Step 2. Define Frequency Ranges**

Channel 1 will have five frequency ranges.

**Ch 1**

Swept Input frequencies.

Use the Couple settings to create the following Tone Spacings:
Click Stimulus, then Frequency, then Differential IQ Setup
Click New four times for a total of 5 ranges.
Click Edit in each range, then enter the following:

- **F1 (Tone offset):** 10 MHz to 10 MHz (CW); IFBW 1 kHz; Uncheck Couple
- **F2 (Main Lo In):** 20 MHz to 7 GHz; IFBW 1 kHz; Uncheck Couple.
- **F3 (Main Hi In: F2+F1):** IFBW 1 kHz; Check Couple; Couple to: F2; Offset: F1; Check UP.
- **F4 (IM3Lo: F2-F1):** IFBW 1 kHz; Check Couple; Couple to: F2; Offset: F1; Uncheck UP.
- **F5 (IM3Hi: F3+F1):** IFBW 1 kHz; Check Couple; Couple to: F3; Offset: F1; Check UP.

**Step 3. Add Two External Sources**

Click Utility, then Configure, then External Device Configuration
Click New, then complete the dialog. Learn how.

**Step 4. Define Sources**

Click Stimulus, then Frequency, then Differential IQ Setup
Learn more about these settings.

Make the following Source settings:

**Ch 1 (frequency sweep)**

Do NOT manually set the two external sources to zero degrees. They will get set automatically when configured as reference ports.

Set BOTH external sources to receiver leveling using the R3 receiver as follows:
On the Source Configuration dialog, click Power and Attenuators, then Receiver Leveling.

![Source Configuration dialog](image1)

Select the external source (MXG_N51 in this example).
Then select Receiver name: R3.

**Set Path Configuration**

These settings route the combined sources to the differential input ports.

On the Source Configuration dialog, click Power and Attenuators (shown above), then Path Configuration.

![Path Configuration](image2)

Make the two settings in yellow circles (shown above).

**Optional Source Configuration Settings**

**Match Correction ON to improve Power Accuracy**
For Ports 2 and 4, on the Source Configuration dialog:

Check Match Correction ON.

Then apply Match Correction for F1, F2, F3, F4, F5

"Off+Match" is annotated on the Source setting dialog, although this is NOT shown in the images above.

Decrease Mag/Phase Tolerance and Increase Iterations for Improved Accuracy

On any source in which a Phase setting is made, click Phase Control Setup. Decrease the Tolerance and increase the Max Iterations to improve phase accuracy. Learn more.

Step 5 Define and Select Trace Parameters

Note: Some of the parameters above are NOT displayed. These parameters can be used as diagnostic or troubleshooting parameters. For example, the "M_" terms are created when match correction is applied, but they are not usually displayed directly.
Click Trace/Chan, then Trace, then New Trace, then Edit Parameters.

Click Load, then navigate to the following PNA folder location to load these parameters:

C:\Users\Public\Documents\Network Analyzer\Samples\Setups\DiffIQ\Diff_IMD.xml

Once these parameters are defined, click Trace/Chan, then Trace, then New Trace. Then select (check) the following parameters:

Ch1: PwrMainLoIn, PwrMainHiIn, PwrMainLo, PwrMainHi, Pwr3Lo, Pwr3Hi, ToneGainLo, ToneGainHi

**Step 6. Optional. Define X-Axis For Each Parameter**

For each parameter, click Response, then Measure, then Select X-Axis

**Step 7. Calibrate using Cal All Channels**

Click Response, then Start Cal, then Cal All Channels

![Image of Calibrate All Selected Channels]

Select ports 1 through 4.

**After Calibration, Turn Off Source Cal**

Because the path configuration is different during the measurement as during Cal All, receiver leveling is used to set the correct port powers of the external sources. And since the internal sources are controlled, their power is set relative to the external sources.

Click Response, then Cal, then Power Cal, then Source Correction OFF.
Differential I/Q Modulator

The upconverter has differential IQ, balanced inputs.

Three external generators are required.

The two VNA internal sources are used as the differential I inputs.

- Connect to ports 1 and 3 using the same frequency range.
- Set the phase of port 3 to 180° relative to port 1.

Two external sources are used as the differential Q inputs.

- Connect to ports 2 and 4 using the same frequency range as ports 1 and 3.
- Set the phase of port 2 to +90° relative to port 1.
- Set the phase of port 4 to -90° relative to port 1.

An external RF Source is connected to the LO input. If the LO port is balanced, a balun or 180° hybrid coupler may be used as shown in the example.

- When using a Fixed LO, there is no need to control the source.
- Otherwise, first configure the source in the External Source configuration dialog.

Remove the port 3 front-panel jumper and connect the mixer output to the C receiver input.
This receiver is calibrated using the CPM - Direct Access Receiver Calibration. Learn more.
Differential Mixer

Physical Setup
Measurement Description
Step 1. Create 5 Differential IQ Channels
Step 2. Define Frequency Ranges
Step 3. Add External Source
Step 4. Define Sources
Step 5. Define and Select Trace Parameters
Step 6. Define X-Axis for each Parameter
Step 7. Calibrate using Cal All Channels

See All DIQ Examples
The two VNA internal sources are used as the mixer differential RF inputs.

Connect to Ports 1 and 3 using the same frequency range but with a phase offset of 180°. An external RF Source is connected to the LO input.

When using a Fixed LO, there is no need to control the source. Connect the two IF outputs to Ports 4 and 2.

**Measurement Description**

Ch1: Sweep differential-RF frequency with fixed LO, measure forward input and output powers, forward conversion loss (SC21), differential input match (Sdd11) and LO-to-RF and LO-to-IF feed-through.

Ch2: Sweep differential-IF frequency with fixed LO, measure reverse input and output powers, reverse conversion loss (SC12), differential output match (Sdd22)

Ch3: Sweep differential-RF phase, measure input and output powers and forward conversion loss

Ch4: Sweep differential-RF input power to measure gain compression

**Step 1. Create 4 Differential IQ Channels**

On the VNA front panel, press Meas > S-Param > Meas Class.... Select Differential I/Q, then either:

OK delete the existing measurement, or

New Channel to create the measurement in a new channel.
A Differential I/Q measurement is displayed.

**TIP:** Set up one channel, then use **Copy Channel** to create additional channels (Trace/Chan, Channel, Copy Channel)

### Step 2. Define Frequency Ranges

All channels will have the following ranges:

- **F1:** RF frequencies
- **F2:** LO frequencies
- **F3:** IF frequencies

**Ch 1 and Ch 2 (Select to make active)**

Swept RF and IF; Fixed LO.

**TIP:** Set up Ch 1, then use **Copy Channel** to overwrite channel 2.

- Click Stimulus, then DIQ Setup...
- Click New two times for a total of 3 ranges.
- Click Edit in each range, then enter the following:
  - **F1:** 2.5 GHz to 3.0 GHz; IFBW 1 kHz; Uncheck Couple
  - **F2:** 2.0 GHz to 2.0 GHz (CW); IFBW 1 kHz; Uncheck Couple
  - **F3:** IFBW 1 kHz; Check Couple; Couple to: F1; Offset: F2, Uncheck UP

**Ch 3 and Ch 4**

Fixed RF, IF; and LO.

**TIP:** Set up Ch 3, then use **Copy Channel** to overwrite channel 4.

- Click Stimulus, then DIQ Setup...
- Click New two times for a total of 3 ranges.
- Click Edit in each range, then enter the following:
  - **F1:** 2.75 GHz to 2.75 GHz (CW); IFBW 1 kHz; Uncheck Couple
  - **F2:** 2.0 GHz to 2.0 GHz (CW); IFBW 1 kHz; Uncheck Couple
  - **F3:** IFBW 1 kHz; Check Couple; Couple to: F1; Offset: F2, Uncheck UP

### Step 3. Add One External Source

- Click Instrument, then Setup, then External Hardware, then **External Device**...
- Click New, then complete the dialog. [Learn how](#)

### Step 4. Define Sources

- Click Stimulus, then DIQ Setup...
- [Learn more about these settings](#)

Make the following Source settings:

**Ch 1 (fwd. sweep, fixed 180o offset)**
Ch2 (rev. sweep, fixed 180° offset)

<table>
<thead>
<tr>
<th>Source Name</th>
<th>State</th>
<th>Frequency</th>
<th>Power</th>
<th>Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td>Auto_On</td>
<td>F1</td>
<td>-15.00dBm</td>
<td>0.00°</td>
</tr>
<tr>
<td>Port 2</td>
<td>Off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 3</td>
<td>Auto_On</td>
<td>F1</td>
<td>-15.00dBm</td>
<td>180.00°</td>
</tr>
<tr>
<td>Port 4</td>
<td>Off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 1 Src2</td>
<td>Off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>MXG_N5183A</td>
<td>Always_On</td>
<td>F2</td>
<td>8.000dBm</td>
<td></td>
</tr>
</tbody>
</table>

Ch 3 (RF phase sweep)

<table>
<thead>
<tr>
<th>Source Name</th>
<th>State</th>
<th>Frequency</th>
<th>Power</th>
<th>Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td>Off</td>
<td>F1</td>
<td>-15.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 2</td>
<td>Auto_On</td>
<td>F3</td>
<td>-15.00dBm</td>
<td>0.00°</td>
</tr>
<tr>
<td>Port 3</td>
<td>Off</td>
<td>F1</td>
<td>-15.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 4</td>
<td>Auto_On</td>
<td>F3</td>
<td>-15.00dBm</td>
<td>180.00°</td>
</tr>
<tr>
<td>Port 1 Src2</td>
<td>Off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>MXG_N5183A</td>
<td>Auto_On</td>
<td>F1</td>
<td>8.000dBm</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Ch 4 (I/Q power sweep)
Note: Do not attempt to control the phase of port 1. It will automatically get set to zero degrees when configured as the reference for port 3.

Optional Source Configuration Settings

**Match Correction ON to improve Power Accuracy**

For ALL channels, on the Port 2 dialog:

- Check MatchCorrection ON.
- Then apply Match Correction for ranges (F1, F2, F3)

"Off+Match" is annotated on the Source setting dialog, although this is NOT shown in the images above.
Decrease Mag/Phase Tolerance and Increase Iterations for Improved Accuracy

On any source in which a Phase setting is made, click Phase Control Setup. Decrease the Tolerance and increase the Max Iterations to improve phase accuracy. Learn more.

Step 5 Define and Select Trace Parameters

![New Trace](image)

Note: Some of the parameters above are NOT displayed. These parameters can be used as diagnostic or troubleshooting parameters. For example, the "M_" terms are created when match correction is applied, but they are not usually displayed directly.

Click Instrument, then Trace, then Add Trace, New Trace..., then Edit Parameters.

Click Load, then navigate to the following VNA folder location to load these parameters:

C:\Users\Public\Documents\Network Analyzer\Samples\Setups\DiffIQ\Diff_mixer.xml

Once these parameters are defined, select (check) the parameters for each channel as follows:

Ch1:
Ch2:
Ch3:
Ch4:
Ch5:

Step 6. Define X-Axis For Each Parameter

For each channel, click Stimulus, then Sweep, then X-axis Type...

Make the following selections:

<table>
<thead>
<tr>
<th>Channel</th>
<th>X-Axis Domain</th>
<th>X-Axis Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 2</td>
<td>Frequency</td>
<td>F1, F3</td>
</tr>
<tr>
<td>3</td>
<td>Phase</td>
<td>Port 3</td>
</tr>
<tr>
<td>4</td>
<td>Power</td>
<td>Port 1</td>
</tr>
</tbody>
</table>

Step 7. Calibrate using Cal All Channels
Click Response, then Cal, then Cal All...

Select the ports (in circle) for ALL channels.

See Cal All for more information.
DIQ Mixer Spurs SE
DIQ Spur Search for Single-ended Mixer

Physical Setup

Measurement Description
Step 1. Create Differential IQ Channel
Step 2. Define Frequency Ranges
Step 3. Add External Sources
Step 4. Define Sources
Step 5. Define and Select Trace Parameters
Step 6. Define X-Axis for each Parameter
Step 7. Calibrate using Cal All Channels

See All DIQ Examples

Physical Setup

Connect the RF input to port 1.
Connect the IF output to ports 2.
Connect the LO to port 3.
### Measurement Description

Ch1: Sweep input frequency and at output, measure LO fundamental and harmonic powers, plus spurs (in dBc relative to the input power), of order n*RF - m*LO, where n and m are integers 1, 2, and 3.

### Step 1. Create a Differential IQ Channel

On the VNA front panel, press **Meas > S-Param > Meas Class....**

Select Differential I/Q, then either:

- OK delete the existing measurement, or
- New Channel to create the measurement in a new channel.

A Differential I/Q measurement is displayed.

### Step 2. Define Frequency Ranges

Channel 1 will have 13 frequency ranges.

Click Stimulus, then DIQ Setup...

Click New 12 times for a total of 13 ranges.

Click Edit in each range, then enter the following:

- **F1 (RF Input):** 3.0 GHz to 3.2 GHz; IFBW 1 kHz; Uncheck Couple
- **F2 (LO):** 2.9 GHz to 2.9 GHz (CW); IFBW 1 kHz; Uncheck Couple.
- **F3 (2*LO):** IFBW 1 kHz; Check Couple; Couple to: F2; Offset: F2; Multiplier: 2. Check UP.
- **F4 (3*LO):** IFBW 1 kHz; Check Couple; Couple to: F2; Offset: F2; Multiplier: 3. Check UP.
- **F5 (1RF - LO):** IFBW 1 kHz; Check Couple; Couple to: F1; Offset: F2; Uncheck UP.

Complete the remaining ranges (F6 to F13) using the above pattern and following formulas:
### Step 3. Add External Sources

This example does NOT use external sources.

### Step 4. Define Sources

Click Stimulus, then DIQ Setup...

Learn more about these settings.

Make the following Source settings:

**Ch 1 (frequency sweep)**

<table>
<thead>
<tr>
<th>Source Name</th>
<th>State</th>
<th>Frequency</th>
<th>Power</th>
<th>Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td>Auto On</td>
<td>F1</td>
<td>-15.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 2</td>
<td>Off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 3</td>
<td>Always On</td>
<td>F2</td>
<td>8.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 4</td>
<td>Off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 1 Src2</td>
<td>Off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
</tbody>
</table>

This example does NOT use phase control.

**Optional Source Configuration Settings**

**Match Correction ON to improve Power Accuracy**

For Port 2 and 4, on the Source Configuration dialog:

Check Match Correction ON.

Then apply Match Correction for all ranges.

"Off+Match" is annotated on the Source setting dialog, although this is NOT shown in the images above.
Step 5 Define and Select Trace Parameters

Note: Some of the parameters above are NOT displayed. These parameters can be used as diagnostic or troubleshooting parameters. For example, the “M_” terms are created when match correction is applied, but they are not usually displayed directly.

Click Instrument, then Trace, then Add Trace, New Trace..., then Edit Parameters.

Click Load, then navigate to the following VNA folder location to load these parameters:

C:\Program Files\Keysight\Network Analyzer\Documents\Templates\Samples\Setups\DiffIQ\Mixer_spurs.xml

Once these parameters are defined, click Instrument, then Trace, then Add Trace, New Trace... Then select (check) ALL parameters.

Step 6. Optional. Define X-Axis For Each Parameter

For each parameter, click Stimulus, then Sweep, then X-axis Type...

Step 7. Calibrate using Cal All Channels
Click Response, then Cal, then Cal All...

Select ports 1 through 3.
DIQ Single-Ended IQ Upconverter

Physical Setup
Measurement Description
Step 1. Create 5 Differential IQ Channels
Step 2. Define Frequency Ranges
Step 3. Add External Source
Step 4. Define Sources
Step 5. Define and Select Trace Parameters
Step 6. Define X-Axis for each Parameter
Step 7. Calibrate using Cal All Channels

See All DIQ Examples

Physical Setup

IQ signals normally contain digital information, but when tested with a network analyzer, are simple sinusoidal signals.

LO is used as the RF carrier signal.
RF is the modulated output.
The two VNA internal sources are used as the converter’s quadrature inputs.

Connect to ports 1 and 3 using the same frequency range but with a phase offset of 90°.
An external RF Source is connected to the LO input(s).

When using a Fixed LO, there is no need to control the source.
Connect the output to port 2.

Measurement Description

Ch1: Sweep I/Q frequencies with fixed LO; at RF output, measure LO+I/Q and LO-I/Q sideband frequency response, image rejection (upper-sideband/lower-sideband), and LO leakage.

Ch2: Sweep Q phase at fixed I/Q, LO, and RF frequencies; measure sidebands.

Ch3: Sweep I power at fixed I/Q, LO, and RF frequencies; measure sidebands.

Ch4: Sweep LO frequency with fixed I/Q input frequencies; measure sidebands.

Ch5: Sweep I/Q power at fixed I/Q, LO, and RF frequencies; measure sideband compression.

Step 1. Create 5 Differential IQ Channels

On the VNA front panel, press **Meas > S-Param > Meas Class....**
Select Differential I/Q, then either:
  - OK delete the existing measurement, or
  - New Channel to create the measurement in a new channel.
A Differential I/Q measurement is displayed.
**Step 2. Define Frequency Ranges**

All five channels will have the following ranges:

- **F1**: I/Q frequencies
- **F2**: LO frequencies
- **F3**: Upper sideband frequencies (F2 + F1)
- **F4**: Lower sideband frequencies (F2 - F1)

**Ch 1 (Select to make active)**

Click Stimulus, then DIQ Setup...

Click New three times for a total of 4 ranges.

Click Edit in each range, then enter the following:

- **F1**: 10 MHz to 2.4 GHz; IFBW 1 kHz; Uncheck Couple
- **F2**: 2.5 GHz to 2.5 GHz (CW); IFBW 1 kHz; Uncheck Couple
- **F3**: IFBW 1 kHz; Check Couple; Couple to: F2; Offset: F1, Check UP
- **F4**: 2.49 GHz to 100 MHz; IFBW 1 kHz; Uncheck Couple

**Ch 2, Ch 3, and Ch 5**

**TIP**: Set up Ch 2, then use **Copy Channel** to overwrite channels 3 and 5.

Click Stimulus, then DIQ Setup...

Click New three times for a total of 4 ranges.

Click Edit in each range, then enter the following:

- **F1**: 1.2 GHz to 1.2 GHz (CW); IFBW 1 kHz; Uncheck Couple
- **F2**: 2.5 GHz to 2.5 GHz (CW); IFBW 1 kHz; Uncheck Couple
- **F3**: 3.7 GHz to 3.7 GHz (CW); IFBW 1 kHz; Uncheck Couple
- **F4**: IFBW 1 kHz; Check Couple; Couple to: F2; Offset: F1, Uncheck UP

**Ch 4**

Click Stimulus, then DIQ Setup...

Click New three times for a total of 4 ranges.

Click Edit in each range, then enter the following:

- **F1**: 500 MHz to 500 MHz (CW); IFBW 1 kHz; Uncheck Couple
- **F2**: 1.0 GHz to 4.4 GHz; IFBW 1 kHz; Uncheck Couple
- **F3**: 1.5 GHz to 4.9 GHz; IFBW 1 kHz; Uncheck Couple
- **F4**: 500 MHz to 3.9 GHz; IFBW 1 kHz; Uncheck Couple

**Step 3. Add External Sources**

Click Instrument, then Setup, then External Hardware, then External Device...

Click New, then complete the dialog. Learn how.

**Step 4. Define Sources**
Click Stimulus, then DIQ Setup...

Learn more about these settings.

Make the following Source settings:

**Ch 1 and 4 (fixed 90° offset)**

<table>
<thead>
<tr>
<th>Source Name</th>
<th>State</th>
<th>Frequency</th>
<th>Power</th>
<th>Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td>Auto_On</td>
<td>F1</td>
<td>-10.00dBm</td>
<td>0.00°</td>
</tr>
<tr>
<td>Port 2</td>
<td>off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 3</td>
<td>Auto_On</td>
<td>F1</td>
<td>-10.00dBm</td>
<td>90.00°</td>
</tr>
<tr>
<td>Port 4</td>
<td>off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 1 Src2</td>
<td>off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>MXG_N5183A</td>
<td>Always_On</td>
<td>F2</td>
<td>11.00dBm</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Ch2 (Q phase sweep)**

<table>
<thead>
<tr>
<th>Source Name</th>
<th>State</th>
<th>Frequency</th>
<th>Power</th>
<th>Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td>Auto_On</td>
<td>F1</td>
<td>-10.00dBm</td>
<td>0.00°</td>
</tr>
<tr>
<td>Port 2</td>
<td>off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 3</td>
<td>Auto_On</td>
<td>F1</td>
<td>-10.00dBm</td>
<td>-180.00° to 180.00°</td>
</tr>
<tr>
<td>Port 4</td>
<td>off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 1 Src2</td>
<td>off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>MXG_N5183A</td>
<td>Always_On</td>
<td>F2</td>
<td>11.00dBm</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Ch 3 (I power sweep)**

<table>
<thead>
<tr>
<th>Source Name</th>
<th>State</th>
<th>Frequency</th>
<th>Power</th>
<th>Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td>Auto_On</td>
<td>F1</td>
<td>-12.00dBm to -8.00dBm</td>
<td>0.00°</td>
</tr>
<tr>
<td>Port 2</td>
<td>off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 3</td>
<td>Auto_On</td>
<td>F1</td>
<td>-10.00dBm</td>
<td>90.00°</td>
</tr>
<tr>
<td>Port 4</td>
<td>off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>Port 1 Src2</td>
<td>off</td>
<td>F1</td>
<td>-5.00dBm</td>
<td>N/A</td>
</tr>
<tr>
<td>MXG_N5183A</td>
<td>Always_On</td>
<td>F2</td>
<td>11.00dBm</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Ch 5 (I/Q power sweep)**
Note: Do not attempt to control the phase of port 1. It will automatically get set to zero degrees when configured as the reference for port 3.

Optional Source Configuration Settings

**Match Correction ON to improve Power Accuracy**

For ALL channels, on the Port 2 dialog:

- Check Match Correction ON.
- Then apply Match Correction for measurements of LO and sidebands (F2, F3, F4)

"Off+Match" is annotated on the Source setting dialog, although this is NOT shown in the images above.
Decrease Mag/Phase Tolerance and Increase Iterations for Improved Accuracy

On any source in which a Phase setting is made, click Phase Control Setup. Decrease the Tolerance and increase the Max Iterations to improve phase accuracy. Learn more.

Step 5 Define and Select Trace Parameters

Click Instrument, then Trace, then Add Trace, New Trace..., then Edit Parameters.

Then do one of the following to create these Diff I/Q measurement parameters:

Click Load, then navigate to the following VNA folder location to load these parameters:

C:\Users\Public\Documents\Network Analyzer\Samples\Setups\DiffIQ\IQ_up_converter.xml

Use the Edit Parameters dialog to manually create these parameters.

Use a text editor to create an *.xml file. Learn how.

Once these parameters are defined, select (check) the parameters for each channel as follows:

Ch1: LO+I/Q, LO-I/Q, Image rejection, LO leakage
Ch2: LO+I/Q, LO-I/Q, Image rejection
Ch3: LO+I/Q, LO-I/Q, Image rejection
Ch4: LO+I/Q, LO-I/Q
Ch5: LO+I/Q, LO-I/Q, LO+I/Q com

Step 6. Define X-Axis For Each Parameter

For each channel, click Stimulus, then Sweep, then X-axis Type...

Make the following selections:
<table>
<thead>
<tr>
<th>Channel</th>
<th>X-Axis Domain</th>
<th>X-Axis Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Frequency</td>
<td>F1</td>
</tr>
<tr>
<td>2</td>
<td>Phase</td>
<td>Port 3</td>
</tr>
<tr>
<td>3</td>
<td>Power</td>
<td>Port 1</td>
</tr>
<tr>
<td>4</td>
<td>Frequency</td>
<td>F2</td>
</tr>
<tr>
<td>5</td>
<td>Power</td>
<td>Port 1</td>
</tr>
</tbody>
</table>

**Step 7. Calibrate using Cal All Channels**

Click Response, then Cal, then Cal All...

Select the ports (in circle) for ALL channels.
Noise Figure with External Testset

The low-noise receiver and the noise receiver path switch is added between the port 2 CPLR THRU jumper connector and the port 2 bias tee as shown in the images below.

When it is configured with a multiport test set (i.e. U3024AH10), the low-noise receiver cannot be used with test ports on the test set with standard multiport test set jumper connections. See the following modified test set connections to use 50 GHz low-noise receiver with multiport test set.

50 GHz 2-port Noise Figure Components (shaded in yellow)
Standard Connections

Modified Connections w/External Coupler

2-port coupler used with 50 GHz Opt. 029 (low-noise receivers) with multiport test set.
4-port coupler used with 50 GHz Opt. 029 (low-noise receivers) with multiport test set.
Swept IMD Configure External Source and Combiner

Configure an External Source and Combiner for Swept IMD Measurements using the N522xB Models

The following configurations can be used on the N522xB models for use with the IMD Apps (Swept IMD, IMDx, IM Spectrum, and IMx Spectrum).

- N522xB 2-port models
- N522xB 4-port models using Internal Components
- N522xB 4-port models using External Components

These are NOT the only supported configurations, but are intended to be used as examples. However, base N522xB models (without front-panel jumpers) are NOT supported with the IMD Apps.

See Also

- N522xB Specs - Block diagrams appear in the last pages.
- N522xB Models and Options

N522xB 2-port models (Single source; no internal combiner)

Required

- **External Source** - see list of supported sources.
- **Combiner** - Used to combine f1 (from internal source) and f2 (from external source).
- **Coupler** - Used to route input tones to R1 receiver.
- **Cables** - Used to make connections as shown in image.

Notes
The R1 reference switch MUST be set to "External". Press Setup > External Hardware > External Testset. R1 receiver is used to measure both the $f_1$ and $f_2$ tones. R1 is inferred by the VNA when you select Port 1 as the DUT input on the IMD Setup dialog Power tab. Connection at 'd' above is to measure DUT input reflections at receiver A.

**Procedure**

Remove the front-panel jumpers with labels in above image. Configure the setup as shown in above image.

In the External Source Configuration dialog, configure the external source.

In the IMD Setup dialog; Configuration tab, select the external source as f2 tone.

---

**N522xB 4-port models using internal VNA components**

Port 3 coupler is used to combine $f_1$ and $f_2$.

Port 4 coupler is used as a splitter to route the tones to the R1 receiver.

You can also use external components for one or both of these functions for less loss or better power-handling capability.

This configuration can also be used for 4-port PNA-X models with no internal combiner.
Required

Cables - Used to make connections as shown in image.

Notes

R1 receiver must measure both the \( f_1 \) and \( f_2 \) tone. R1 is inferred by the VNA when you select Port 1 as the DUT input on the IMD Setup dialog Power tab.

Connection at 'c' above is to measure DUT input reflections at receiver A.

Procedure

Remove the front-panel jumpers that have labels in above image. Not all front-panel connectors will be cabled when finished.

Configure the setup as shown in above image. Be sure to make connections to ALL four VNA test ports as shown in about image.

N522xB 4-port models using External Components

This configuration can also be used for 4-port PNA-X models with no internal combiner.
Required

**Combiner** - Used to combine $f_1$ and $f_2$.

**Coupler** - Used to route input tones to R1 receiver.

**Cables** - Used to make connections as shown in image.

Notes

R1 receiver is used to measure both the $f_1$ and $f_2$ tones. R1 is inferred by the VNA when you select Port 1 as the DUT input on the IMD Setup dialog Power tab.

Connection at 'c' above is to measure DUT input reflections at receiver A.

Procedure

Remove the front-panel jumpers with labels in above image.

Configure the setup as shown in above image.
**SweptIMDLimitedPortMapping**

The IMD, IMDx, IMSpectrum, and IMxSpectrum Apps have a limited port mapping capability. This would allow you, for example, to simultaneously measure two 2-port DUTs in different channels. You could input the 2-tone signal to port 1 and port 3. The DUT output would be connected to port 2 and port 4.

Port selections are made on the Power Tab for each Application.

**Note:** This setup requires an external combiner connected to the rear-panel. The external combiner that is shown has two inputs and two outputs. This can also be accomplished by using two components.

This example setup shows a 4-port transceiver connected to the VNA ports as follows:

<table>
<thead>
<tr>
<th>VNA Port</th>
<th>DUT Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Rx In</td>
</tr>
<tr>
<td>2</td>
<td>Rx Out</td>
</tr>
<tr>
<td></td>
<td>Tx In</td>
</tr>
<tr>
<td>---</td>
<td>-------</td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Tx Out</td>
</tr>
</tbody>
</table>

**Notes**

When DUT Input = VNA port 1, then DUT Output = VNA port 2 (default) OR port 4.
When DUT Input = VNA port 3, then DUT Output = VNA port 4.
Each port mapping configuration must be setup in separate channels.
The input and output frequencies for each channel can be at different frequencies.
The Path Configuration settings are made automatically.
Calibration is performed normally using the IMD Cal Wizard.
Because an external combiner is used, input power to the DUT must be monitored by you to ensure that the DUT is not being overdriven.
Both input tones must be supplied by the internal VNA sources.
A mixer LO can be a VNA internal source, an external source, or an embedded LO.
Critical Information

During boot up of Windows or of the Network Analyzer application program, do NOT press keys on the front panel, rotate the RPG knob, or connect a USB device. Doing so MAY lead to a front panel lockup state. To clear this state; **Shutdown and Restart** (do not Hibernate) the VNA.

**Prevent Damage to your VNA Hard Disk Drive.**
**Follow proper VNA Shutdown procedures.**
**Perform Display Test to see if your VNA meets specification.**
**See DC Voltage damage levels for your VNA model.**
**See Preventing Electrostatic Damage on your VNA**
**See Preventing Test Port Connector Damage**
**See Extremely Fragile N5247B front-panel loops**

The VNA does NOT have antivirus protection when shipped. Use of an antivirus program is strongly recommended if you connect the VNA to the Internet.

We no longer sell VNA printed manuals. However, several printable versions of VNA Help in pdf format are available at [http://na.support.keysight.com/pna/help/](http://na.support.keysight.com/pna/help/)
Home Frequency Offset Measurements

Frequency Offset measurements require the VNA source to tune to a different frequency than the VNA receiver.

This capability is offered as an option on some VNA models.

To view the options that are installed on your VNA, click Help then About Network Analyzer.

- **Frequency Offset (Option S93080A)** - Allows basic frequency converter measurements.
- **Frequency Converter Application (Option S93083A/B)** - Allows you to easily make extremely accurate scalar and phase frequency converter measurements.
- **Tutorials** - Learn how to make specific Frequency Conversion Device Measurements.

SMC

FCA Overview
SMC Measurements
Mixer Converter Setup
Frequency Offset Mode
Front Panel Jumpers

There are many reasons to configure the front-panel jumpers on the VNA.

- High-power measurements using the VNA - App Note 1408-10
- Using a Booster Amp for SMC Measurements
- Improving Dynamic Range

**Note: Phase-locking and the Reference Receivers**

The R1 and R2 reference receivers are used for phase-locking between the RF source and receiver LO. The phase-locking requirement imposes the following restrictions on the reference signal:

- **PNA-X and N522x** power level: limited by the noise floor
- All models require a clean signal without spurious content

In Frequency Offset Mode (FOM) Opt S93080A, the R1 receiver is NOT used for phase locking. Independent internal circuitry is used to phase lock the source and receivers. When FOM is available, turn FOM ON simply to take advantage of the independent phase locking mechanism. If NOT measuring at different source and receiver frequencies, set the offset to zero, so that the source and receiver frequencies are the same.

**Topics**

- High-power measurements

**See Also**

- Arbitrary Ratio
- Block diagrams and Specifications
- VNA Options and Configurations
Active Hot Parameters AvgBw

Main
- Averaging
- Averaging Restart
- Average Type
- IF Bandwidth
- LF Auto BW

Smoothing
- Smoothing
- Smooth Percent
- Smooth Points

Delay Aperture
- Aperture Percent
- Aperture Points
- Aperture Freq
Active Hot Parameters Cal

Main
  Cal All...
  Correction
  Src Power Correct
  Interpolation
  Correction Properties

Cal Sets & Cal Kits
  Cal Set...
  Cal Set Viewer
  Cal Kit...
  Ecal

  Show Connected ECals...
  Restore ECal Memory...

  Cal Pod...

Fixtures
  Apply Fixtures
  Power Comp...
  Fixture Setup

  2-port DeEmbed...
  Cal Plane Manager...
Active Hot Parameters Meas

S-param

HotS11, HotS31, HotS13, HotS33
Other...
Meas Class...

Meas Setup

Conversions
Correction
Trace Hold
Equation Editor
Memory
Time Domain
Pulse Setup
Active Hot Parameters Power

Main
- Power Level
- RF Power
- Start Power
- Stop Power
- Power and Attenuators...
- AHP Setup...

Leveling & Offset
- Select
- Slope
- Offset
- Limit
- Offset and Limits...

Attenuators
- Source Port 1, 2, 3, 4
- Receiver A, B, C, D
Active Hot Parameters Setup

Main
  - Sweep Setup...
  - Meas Class...
  - Quick Start

Layout
  - New Trace
  - New Channel
  - New Window
  - New Sheet
  - Delete
  - Select
  - Measure
  - Meas Class...

System Setup
  - Next/Prev Keys: Select Next/Previous window/channel/trace for the selected one
  - Preferences...
  - Sound: Turn On/Off beep sound
    - Remote Interface...
    - LAN Status...
    - Code Emulation

Internal Hardware
  - RF Path Config...
  - IF Path Config...
  - Pulse Gen Config...
  - Mechanical Devices...
  - Interface Control...

External Hardware
  - External Device...
  - Power Meter Setup...
  - Multiport
  - Millimeter Config...
Active Hot Parameters Sweep

Main
- Number of Points
- Sweep Type
- Start (Frequency sweep, Power Sweep)
- Stop (Frequency sweep, Power Sweep)
- AHP Setup...

Sweep Timing
- Sweep Time
- Dwell Time
- Sweep Delay
- Sweep Mode
- Sweep Sequence
- Fast Sweep

Source Control
- Frequency Offset...
- Pulse Setup...
- Balanced Source...
- Phase Control...
- DC Source...

Segment Table
- Add Segment, Insert Segment, Delete Segment, Delete All Segments
- Segment Table...
- Show Table
Active Hot Parameters Freq

Main

Start, Stop, Center, Span, Step
CW
Frequency Offset...
AHP Setup...
DIQ Cal

Differential I/Q Cal

Main

Cal All...
Correction
  Correction On
  Correction Off
  Cal Set...
Src Power Correct
Interpolation
Correction Methods...
Correction Properties...

Port Extension

Select
Port Extension
Time
Distance
Velocity Factor
DC Loss
Port Extensions...
Auto Port Extension...

Cal Sets & Cal Kits

Cal Set...
Cal Set Viewer
Cal Kit...
Ecal

Show Connected ECals...
Restore ECal Memory..
ECal Confidence Check...
Characterize ECal...
Manage ECal Disk Memory...
Cal Pod...
Uncertainty Setup...

Fixtures

Apply Fixtures
Power Comp...
Fixture Setup...

  Port Match...
  Port Z...
  2-port DeEmbed...
Cal Plane Manager...
Auto Fixture Removal
DIQ Freq
Differential I/Q Freq
Main
  DIQ Setup...
DIQ Meas

Differential I/Q Meas

Meas

IPwrF1
OPwrF1
GainF1
Other...
Meas Class...

Meas Setup

Conversions
Correction
  Channel Correction On
  Channel Correction Off
  Cal Set...
  Cal All...
Trace Hold
Equation Editor
Memory
Time Domain
Pulse Setup
DIQ Power

Differential I/Q Power

Main
- Power Level
- RF Power
- Start Power
- Stop Power
- Power and Attenuators...
- DIQ Setup...

Port Power
- Select
- Power Level
- Start Power
- Stop Power
- Source State
- Coupling

Leveling & Offset
- Select
- Slope
- Offset
- Limit
- Offset and Limits...
- ALC Hardware (not available in M937xA/P937xA/M980xA/P50xxA)
- Receiver Leveling... (not available in M937xA/P937xA)

Attenuators
- Source Port 1, 2, 3, 4
- Receiver A, B, C, D
DIQ Setup

Layout

- New Trace
- New Channel
- New Window
- New Sheet
- Delete
- Select
- Measure
- Meas Class...

System Setup

- Next/Prev Keys: Select Next/Previous window/channel/trace for the selected one
- Preferences...
- Sound: Turn On/Off beep sound
- Remote Interface...
- LAN Status...
- Code Emulation

Internal Hardware

- RF Path Config...
- IF Path Config...
- Pulse Gen Config...
- Mechanical Devices...
- Interface Control...
- Reference (This feature applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above.)

External Hardware

- External Device...
- Power Meter Setup...
- External Testset...
- Multiport
- Millimeter Config...
DIQ Sweep

Differential IQ Sweep

Main

- Number of Points
- Sweep Type
- Start (Frequency sweep, Power Sweep)
- Stop (Frequency sweep, Power Sweep)
- X-axis Type
- DIQ Setup...

Sweep Timing

- Sweep Time
- Dwell Time
- Sweep Delay
- Sweep Mode
- Sweep Sequence
- Fast Sweep

Source Control

- Frequency Offset...
- Pulse Setup...
- Balanced Source...
- Phase Control...
- DC Source...
- Embedded LO...

Segment Table

- Add Segment, Insert Segment, Delete Segment, Delete All Segments
- Segment Table...
- Show Table
GCA_AvgBw
Gain Compression and Gain Compression Converters Avg BW

Main
  Averaging
  Averaging Restart
  Average Type
  IF BW
  LF Auto BW

Smoothing
  Smoothing
  Smooth Percent
  Smooth Points

Delay Aperture
  Aperture Percent
  Aperture Points
  Aperture Freq
GCA Freq
Gain Compression Freq

Main

- Start, Stop, Center, Span, Step
- CW
- Frequency Offset...
- GCA Setup...
GCA/GCX Cal

Gain Compression and Gain Compression Converters Cal

Main

Smart Cal...

GCA Cal...

Other Cals

Cal All...
Source Power Cal...

Correction
Correction On
Correction Off
Cal Set...
Src Power Correct
Interpolation
Correction Methods...
Correction Properties

Port Extension

Select
Port Extension
Time
Distance
Velocity Factor
DC Loss
Port Extensions...
Auto Port Extension...

Cal Sets & Cal Kits

Cal Set...
Cal Set Viewer
Cal Kit...
Ecal

Show Connected ECals...
Restore ECal Memory...
Cal Pod...
Uncertainty Setup...

**Fixtures**

Apply Fixtures
Power Comp...
Fixture Setup...

Port Match...
Port Z...
2-port DeEmbed...
Cal Plane Manager...
Auto Fixture Removal
GCA/GCX Sweep
Gain Compression and Gain Compression Converters Sweep

Main
- Number of Points
- Sweep Type
- Start (Frequency sweep, Power Sweep)
- Stop (Frequency sweep, Power Sweep)
- X-axis Type
- GCA Setup...
- GCX Setup... (M9485A does not support this)

Sweep Timing
- Sweep Time
- Dwell Time
- Sweep Delay
- Sweep Mode
- Sweep Sequence
- Fast Sweep

Source Control
- Frequency Offset...
- Pulse Setup...
- Balanced Source... (not available in M937xA/P937xA)
- Phase Control...
- DC Source...
- Global Source

Segment Table
- Add Segment, Insert Segment, Delete Segment, Delete All Segment
- Segment Table...
- Show Table
GCA Meas

Compression

Compln21/CompOut21
DeltaGain21
CompGain21
CompS11
RefS21
Other...
Meas Class...

S-Param

S11, S21, S12, S22
Other...

Auxiliary

AuxIn(N) Source Port (N)
CompAI1/CompAI2
Other...
AuxIn(N) Range

Meas Setup

Conversions
Correction
  Channel Correction On
  Channel Correction Off
  Cal Set...
  Smart Cal...
Trace Hold
Equation Editor
Memory
Time Domain
Pulse Setup
GCA/GCX Power
Gain Compression and Gain Compression Converters Power

Main
- Linear Input Pwr
- RF Power
- Start Power
- Stop Power
- Power and Attenuators...
- GCA Setup
- GCX Setup (M9485A does not support this)

Compress Levels
- Comp Method
- Linear Input Pwr
- Reverse Pwr
- Compression Level
- Back Off Level
- Delta X
- Delta Y
- Saturation

Leveling & Offset
- Select
- Slope
- Offset
- Limit
- Offset and Limits...
- ALC Hardware
- Receiver Leveling...

Attenuators
- Source Port 1, 2, 3, 4
- Receiver A, B, C, D
GCA/GCX Setup

Gain Compression Setup

Main
   GCA Setup...
   GCX Setup...
   Meas Class...
   Quick Start
   Device Expert...

Layout
   New Trace
   New Channel
   New Window
   New Sheet
   Delete
   Select
   Measure
   Meas Class...

System Setup
   Next/Prev Keys : Select Next/Previous window/channel/trace for the selected one
   Preferences...
   Sound: Turn On/Off beep sound
   Remote Interface...
   LAN Status...
   Code Emulation

Internal Hardware
   RF Path Config...
   IF Path Config...
   Pulse Gen Config...
   Mechanical Devices...
   Interface Control...
   Reference (This feature applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above.)

External Hardware
   External Device...
   Power Meter Setup...
   External Testset...
   Multiport
   Millimeter Config...
GCX Freq
Gain Compression Converters Freq

Main

- Start, Stop, Center, Span, Step
- CW
- Frequency Offset...
- GCX Setup...
GCX Meas
Gain Compression Converters Meas

Compression
  ComplIn21/CompOut21
  DeltaGain21
  CompGain21
  CompS11
  RefS21
  Other...
  Meas Class...

S-Param
  S11
  SC21
  SC12
  S22
  Other...

Power
  IPwr
  OPwr
  RevIPwr
  RevOPwr
  Other...

Auxiliary
  AuxIn(N) Source Port (N)
  CompAI1/CompAI2
  Other...

Meas Setup
  Conversions
  Correction
    Channel Correction On
    Channel Correction Off
    Cal Set...
    Smart Cal...
  Trace Hold
  Equation Editor
  Memory
  Time Domain
  Pulse Setup
IMD/IMDX AvgBW

Main
- Averaging
- Averaging Restart
- Average Type
- Main Tone IFBW
- IM Tone IFBW
- LF Auto BW

Smoothing
- Smoothing
- Smooth Percent
- Smooth Points

Delay Aperture
- Aperture Percent
- Aperture Points
- Aperture Freq
IMD/IMDX Cal

Swept IMD and Swept IMD Converters Cal

Main

Smart Cal...

IMD Cal...
IMDX Cal...

Other Cals

Cal All...
Correction
  Correction On
  Correction Off
  Cal Set...
Src Power Correct
Interpolation
Correction Methods...
Correction Properties...

Port Extension

Select
Port Extension
Time
Distance
Velocity Factor
DC Loss
Port Extensions...
Auto Port Extension...

Cal Sets & Cal Kits

Cal Set...
Cal Set Viewer
Cal Kit...
Ecal

Show Connected ECals...
Resto ECal Memory...
Cal Pod...
Uncertainty Setup...

Fixtures

Apply Fixtures
Power Comp...
Fixture Setup...

Port Match...
Port Z...
2-port DeEmbed...
Cal Plane Manager...
Auto Fixture Removal
IMD/IMDX Freq
Swept IMD and Swept IMD Converters Freq

Main

- Start, Stop, Center, Span, DeltaF
- Frequency Offset...
- IMD Setup...
- IMDX Setup...
IMD/IMDX Meas
Swept IMD and Swept IMD Converters Meas

IMD
  Main Tone Power
  IM3
  OIP3
  IIP3
  Other...
  Meas Class...

Meas Setup
  Conversions
  Correction
    Channel Correction On
    Channel Correction Off
    Cal Set...
    Smart Cal...
  Trace Hold
  Equation Editor
  Memory
  Time Domain
  Pulse Setup
IMD/IMDX Power
Swept IMD and Swept IMD Converters Power

Main
  Tone Power
  RF Power
  Start Power
  Stop Power
  Power and Attenuators...
  IMD Setup...
  IMDX Setup...

Port Power
  Select
  Power Level
  Start Power
  Stop Power
  Source State
  Coupling

Leveling & Offset
  Select
  Slope
  Offset
  Limit
  Offset and Limits...
  ALC Hardware
  Receiver Leveling...

Attenuators
  Source Port 1, 2, 3, 4
  Receiver A, B, C, D
IMD/IMDX Setup
Swept IMD and Swept IMD Converters Setup

Main
IMD Setup...
IMDX Setup...
Meas Class...
Quick Start
Device Expert...

Layout
New Trace
New Channel
New Window
New Sheet
Delete
Select
Measure
Meas Class...

System Setup
Next/Prev Keys: Select Next/Previous window/channel/trace for the selected one
Preferences...
Sound: Turn On/Off beep sound
Remote Interface...
LAN Status...
Code Emulation

Internal Hardware
RF Path Config...
IF Path Config...
Pulse Gen Config...
Mechanical Devices...
Interface Control...
Reference (This feature applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above.)

External Hardware
External Device...
Power Meter Setup...
External Testset...
Multiport
IMD/IMDX Sweep
Swept IMD and Swept IMD Converters Sweep

Main
- Number of Points
- Sweep Type
- Start
- Stop
- X-axis Type
- IMD Setup...
- IMDX Setup...

Sweep Timing
- Sweep Time
- Dwell Time
- Sweep Delay
- Sweep Mode
- Sweep Sequence
- Fast Sweep

Source Control
- Frequency Offset...
- Pulse Setup...
- Balanced Source...
- Phase Control...
- DC Source...
- Global Source

Segment Table
- Add Segment, Insert Segment, Delete Segment, Delete All Segment
- Segment Table...
- Show Table
IMS/IMSX Freq
IM Spectrum and IM Spectrum Converters Freq

Main

Fc
DeltaF
F1
F2
Start Spectrum
Stop Spectrum
Center Spectrum
Span Spectrum
IMS/IMSX Cal

Correction
  Correction On
  Correction Off
  Cal Set...
Src Power Correct
Interpolation
Correction Methods...
Correction Properties...

Port Extension
  Select
  Port Extension
  Time
  Distance
  Velocity Factor
  DC Loss
  Port Extensions...
  Auto Port Extension...

Cal Sets & Cal Kits
  Cal Set...
  Cal Set Viewer
  Cal Kit...
  Ecal

  Show Connected ECals...
  Restore ECal Memory...

Cal Pod...
Uncertainty Setup...

Fixtures
  Apply Fixtures
  Power Comp...
  Fixture Setup...

  Port Match...
  Port Z...
  2-port DeEmbed...
Cal Plane Manager...
Auto Fixture Removal
IMS/IMSX Power
IM Spectrum and IM Spectrum Converters Power

Main
  Tone Power
  RF Power
  Start Power
  Stop Power
  Power and Attenuators...
  IMS Setup...
  IMSX Setup...

Port Power
  Select
  Power Level
  Start Power
  Stop Power
  Source State
  Coupling

Leveling & Offset
  Select
  Slope
  Offset
  Limit
  Offset and Limits...
  ALC Hardware
  Receiver Leveling...

Attenuators
  Source Port 1, 2, 3, 4
  Receiver A, B, C, D
IMS/IMSX Sweep
IM Spectrum and IM Spectrum Converters Sweep

Main
- Number of Points
- Sweep Type
- Start
- Stop
- X-axis Type
- IMS Setup...
- IMSX Setup...

Sweep Timing
- Sweep Time
- Dwell Time
- Sweep Delay
- Sweep Mode
- Sweep Sequence
- Fast Sweep

Source Control
- Frequency Offset...
- Pulse Setup...
- Balanced Source...
- Phase Control...
- Embedded LO...
- DC Source...
- Global Source

Segment Table
- Add Segment, Insert Segment, Delete Segment, Delete All Segments
- Segment Table...
- Show Table
IMS Meas

IM Spectrum Meas

Spectrum

  Output Spectrum
  Input Spectrum
  Reflected Spectrum
  Other...
  Meas Class...

Meas Setup

  Conversions
  Correction
    Channel Correction On
    Channel Correction Off
    Cal Set...
    Cal All...
  Trace Hold
  Equation Editor
  Memory
  Time Domain
  Pulse Setup
IMS/IMSX Setup

IM Spectrum and IM Spectrum Converters Setup

Main
- IMS Setup...
- IMSX Setup...
- Meas Class...
- Quick Start
- Device Expert...

Layout
- New Trace
- New Channel
- New Window
- New Sheet
- Delete
- Select
- Measure
- Meas Class...

System Setup
- Next/Prev Keys: Select Next/Previous window/channel/trace for the selected one
- Preferences...
- Sound: Turn On/Off beep sound
- Remote Interface...
- LAN Status...
- Code Emulation

Internal Hardware
- RF Path Config...
- IF Path Config...
- Pulse Gen Config...
- Mechanical Devices...
- Interface Control...
- Reference (This feature applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above.)

External Hardware
- External Device...
- Power Meter Setup...
- External Testset...
- Multiport
IMSX Meas

IM Spectrum Converters Meas

Spectrum
  Output Spectrum
  Other...
  Meas Class...

Meas Setup
  Conversions
  Correction
    Channel Correction On
    Channel Correction Off
    Cal Set...
    Cal All...
  Trace Hold
  Equation Editor
  Memory
  Time Domain
  Pulse Setup
Modulation Distortion AvgBW

Main
  Noise BW

Smoothing
  Smoothing
  Smooth Percent
  Smooth Points

Delay Aperture
  Aperture Percent
  Aperture Points
  Aperture Freq
Modulation Distortion Cal

Main

Cal All...

Other Cals

Source Modulation Cal...
LO Feedthru Monitor...
Receiver IF Cal...
Phase Reference Wizard...

VNA Correction

Channel Correction On
Channel Correction Off
Cal Set...
Src Mod Correct ON | OFF
Interpolation
Correction Methods...
Correction Properties

Port Extension

Select
Port Extension
Time
Distance
Velocity Factor
DC Loss
Port Extensions...
Auto Port Extension...

Cal Sets & Cal Kits

Cal Set...
Cal Set Viewer
Cal Kit...

Ecal

Show Connected ECals...
Restore ECal Memory...
Cal Pod...
Uncertainty Setup...

Fixtures

Apply Fixtures
Power Comp...
Fixture Setup...

Port Match...
Port Z...
2-port DeEmbed...
Cal Plane Manager...
Auto Fixture Removal
Modulation Distortion Freq

SA Frequency
  - Start, Stop, Center, Span
  - Span All Meas Bands
  - MOD Setup...
  - MODX Setup...

Modulated Source
  - Fixed Carrier
  - Source Modulation
  - Pulse Setup...
  - MOD Setup...
  - MODX Setup...
Modulation Distortion Meas

Main
- PIn1
- POut2
- MSig2
- S21
- Other...
- Distortion Table...
- MOD Setup...
- MODX Setup...
- Meas Class...

Auxiliary
- AuxIn(N) Source Port (N)
- Other...

Meas Setup
- Conversions
- Correction
  - Channel Correction On
  - Channel Correction Off
- Cal Set...
- Cal All...
  - Src Mod Cal...
- Trace Hold
- Equation Editor
- Memory
- Time Domain
- Pulse Setup
Modulation Distortion Power

Main
- Carrier Level
- RF Power ON | OFF
- S-Param Input Pwr
- Power and Attenuators...
- MOD Setup...
- MODX Setup...

Leveling & Offset
- Offset
- Limit
- Offset and Limits...
- Receiver Leveling...

Attenuators
- Source Port 1, 2, 3, 4
- Receiver A, B, C, D
Modulation Distortion Setup

Main

MOD Setup...
MODX Setup...
Meas Class...
Quick Start
Device Expert...

Layout

New Trace
New Channel
New Window
New Sheet
Delete
Select
Measure
Meas Class...

System Setup

Next/Prev Keys: Select Next/Previous window/channel/trace for the selected one
Preferences...
Sound: Turn On/Off beep sound
Remote Interface...
LAN Status...
Code Emulation

Internal Hardware

RF Path Config...
IF Path Config...
Pulse Gen Config...
Mechanical Devices...
Interface Control...
Reference (This feature applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above.)

External Hardware

External Device...
Power Meter Setup...
Multiport
Millimeter Config...
Modulation Distortion Sweep

Main
- Number of Points
- Sweep Type
- Start (Frequency sweep, Power Sweep)
- Stop (Frequency sweep, Power Sweep)
- X-axis Type...
- MOD Setup...
- MODX Setup...

Sweep Timing
- Sweep Delay
- Dist Meas Delay
- Re-use S-Param (MOD only)

Source Control
- Source Modulation...
- Pulse Setup...
- DC Source...
NF AvgBW

Noise Figure Cold Source and Noise Figure Converters AvgBW

Main
  Averaging
  Averaging Restart
  Average Type
  IF BW
  LF Auto BW
  Noise Avg
  Noise BW

Smoothing
  Smoothing
  Smooth Percent
  Smooth Points

Delay Aperture
  Aperture Percent
  Aperture Points
  Aperture Freq
NF Freq

Noise Figure Cold Source Freq

Main

- Start, Stop, Center, Span, Step
- CW
- Frequency Offset...
- NF Setup...
NF Meas

Noise Figure Cold Source Meas

Noise

NF
T-Eff
ENR
NFmin
GammaOpt
Rn
Other...
Meas Class...

Incident Noise Power

Incident DUT Relative NPwr
Incident DUT NPwr Density
Incident Syst Relative NPwr
Incident Syst NPwr Density
Other...

Available Noise Power

Available DUT Relative NPwr
Available DUT NPwr Density
Available Syst Relative NPwr
Available Syst NPwr Density
Other...

Noise Correlation

NCorr_11
NCorr_12
NCorr_21
NCorr_22
Other...

S-Param

S11
S21
S12
S22
Other...

Receivers

A Source Port 1/2
B Source Port 1/2
R1 Source Port 1
R2 Source Port 2
Other...

Meas Setup

Conversions
Correction
  Channel Correction On
  Channel Correction Off
  Cal Set...
  Smart Cal...
Trace Hold
Equation Editor
Memory
Time Domain
Pulse Setup
NF/NFX Cal

Noise Figure Cold Source and Noise Figure Converters Cal

Main
  Smart Cal...

  Noise Figure Cal

Other Cals
  Cal All...
  Source Power Cal...
Correction
  Correction On
  Correction Off
  Cal Set...
  Src Power Correct
  Interpolation
  Correction Methods...
  Correction Properties

Port Extension
  Select
  Port Extension
  Time
  Distance
  Velocity Factor
  DC Loss
  Port Extensions...
  Auto Port Extension...

Cal Sets & Cal Kits
  Cal Set...
  Cal Set Viewer
  Cal Kit...
  Ecal

  Show Connected ECals...
Restore ECAL Memory...
Cal Pod...
Uncertainty Setup...

Fixtures
Apply Fixtures
Power Comp...
Fixture Setup...

Port Match...
Port Z...
2-port DeEmbed...
Cal Plane Manager...
Auto Fixture Removal
NF/NFX Power
Noise Figure Cold Source and Noise Figure Converters Power

Main
- Power Level
- RF Power
- Start Power
- Stop Power
- Noise Source
- Power and Attenuators...
- NF Setup...
- NFX Setup...

Port Power
- Select
- Power Level
- Start Power
- Stop Power
- Source State
- Coupling

Leveling & Offset
- Select
- Slope
- Offset
- Limit
- Offset and Limits...
- ALC Hardware
- Receiver Leveling...

Attenuators
- Source Port 1, 2, 3, 4
- Receiver A, B, C, D
NF/NFX Setup

Noise Figure Cold Source and Noise Figure Converters Setup

Main
- NF Setup...
- NFX Setup...
- Meas Class...
- Quick Start
- Device Expert...

Layout
- New Trace
- New Channel
- New Window
- New Sheet
- Delete
- Select
- Measure
- Meas Class...

System Setup
- Next/Prev Keys: Select Next/Previous window/channel/trace for the selected one
- Preferences...
  - Sound: Turn On/Off beep sound
- Remote Interface...
- LAN Status...
- Code Emulation

Internal Hardware
- RF Path Config...
- IF Path Config...
- Mechanical Devices...
- NF Rcvr Gain (30 High, 15: Medium, 0: Low)
- Pulse Gen Config...
- Interface Control...
- Reference (This feature applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above.)

External Hardware
- External Device...
- Power Meter Setup...
- External Testset...
- Multiport
Millimeter Config...
NF/NFX Sweep

Main (Noise Figure Cold Source and Noise Figure Converters)

- Number of Points
- Sweep Type
- Start
- Stop
- X-axis Type
- NF Setup...
- NFX Setup...

Sweep Timing

- Sweep Time
- Dwell Time
- Sweep Delay
- Sweep Mode
- Sweep Sequence
- Fast Sweep

Source Control

- Frequency Offset...
- Pulse Setup...
- Balanced Source... (not available in M937xA/P937xA)
- Phase Control...
- DC Source...
- Embedded LO...
- Global Source

Segment Table

- Add Segment, Insert Segment, Delete Segment, Delete All Segment
- Segment Table...
- Show Table
NFX Freq

Noise Figure Converters Freq

Main

Start, Stop, Center, Span, Step
CW
Frequency Offset...
NFX Setup...
NFX Meas
Noise Figure Converters Meas

Noise
  NF
  T-Eff
  ENR
  NFmin
  GammaOpt
  Rn
  Other...
  Meas Class...

Incident Noise Power
  Incident DUT Relative NPwr
  Incident DUT NPwr Density
  Incident Syst Relative NPwr
  Incident Syst NPwr Density
  Other...

Available Noise Power
  Available DUT Relative NPwr
  Available DUT NPwr Density
  Available Syst Relative NPwr
  Available Syst NPwr Density
  Other...

Noise Correlation
  NCorr_11
  NCorr_12
  NCorr_21
  NCorr_22
  Other...

S-Param
  S11
  SC21
  SC12
  S22
  Other...

Power
  IPwr
OPwr
RevIPwr
RevOPwr
Other...

Receivers
A Source Port 1/2
B Source Port 1/2
R1 Source Port 1
R2 Source Port 2
Other...

Meas Setup
Conversions
Correction
    Channel Correction On
    Channel Correction Off
    Cal Set...
    Smart Cal...
Trace Hold
Equation Editor
Memory
Time Domain
Pulse Setup
PN AvgBW
Phase Noise Avg BW

Main
  Averaging
  Averaging Restart
  RBW Ratio
  FFT Avg Factor
  Noise Mode

Smoothing
  Smoothing
  Smooth Percent
  Smooth Points
PN Cal
Phase Noise Cal

Main
  Cal All...
  Correction
    Correction On
    Correction Off
  Cal Set...
  Src Power Correct
  Interpolation
  Correction Methods...
  Correction Properties

Port Extension
  Select
  Port Extension
  Time
  Distance
  Velocity Factor
  DC Loss
  Port Extensions...
  Auto Port Extension...

Cal Sets & Cal Kits
  Cal Set...
  Cal Set Viewer
  Cal Kit...
  Ecal

  Show Connected ECals...
  Restore ECal Memory...

  Cal Pod...
  Uncertainty Setup...

Fixtures
  Apply Fixtures
  Power Comp...
  Fixture Setup...
Port Match...
Port Z...
2-port DeEmbed...
Cal Plane Manager...
Auto Fixture Removal
PN Freq
Phase Noise Freq

Main
  Start Offset
  Stop Offset
  Carrier
  Source Frequency...
  Phase Noise Setup...
PN Meas

Phase Noise Meas

Main

PN_b2
AM_b2
Other...
VNA Input
Noise Type
Phase Noise Setup...
Meas Class...

Spurious & Noise Tables

Spurious Table...
Spot Noise Table...
Integrated Noise Table...

Meas Setup

Conversions
Correction
  Correction On
  Correction Off
Cal Set...
  Cal All...
  Src Mod Cal...
Trace Hold
Equation Editor
Memory
Time Domain
Pulse Setup
PN Power

Phase Noise Power

Main
  - Power Level
  - RF Power
  - Power and Attenuators...
  - Phase Noise Setup...

Port Power
  - Select
  - Power Level
  - Start Power
  - Stop Power
  - Source State
  - Coupling

Attenuators
  - Source Port 1, 2, 3, 4
  - Receiver A, B, C, D
PN Setup

Phase Noise Setup

Main

- Phase Noise Setup...
- Meas Class...
- Quick Start
- Device Expert

Layout

- New Trace
- New Channel
- New Window
- New Sheet
- Delete
- Select
- Measure
- Meas Class...

System Setup

- Next/Prev Keys: Select Next/Previous window/channel/trace for the selected one
- Preferences...
- Sound: Turn On/Off beep sound
- Remote Interface...
- LAN Status...
- Code Emulation

Internal Hardware

- RF Path Config...
- IF Path Config...
- Mechanical Devices...
- Interface Control...

Reference (This feature applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above.)

External Hardware

- External Device...
- Power Meter Setup...
- Multiport
PN Sweep

Phase Noise Sweep

Main

- Number of Points
- Sweep Type
- Start
- Stop
- X-axis Type
- Phase Noise Setup...

Sweep Timing

- Sweep Time
- Dwell Time
- Sweep Delay
- Sweep Mode
- Sweep Sequence
- Fast Sweep

Source Control

- Pulse Setup...
- DC Source...
- Embedded LO...
- Global Source
SA Avg BW

Main
- Averaging
- Averaging Restart
- Vector Avg
- Res BW
- Video BW
- Detector Type
- Video Avg Type

Smoothing
- Smoothing
- Smooth Percent
- Smooth Points

Delay Aperture
- Aperture Percent
- Aperture Points
- Aperture Freq

SA Filters
- RBW Shape
- RBW / VBW
- Span / RBW
- Image Reject
- ADC Filter

SA Receivers
- Select Receiver
- Receiver Atten
- IF Gain
- Receiver Attenuators...
- IF Path Config...
SA Cal

Correction
  Correction On
  Correction Off
  Cal Set...
Src Power Correct
Interpolation
Correction Methods...
Correction Properties

Port Extension
  Select
  Port Extension
  Time
  Distance
  Velocity Factor
  DC Loss
  Port Extensions...
  Auto Port Extension...

Cal Sets & Cal Kits
  Cal Set...
  Cal Set Viewer
  Cal Kit...
  Ecal

  Show Connected ECals...
  Restore ECal Memory...
  ECal Confidence Check...
  Characterize ECal...
  Manage ECal Disk Memory...

  Cal Pod...
  Uncertainty Setup...

Fixtures
  Apply Fixtures
  Power Comp...
  Fixture Setup...
Port Match...
Port Z...
2-port DeEmbed...
N-port DeEmbed...
Differential Port Match...
Differential Z...
Common Z...
Cal Plane Manager...
Auto Fixture Removal
SA Freq
Spectrum Analyzer Freq

Main
  Start, Stop, Center, Span
  CF Step
  Full Span
  Auto Tune
  SA Setup...

Source Frequency
  Port X...
  Port 1 Src 2
  Source3

Source IQ Mod.
  MXG (the button displays the name assigned to the source)
SA Meas

Spectrum Analyzer Meas

Receivers
A, B, C, D
R1, R2, R3, R4
Other...
Meas Class...

Waves
a1, a2, a3, a4
b1, b2, b3, b4
Other...

Meas Setup
Conversions
Correction
Channel Correction On
Channel Correction Off
Cal Set...
Cal All...
Trace Hold
Equation Editor
Memory
Time Domain
Pulse Setup
SA Power
Spectrum Analyzer Power

Main
- Power Level
- RF Power
- Port N
- Power and Attenuators...
- SA Setup...

Port Power
- Select
- Power Level
- Start Power
- Stop Power
- Step Power
- Source State
- Coupling

Leveling & Offset
- Select
- Slope
- Offset
- Limit
- Offset and Limits...
- ALC Hardware (not available in M937xA/P937xA/M980xA/P50xxA)
- Receiver Leveling... (not available in M937xA/P937xA)

Attenuators
- Source Port 1, 2, 3, 4
- Receiver A, B, C, D

Source
- Port X
- Port 1 Src 2
- Source3
SA Setup

Layout

- New Trace
- New Channel
- New Window
- New Sheet
- Delete
- Select
- Measure
- Meas Class...

System Setup

- Next/Prev Keys: Select Next/Previous window/channel/trace for the selected one
- Preferences...
- Sound: Turn On/Off beep sound
- Remote Interface...
- LAN Status...
- Code Emulation

Internal Hardware

- RF Path Config...
- IF Path Config...
- Mechanical Devices...
- Interface Control...
- Reference (This feature applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above.)

External Hardware

- External Device...
- Power Meter Setup...
- External Testset...
- Multiport
- Millimeter Config...
SA Sweep
Spectrum Analyzer Sweep

Main
  Number of Points
  Sweep Type
  Start (Frequency sweep, Power Sweep)
  Stop (Frequency sweep, Power Sweep)
  X-axis Type
  SA Setup...

Sweep Timing
  Sweep Time
  Dwell Time
  Sweep Delay
  Sweep Mode
  Sweep Sequence
  Fast Sweep

Source Control
  Frequency Offset...
  Pulse Setup...
  Balanced Source...
  Phase Control...
  DC Source...
  Global Source

Segment Table
  Add Segment, Insert Segment, Delete Segment, Delete All Segments
  Segment Table...
  Show Table

SA Coherence
  SA Multitone
  Tone Spacing
  Reference Tone
  Data Display
  Detector Bypass
SMC Meas
Scalar Mixer/Converter + Phase Meas

S-Param
- S11
- SC21
- SC12
- S22
- Other...
- Meas Class...

Power
- IPwr
- OPwr
- RevIPwr
- RevOPwr
- Other...

Auxiliary
- AuxIn(N) Source Port (N)
- Other...
- AuxIn(N) Range

Meas Setup
- Conversions
- Correction
  - Channel Correction On
  - Channel Correction Off
- Cal Set...
  - Smart Cal...
- Trace Hold
- Equation Editor
- Memory
- Time Domain
- Pulse Setup
SMC/VMC Cal

Other Cals

    Cal All...
    Source Power Cal...
    Mixer Char Wizard...
    Phase Reference Wizard...

Correction

    Correction On
    Correction Off

    Cal Set...

    Src Power Correct

Interpolation

    Correction Methods

    Correction Properties

Port Extension

    Select
    Port Extension
    Time
    Distance
    Velocity Factor
    DC Loss
    Port Extensions...
    Auto Port Extension...

Cal Sets & Cal Kits

    Cal Set...
    Cal Set Viewer
    Cal Kit...

    Ecal

    Show Connected ECals...
    Restore ECal Memory...

Cal Pod...

    Uncertainty Setup...

Fixtures

    Apply Fixtures
Power Comp...
Fixture Setup...

Port Match...
Port Z...
2-port DeEmbed...
Cal Plane Manager...
Auto Fixture Removal
SMC/VMC Freq
Scalar Mixer/Converter + Phase and Vector Mixer/Converter Freq

Main
- Start, Stop, Center, Span, Step
- CW
- Frequency Offset...
- SMC Setup...
- VMC Setup...
SMC/VMC Power
Scalar Mixer/Converter + Phase and Vector Mixer/Converter Power

Main
- Power Level
- RF Power
- Start Power
- Stop Power
- Power and Attenuators...
- SMC Setup...
- VMC Setup...

Port Power
- Select
- Power Level
- Start Power
- Stop Power
- Source State
- Coupling

Leveling & Offset
- Select
- Slope
- Offset
- Limit
- Offset and Limits...
- ALC Hardware
- Receiver Leveling...

Attenuators
- Source Port 1, 2, 3, 4
- Receiver A, B, C, D
SMC/VMC Setup

Scalar Mixer/Converter + Phase and Vector Mixer/Converter Setup

Main

SMC Setup...
VMC Setup...
Meas Class...
Quick Start
Device Expert...

Layout

New Trace
New Channel
New Window
New Sheet
Delete
Select
Measure
Meas Class...

System Setup

Next/Prev Keys: Select Next/Previous window/channel/trace for the selected one
Preferences...

Sound: Turn On/Off beep sound
Remote Interface...
LAN Status...
Code Emulation

Internal Hardware

RF Path Config...
IF Path Config...
Pulse Gen Config...
Mechanical Devices...
Interface Control...

Reference (This feature applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above.)

External Hardware

External Device...
Power Meter Setup...
External Testset...
Multiport
Millimeter Config...
SMC/VMC Sweep

Scalar Mixer/Converter + Phase and Vector Mixer/Converter Sweep

Main

- Number of Points
- Sweep Type
- Start
- Stop
- X-axis Type
- SMC Setup...
- VMC Setup...

Sweep Timing

- Sweep Time
- Dwell Time
- Sweep Delay
- Sweep Mode
- Sweep Sequence
- Fast Sweep

Source Control

- Frequency Offset...
- Pulse Setup...
- Balanced Source...
- Phase Control...
- DC Source...
- Embedded LO...
- Global Source

Segment Table

- Add Segment, Insert Segment, Delete Segment, Delete All Segments
- Segment Table...
- Show Table
Standard Avg BW

Standard / Differential IQ / IM Spectrum / Scalar Mixer/Converter + Phase and Vector Mixer/Converter  Avg BW

Main

  Averaging
  Averaging Restart
  Average Type
  IF BW
  IF Bandwidth... (DIQ only)
  Resolution BW (IM Spectrum only)
  LF Auto BW

Smoothing

  Smoothing
  Smooth Percent
  Smooth Points

Delay Aperture

  Aperture Percent
  Aperture Points
  Aperture Freq
Standard Cal

Main

Smart Cal...
Other Cals

Cal All...
Basic Cal...
E Cal...
Receiver Power Cal...
Response Cal... (not available in M937xA and E5080A)
Source Power Cal...

Correction
Correction On
Correction Off
Cal Set...

Factory Cal
Src Power Correct
Interpolation
Correction Methods...
Correction Properties...

Port Extension

Select
Port Extension
Time
Distance
Velocity Factor
DC Loss
Port Extensions...
Auto Port Extension...

Cal Sets & Cal Kits

Cal Set...
Cal Set Viewer
Cal Kit...
Ecal

Show Connected ECals...
Restore ECal Memory...
ECal Confidence Check...
Characterize ECal...
Manage ECal Disk Memory...
Cal Pod...
Uncertainty Setup...

**Fixtures**

Apply Fixtures
Power Comp...
Fixture Setup...

Port Match...
Port Z...
2-port DeEmbed...
N-port DeEmbed...
Differential Port Match...
Differential Z...
Common Z...
Cal Plane Manager...
Auto Fixture Removal
Standard Freq

Main

  Start, Stop, Center, Span, Step
  CW
  Frequency Offset...
Standard Meas

S-param

Sxy
Other...
Meas Class...

Balanced

Sssxy, Sdsxy, Ssdxy, Sddxy
Other...
Topology

Receivers

A-D/R1-4 Source Port 1-4
Other...

Waves

a1-4/b1-4 Source Port 1-4
Other...

Auxiliary

AuxIn(N) Source Port (N)
AuxIn(N) Range
Other...

Meas Setup

Conversions
Correction
Channel Correction On
Channel Correction Off
Cal Set...
Basic Cal...
Smart Cal...
Trace Hold
Equation Editor
Memory
Time Domain
Pulse Setup
Standard Power

Main
- Power Level
- RF Power
- Start Power
- Stop Power
- Power and Attenuators...

Port Power
- Select
- Power Level
- Start Power
- Stop Power
- Source State
- Coupling

Leveling & Offset
- Select
- Slope
- Offset
- Limit
- Offset and Limits...
- ALC Hardware (not available in M937xA/P937xA/M980xA/P50xxA)
- Receiver Leveling... (not available in M937xA/P937xA)

Attenuators
- Source Port 1, 2, 3, 4
- Receiver A, B, C, D
Standard Setup

Main
  - Sweep Setup...
  - Meas Class...
  - Quick Start
  - Device Expert...

Layout
  - New Trace
  - New Channel
  - New Window
  - New Sheet
  - Delete
  - Select
  - Measure
  - Meas Class...

System Setup
  - Next/Prev Keys: Select Next/Previous window/channel/trace for the selected one
  - Preferences...
  - Sound: Turn On/Off beep sound
  - Remote Interface...
  - LAN Status...
  - Code Emulation

Internal Hardware
  - RF Path Config...
  - IF Path Config...
  - Pulse Gen Config...
  - Mechanical Devices...
  - Interface Control...
  - Reference (This feature applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above.)

  - LF Extension ON/OFF

External Hardware
  - External Device...
  - Power Meter Setup...
  - External Testset...
  - Multiport
  - Millimeter Config...
Standard Sweep

Main

  Number of Points
  Sweep Type

Start (Frequency sweep, Power Sweep)

Stop (Frequency sweep, Power Sweep)

X-axis Type

  Sweep Setup...

Sweep Timing

  Sweep Time
  Dwell Time
  Sweep Delay
  Sweep Mode
  Sweep Sequence
  Fast Sweep

Source Control

  Frequency Offset...
  Pulse Setup... (not available in M937xA/P937xA)
  Balanced Source... (not available in M937xA/P937xA)
  Phase Control...
  DC Source...
  LF Extension ON/OFF
  Global Source

Segment Table

  Add Segment, Insert Segment, Delete Segment, Delete All Segments
  Segment Table...
  Show Table
VMC Meas
Vector Mixer/Converter Meas

Main
- S11
- VC21
- S22
- R1
- B
- Other...
- Meas Class...

Meas Setup
- Conversions
- Correction
  - Channel Correction On
  - Channel Correction Off
- Cal Set...
- Smart Cal...
- Trace Hold
- Equation Editor
- Memory
- Time Domain
- Pulse Setup
IF Access Overview

Applications that use IF Path Configuration

- External Millimeter Module Configuration
- FIFO and other Antenna Features
- N5264B

PNA-X and N522x Opt 020 ONLY

- IF Path Configuration User Interface
- PNA-X and N522x rear panel connectors

Programming commands

- SCPI Commands
- COM Commands

IFConfiguration Object
PathConfiguration Object

Notes:

- Opt S93026A/B is available only on PNA-X and N522x models
- See VNA Configuration and options
Millimeter Wave Configuration

Overview
Supported Applications
Supported Configurations
Starting and Exiting Millimeter Wave Mode
Millimeter Configuration
Calibration
Broadband System Measurement Setup
Banded System Measurement Setup
Mixer Measurements
Remote Programming Commands
Accessing Data

Click a box or circle to view details:

See 8510 data processing mode.
See larger Data Processing map.

Measurement - Receivers gather complex trace data which is ratioed if required by the parameter, such as S11 or A/B. Otherwise it is raw receiver data, such as A or B. See Measurement Parameters.

Averaging - If turned ON, data is averaged with specified number of measurement traces. See Averaging.

Data Access Point 0 - Get or Put RAW MEASUREMENT data using:

SCPI - Write data using Calc:Data SDATA - Read data from Data Access Point 1

COM - getData and putDataComplex - naRawData (0)

Formatting Note:

COM - getData allows you to request data from locations 0 to 5 in a format other than the displayed format. SMOOTHED data is only attainable from locations 2 & 4, and only when you request data in the same format as the displayed format.

SCPI: you can only request data in the displayed format.

Acquired Cal Data - Calibration standards are measured. When the calibration is complete, complex data is stored in a Cal Set.

Data Access Point 6 - Get or Put RAW CAL ACQUISITION data using:

SCPI - None

COM - getStandardComplex and putStandardComplex

Calculate Error Terms - Error terms are calculated from Acquisition data using formulas which are appropriate for the selected calibration method. Complex error terms are stored in a Cal Set. See Systematic Errors.

Data Access Point 5 - Get or Put ERROR TERM data using:

SCPI - Calc:Data Error Terms

COM - Error Term commands

Note: Normalization, formerly access location 5, no longer exists and was used ONLY by Receiver Power Cal. That cal type now uses Acquire Cal Data and Calculate Error Terms like all other Cals.
Apply Error Terms - If error correction is ON, error terms are applied to the raw measurement data. Otherwise, this data is identical to Raw Measurement Data. In addition, the Fixture Simulator functions occur at the same time as the Apply Error Terms block.

**Data Access Point 1** - Get or Put CORRECTED data using:

- **SCPI** - Read data using **Calc:Data SDATA** - Write data to Data Access Point 0
- **COM** - **getData** and **putDataComplex** - naCorrectedData (1)

See Formatting Note

---

**Equation Editor** - Allows custom equations to perform advanced math operations between data traces. See Equation Editor Notes:

- **SCPI**: Calc:Data
- **COM**: Get Data Method or Get DataByString Method.

---

**Normalization** - No longer available

**Data Access Point 5** - No longer available

---

**Trace Math** - When turned ON, memory data is combined with measurement data using the selected math function. Available functions are: Data+Mem, Data-Mem, Data*Mem, and Data/Mem. See Math Operations.

---

**Memory** - Data that is stored as a result of a Data-To-Memory operation. Each measurement can have one memory trace. The memory data parallels the measurement data through the remaining processing blocks. For example, turning smoothing ON will smooth both the measurement and memory traces.

**Data Access Point 3** - Get or Put MEMORY data using:

- **SCPI** - Calc:Data SMEM
- **COM** - **getData** and **putDataComplex** - naRawMemory (3)

See Formatting Note

---

**Gating** - When turned ON, Filter Gating is applied to the measurement data. Gating "virtually" removes undesired responses from selected regions of the trace. See Gating.

---

**Phase Correction** - When turned ON, applies electrical delay, phase offset, and port extensions. These are all separate features that are controlled individually.
**Magnitude Offset** - When entered, offset values are applied to the magnitude (real) portion of the data. See Magnitude Offset.

**Time Domain** - When turned ON, transforms the data from the frequency domain to the time domain. See Time Domain

**Formatter** - Complex data is converted into scalar data formats for screen display and remote access. For smoothed data, request the data in the same format as the displayed data. See Data Format

**Smoker** - When turned ON, removes discontinuities in the measurement and memory trace. See Smoothing.

**Display** - Displays the processed measurement, memory data, or both, in the format of your choice.

**Data Access Point 2** - Get or Put MEAS RESULT data using:

- **SCPI** - `Calc:Data FDATA`
- **COM** - `getData` and `putDataComplex` - `naMeasResult (2)`

**Data Access Point 4** - Get or Put MEMORY RESULT data using:

- **SCPI** - `Calc:Data FMEM`
- **COM** - `getData` and `putDataComplex` - `naMemoryResult (4)`

See Formatting Note
<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perform an Unguided Calibration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Launch Cal Wizard</td>
<td>SYSTem:CORRection:WIZard</td>
<td>app.LaunchCalWizard</td>
</tr>
<tr>
<td>Set Cal Type</td>
<td>SENSe:CORRection:COLLect:METHod</td>
<td>cal.SetCallInfo</td>
</tr>
<tr>
<td>Select a Cal Kit</td>
<td>SENSe:CORRection:COLLect:CKIT</td>
<td>app.CalKitType</td>
</tr>
<tr>
<td>Get a Handle to the Active Cal Kit</td>
<td>None</td>
<td>app.ActiveCalKit</td>
</tr>
<tr>
<td>Simultaneous 2-Port Calibration</td>
<td>SENSe:CORRection:TSTandards</td>
<td>cal.Simultaneous2PortAcquisition</td>
</tr>
<tr>
<td>Acquisition Direction</td>
<td>SENSe:CORRection:SFORward</td>
<td>cal.AcquisitionDirection</td>
</tr>
<tr>
<td>Measure a Standard</td>
<td>SENSe:CORRection:COLLect</td>
<td>cal.AcquireCalStandard</td>
</tr>
<tr>
<td>Calculate Errors</td>
<td>SENSe:CORRection:COLLect:SAVE</td>
<td>cal.CalculateErrorCoeffecients</td>
</tr>
<tr>
<td>Do Isolation</td>
<td>SENSe:CORRection:COLLect</td>
<td>cal.AcquireCalStandard</td>
</tr>
<tr>
<td>Perform and apply Response (Normalization) cal</td>
<td>SENSe:CORRection:COLLect:METHod</td>
<td>DoResponseCal</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perform a Guided Cal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Initiate a Guided Cal</td>
<td>SENSe:CORRection:COLLect:GUIDed:INITiate</td>
<td>Initialize</td>
</tr>
<tr>
<td>Operation</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>Select a Connector Type</td>
<td>SENSE:CORRection:COLLect:GUIDed:CONNector:PORT</td>
<td>ConnectorType</td>
</tr>
<tr>
<td>Select a Cal Kit</td>
<td>SENSE:CORRection:COLLect:GUIDed:CKIT:PORT</td>
<td>CalKitType</td>
</tr>
<tr>
<td>Set cal method for each port pair.</td>
<td>SENSE:CORRection:COLLect:GUIDed:PATH:CMETHod</td>
<td>PathCalMethod</td>
</tr>
<tr>
<td>Set Thru Method for each port pair.</td>
<td>SENSE:CORRection:COLLect:GUIDed:PATH:TMEThod</td>
<td>PathThruMethod</td>
</tr>
<tr>
<td>Set Thru Port Pairs</td>
<td>SENSE:CORRection:COLLect:GUIDed:THRU:PORTs</td>
<td>ThruPortList</td>
</tr>
<tr>
<td>Return Number of Steps in a Cal</td>
<td>SENSE:CORRection:COLLect:GUIDed:STEPs?</td>
<td>GenerateSteps</td>
</tr>
<tr>
<td>Return a Description of a Cal Step</td>
<td>SENSE:CORRection:COLLect:GUIDed:DESCription?</td>
<td>GetStepDescription</td>
</tr>
<tr>
<td>Save Cal</td>
<td>SENSE:CORRection:COLLect:GUIDed:SAVE</td>
<td>GenerateErrorTerms</td>
</tr>
<tr>
<td>Return Number of Steps in a Cal</td>
<td>SENSE:CORRection:COLLect:GUIDed:LIST:COUNt?</td>
<td>None</td>
</tr>
<tr>
<td>Return number of ports on standard used in the step</td>
<td>SENSE:CORRection:COLLect:GUIDed:LIST:STEP:PORTs?</td>
<td>None</td>
</tr>
<tr>
<td>Return label for one of the standards in the step</td>
<td>SENSE:CORRection:COLLect:GUIDed:LIST:STEP:STANdard:LABel?</td>
<td>None</td>
</tr>
<tr>
<td>Return number of ports on one of the standards</td>
<td>SENSE:CORRection:COLLect:GUIDed:LIST:STEP:STANdard:PORTs?</td>
<td>None</td>
</tr>
<tr>
<td>standards used in the step</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Return the enumeration for the type of standard</td>
<td>SENSE:CORRection:COLLect:GUIDed:LIST:STEP:STANdard:STYPe?</td>
<td></td>
</tr>
<tr>
<td>Return list of VNA test ports to which one of the standards is attached</td>
<td>SENSE:CORRection:COLLect:GUIDed:LIST:STEP:STANdard:TPORts?</td>
<td></td>
</tr>
<tr>
<td>Return enumeration for the type of standard device used in the step</td>
<td>SENSE:CORRection:COLLect:GUIDed:LIST:STEP:STYPe?</td>
<td></td>
</tr>
<tr>
<td>Return list of VNA test ports to which the standard(s) in this step is attached</td>
<td>SENSE:CORRection:COLLect:GUIDed:LIST:STEP:TPORts?</td>
<td></td>
</tr>
<tr>
<td>Return measurement parameters measured in the specified step number of a guided calibration</td>
<td>SENSE:CORRection:COLLect:GUIDed:DATA:CATalog?</td>
<td></td>
</tr>
<tr>
<td>Set and return measurement data for a specified measurement parameter of a particular step of a guided cal</td>
<td>SENSE:CORRection:COLLect:GUIDed:DATA</td>
<td></td>
</tr>
<tr>
<td>Enable/disable using existing source power calibration array when acquiring calibration standard data:</td>
<td>SENSE:CORRection:COLLect:GUIDed:PCAL:APPLY</td>
<td></td>
</tr>
<tr>
<td>Return list of ports being calibrated by an</td>
<td>SENSE:CORRection:COLLect:GUIDed:PORTs?</td>
<td></td>
</tr>
</tbody>
</table>
active calibration session

### Adapter settings for Unknown Thru or Adapter Removal

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets use of a THRU adapter</td>
<td>SENSe:CORRection:COLLect:GUIDed:ADAPter:CREate</td>
<td>None</td>
<td>Sets use of a THRU adapter</td>
</tr>
<tr>
<td>Set adapter delay</td>
<td>SENSe:CORRection:COLLect:GUIDed:ADAPter:DELay</td>
<td>None</td>
<td>Set adapter delay</td>
</tr>
<tr>
<td>Set adapter description</td>
<td>SENSe:CORRection:COLLect:GUIDed:ADAPter:DESCription</td>
<td>None</td>
<td>Set adapter description</td>
</tr>
<tr>
<td>Set port pairs for adapter</td>
<td>SENSe:CORRection:COLLect:GUIDed:ADAPter:PATHs</td>
<td>None</td>
<td>Set port pairs for adapter</td>
</tr>
<tr>
<td>Clear the settings</td>
<td>SENSe:CORRection:COLLect:GUIDed:ADAPter:COUNt:ZERO</td>
<td>None</td>
<td>Clear the settings</td>
</tr>
<tr>
<td>Return number of adapters</td>
<td>SENSe:CORRection:COLLect:GUIDed:ADAPter:COUNt?</td>
<td>None</td>
<td>Return number of adapters</td>
</tr>
</tbody>
</table>

### Optional Guided Cal commands

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto-Orient ECal</td>
<td>SENSe:CORRection:PREFerence:ECAL:ORIentation</td>
<td></td>
<td>cal.OrientECALModule</td>
</tr>
<tr>
<td>Manual orient ECAL</td>
<td>SENSe:CORRection:PREFerence:ECAL:PMAP</td>
<td></td>
<td>cal.ECALPortMapEx</td>
</tr>
<tr>
<td>Calculate Error Terms from a Guided Cal</td>
<td>SENSe:CORRection:COLLect:GUIDed:SAVE</td>
<td></td>
<td>GenerateErrorTerms</td>
</tr>
<tr>
<td>Save Cal to an existing Cal Set GUID</td>
<td>SENSe:CORRection:COLLect:GUIDed:SAVE:CSET</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Load Eterms during a cal</td>
<td>SENSe:CORRection:COLLect:GUIDed:ETERms:LOAD</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Abort Guided cal</td>
<td>SENSe:CORRection:COLLect:GUIDed:ABORt</td>
<td></td>
<td>None required. Destroy the GuidedCal object to terminate a cal.</td>
</tr>
<tr>
<td>Execute the Ecal calibration</td>
<td>SENSE:CORRection:COLLect:GUIDed:ECAL:ACQuire</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>Specifies the Ecal Kit for Ecal Calibration</td>
<td>SENSE:CORRection:COLLect:GUIDed:ECAL:SELect</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Compute Error Terms</td>
<td>SENSE:CORRection:COLLect:GUIDed:ETERms:COMPute</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Guided Power Cal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use standard Source Power commands to make advanced settings.</td>
</tr>
<tr>
<td>Use Power Sensor commands to configure the power sensor.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Perform power cal</th>
<th>SENSE:CORRection:COLLect:GUIDed:PSENsor</th>
<th>PerformPowerCalibration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power sensor connector type</td>
<td>SENSE:CORRection:COLLect:GUIDed:PSENsor:CONNector</td>
<td>PowerSensorConnectorType</td>
</tr>
<tr>
<td>Cal Kit for power cal</td>
<td>SENSE:CORRection:COLLect:GUIDed:PSENsor:CKIT</td>
<td>PowerSensorCalKitType</td>
</tr>
<tr>
<td>Perform match-correction</td>
<td>SENSE:CORRection:METHods:MATCH</td>
<td>MatchCorrectPower</td>
</tr>
<tr>
<td>Sets and returns the selected ports to include in a full NPort correction.</td>
<td>SENSE:CORRection:METHods:PORT:SUBSet:FULL[:VALue]</td>
<td>FullyCorrectedPorts</td>
</tr>
<tr>
<td>Resets the full and response list to their default values.</td>
<td>SENSE:CORRection:METHods:PORT:SUBSet:RESet</td>
<td>ResetPortValues</td>
</tr>
<tr>
<td>Sets and returns the selected ports to be corrected with</td>
<td>SENSE:CORRection:METHods:PORT:SUBSet:RESPonse[:VALue]</td>
<td>ResponseCorrectedPorts</td>
</tr>
<tr>
<td>Enhanced response calibration.</td>
<td>Set and return the ON/OFF subset correction state.</td>
<td>CorrectionSubsettingState</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td><strong>SENSe:CORRection:METHODs:PORT:SUBSet:[STATE]</strong></td>
<td><strong>SENSe:CORRection:COLLect:GUIDed:PSENsor:POWTable</strong></td>
<td><strong>PowerTableFilename</strong></td>
</tr>
</tbody>
</table>

**Load Power Table**

- Used with SMC on mmWave systems.

**Perform Enhanced Response Cal**

|-----------------------|---------------------------------------------|---------------|

**Perform Sliding Load Acquisition**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset iterations</td>
<td>SENS:CORRection:COLLect:GUIDed:ITERations:RESet</td>
<td>ResetStep</td>
</tr>
</tbody>
</table>

**Perform an ECAL**

<table>
<thead>
<tr>
<th>Specify Module and Characterization</th>
<th>SENS:CORRection:COLLect:ACQuire</th>
<th>cal.ECALCharacterizationEx</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do ECAL 1-Port</td>
<td>SENS:CORRection:COLLect:CKIT 99</td>
<td>cal.DoECAL1PortEx</td>
</tr>
<tr>
<td>Do ECAL 2-Port</td>
<td>SENS:CORRection:COLLect:CKIT 99</td>
<td>cal.DoECAL2PortEx</td>
</tr>
<tr>
<td>Task Description</td>
<td>Command/Method</td>
<td>Function/Output</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>Get list of ECAL Modules attached to PNA</td>
<td>SENSe:CORRection:CKIT:ECAL:LIST?</td>
<td>cal.ECALModuleNumberList</td>
</tr>
<tr>
<td>Get list of characterizations in ECal module</td>
<td>SENSe:CORRection:CKIT:ECAL:CLIST?</td>
<td>cal.ECALCharacterizationIndexList</td>
</tr>
<tr>
<td>Perform Module Orientation during calibration</td>
<td>SENSe:CORRection:PREFerence:ECAL:ORIentation</td>
<td>cal.OrientECALModule</td>
</tr>
<tr>
<td>Maps ECAL Module to PNA Ports</td>
<td>SENSe:CORRection:PREFerence:ECAL:PMAP</td>
<td>cal.ECALPortMapEx</td>
</tr>
<tr>
<td>Perform ECal Isolation</td>
<td>SENSe:CORRection:COLLect:ISOLation:ECAL</td>
<td>ECALIsolation</td>
</tr>
<tr>
<td>Increment Avg for ECal Isolation</td>
<td>SENSe:CORRection:COLLect:ISOLation:AVERage:INCRement</td>
<td>IsolationAveragingIncrement</td>
</tr>
<tr>
<td>Return the ID string of ECals</td>
<td>SYSTem:COMMunicate:ECAL:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Return a list of characterizations</td>
<td>SYSTem:COMMunicate:ECAL:CLIST?</td>
<td>None</td>
</tr>
<tr>
<td>Return the number of installed cal kits</td>
<td>SYSTem:COMMunicate:ECAL:COUNt?</td>
<td>None</td>
</tr>
<tr>
<td>Delete user characterizations from VNA disk memory</td>
<td>SYSTem:COMMunicate:ECAL:DMEMory:CLEar</td>
<td>None</td>
</tr>
<tr>
<td>Import file into VNA disk memory</td>
<td>SYSTem:COMMunicate:ECAL:DMEMory:IMPort</td>
<td>None</td>
</tr>
<tr>
<td>Save existing ECal characterization to a file</td>
<td>SYSTem:COMMunicate:ECAL:EXPort</td>
<td>None</td>
</tr>
<tr>
<td>Read identification and characterization information for ECal module</td>
<td>SYSTem:COMMunicate:ECAL:INFormation?</td>
<td>None</td>
</tr>
<tr>
<td>Read identification and characterization information from ECal module or VNA disk memory</td>
<td>SYSTem:COMMunicate:ECAL:KNAMe:INFormation?</td>
<td>None</td>
</tr>
<tr>
<td>Return list of index numbers for ECal modules</td>
<td>SYSTem:COMMunicate:ECAL:LIST?</td>
<td>None</td>
</tr>
<tr>
<td>Return number of unique states for specified path name on selected ECal module</td>
<td>SYSTem:COMMunicate:ECAL:PATH:COUNt?</td>
<td>None</td>
</tr>
</tbody>
</table>

**Perform ECal User Characterizations**

| Perform User ECal Characterization | All SCPI commands | All COM commands |

**Manage PNA Disk Memory Characterizations**

<p>| Delete disk memory characterizations. | SENSE:CORRection:CKIT:ECAL:DMEMory:CLEar | None |
| Saves a disk memory characterization to an archive file. | SENSE:CORRection:CKIT:ECAL:EXPort | None |
| Imports the ECal characterization from the specified archive file. | SENSE:CORRection:CKIT:ECAL:DMEMory:IMPort | None |</p>
<table>
<thead>
<tr>
<th>Reads the user-characterization info from ECal module or PNA disk memory.</th>
<th>SENSE:CORRection:CKIT:ECAL:KNAMe:INFormation?</th>
<th>None</th>
</tr>
</thead>
</table>

### ECal Confidence Check

<table>
<thead>
<tr>
<th>Confidence Check Parameter</th>
<th>SENSE:CORRection:CCheck:PARameter</th>
<th>cal.AcquireCalConfidenceCheckECALEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidence Check Acquire</td>
<td>SENSE:CORRection:CCheck</td>
<td>cal.AcquireCalConfidenceCheckECALEX</td>
</tr>
<tr>
<td>Confidence Check Done</td>
<td>SENSE:CORRection:CCheck:DONE</td>
<td>cal.DoneCalConfidenceCheckECAL</td>
</tr>
</tbody>
</table>

### Set/Read ECal State

<table>
<thead>
<tr>
<th>Sets the state of an ECal module</th>
<th>CONTrol:ECAL:MODule:PATH:STATe</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read ECal state data</td>
<td>SENSE:CORRection:CKIT:ECAL:PATH:DATA?</td>
<td>None</td>
</tr>
</tbody>
</table>

### Calibrate All Channels

<table>
<thead>
<tr>
<th>Select the channels to be calibrated.</th>
<th>SYSTem:CALibrate:ALL:SELect</th>
<th>Channels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set the IFBW</td>
<td>SYSTem:CALibrate:ALL:IFBW</td>
<td>IFBW</td>
</tr>
<tr>
<td>Set the power level</td>
<td>SYSTem:CALibrate:ALL:PORT:SOURce:POWer</td>
<td>PowerLevel</td>
</tr>
<tr>
<td>Set the power offset</td>
<td>SYSTem:CALibrate:ALL:PORT:SOURce:POWer:OFFSet</td>
<td>PowerOffset</td>
</tr>
<tr>
<td>Set the receiver attenu</td>
<td>SYSTem:CALibrate:ALL:PORT:RECeiver:ATTen</td>
<td>ReceiverAttenuator</td>
</tr>
<tr>
<td>Set the source attenu</td>
<td>SYSTem:CALibrate:ALL:PORT:SOURce:POWer:ATTen</td>
<td>SourceAttenuator</td>
</tr>
<tr>
<td>Set the User Calset Prefix</td>
<td>SYSTem:CALibrate:ALL:CSET:PREFIX</td>
<td>UserCalsetPrefix</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Set Path Configuration</td>
<td>SYSTem:CALibrate:ALL:PATH:CONFigure:ELEMENT</td>
<td>PathConfigurationElement</td>
</tr>
<tr>
<td>Read unique property values</td>
<td>SYSTem:CAL:ALL:MClAss:PROPerty:VALue:CATalog?</td>
<td>PropertyValues</td>
</tr>
<tr>
<td>Set property name/value</td>
<td>SYSTem:CAL:ALL:MClAss:PROPerty:VALue</td>
<td>PropertyValue</td>
</tr>
<tr>
<td>Read primary Cal channel</td>
<td>SYSTem:CALibrate:ALL:GUIDed:CHANnel[:VALue]?</td>
<td>None</td>
</tr>
<tr>
<td>Get GuidedCal handle</td>
<td>None</td>
<td>GuidedCalibration</td>
</tr>
<tr>
<td>For each channel, sets the ports to be calibrated.</td>
<td>SYSTem:CALibrate:ALL:CHANnel:PORTs</td>
<td>CalibrationPorts</td>
</tr>
<tr>
<td>Returns a final list of ports to be calibrated.</td>
<td>SYSTem:CALibrate:ALL:GUIDed:PORTs?</td>
<td>SParameterCalPorts</td>
</tr>
<tr>
<td>Returns all cal all guided calibration channels</td>
<td>SYSTem:CALibrate:ALL:GUIDed:CHANnel:LIST?</td>
<td>None</td>
</tr>
<tr>
<td><strong>Returns available ports for independent power calibration.</strong></td>
<td><strong>SYSTEM:CALibrate:ALL:INDependent:SOURce:CALibrate:CATalog?</strong></td>
<td><strong>ValidPorts</strong></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td><strong>Adds a power cal range for a specific port &lt;n&gt;.</strong></td>
<td><strong>SYSTEM:CALibrate:ALL:INDependent:SOURce:CALibrate:RANGe:ADD</strong></td>
<td><strong>AddPowerCalRange</strong></td>
</tr>
<tr>
<td><strong>Resets all ranges for the given source port &lt;n&gt;.</strong></td>
<td><strong>SYSTEM:CALibrate:ALL:INDependent:SOURce:CALibrate:RANGe:CLEAR</strong></td>
<td><strong>Reset</strong></td>
</tr>
<tr>
<td><strong>Queries how many ranges are included in the calibration for source port &lt;n&gt;.</strong></td>
<td><strong>SYSTEM:CALibrate:ALL:INDependent:SOURce:CALibrate:RANGe:COUN?</strong></td>
<td><strong>RangeCount</strong></td>
</tr>
<tr>
<td><strong>Sets and gets the number of points for range &lt;m&gt; for source port&lt;n&gt;.</strong></td>
<td><strong>SYSTEM:CALibrate:ALL:INDependent:SOURce:CALibrate:RANGe:POINt</strong></td>
<td><strong>NumberOfPoints</strong></td>
</tr>
<tr>
<td><strong>Sets and gets the start frequency for range &lt;m&gt; for source port&lt;n&gt;.</strong></td>
<td><strong>SYSTEM:CALibrate:ALL:INDependent:SOURce:CALibrate:RANGe:STARt</strong></td>
<td><strong>StartFrequency</strong></td>
</tr>
<tr>
<td><strong>Sets and gets the stop frequency for range &lt;m&gt; for</strong></td>
<td><strong>SYSTEM:CALibrate:ALL:INDependent:SOURce:CALibrate:RANGe:STOP</strong></td>
<td><strong>StopFrequency</strong></td>
</tr>
<tr>
<td>Source</td>
<td>Port&lt;n&gt;</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
<td></td>
</tr>
</tbody>
</table>

**Recall / Save / Apply a Calibration or Error Term**

<table>
<thead>
<tr>
<th>Operation</th>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recall a Calibration</td>
<td>SENSe:CORRection:CSET</td>
<td>app.Recall</td>
</tr>
<tr>
<td>Apply a Calibration to a measurement</td>
<td>SENSe:CORRection:CSET</td>
<td>app.Recall</td>
</tr>
<tr>
<td>Save a Calibration</td>
<td>SENSe:CORR:CSET:SAVE</td>
<td>app.Save</td>
</tr>
<tr>
<td>Save or Recall an Error Term</td>
<td>CALCulate:DATA Scorr</td>
<td>None</td>
</tr>
<tr>
<td>Read/ Write Cal Set data</td>
<td>SENSe:CORRection:CSET:DATA</td>
<td>None</td>
</tr>
<tr>
<td>Apply an Error Term after Uploading</td>
<td>SENSe:CORRection:COLLect:APPLy</td>
<td>None</td>
</tr>
</tbody>
</table>

**Cal Sets**

<table>
<thead>
<tr>
<th>Operation</th>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quickly test a prototype of automation software</td>
<td>SENSe:CORRection:CSET:CREate:DEFault</td>
<td>None</td>
</tr>
<tr>
<td>Create a Cal Set</td>
<td>SENSe:CORRection:CSET:CREate</td>
<td>calMgr.CreateCalSet</td>
</tr>
<tr>
<td>Delete a Cal Set</td>
<td>SENSe:CORRection:CSET:DELeTe</td>
<td>calMgr.DeleteCalSet</td>
</tr>
<tr>
<td>List Cal Sets</td>
<td>CSET:CATalog?</td>
<td>calMgr.GetCalSetCatalog</td>
</tr>
<tr>
<td>List Cal Sets in VNA</td>
<td>None</td>
<td>EnumerateCalSets</td>
</tr>
<tr>
<td>Get Cal Set Information</td>
<td>None</td>
<td>calMgr.GetCalSetUsageInfo</td>
</tr>
<tr>
<td>List Cal Set Error Terms</td>
<td>SENSe:CORRection:CSET:ETERM:CATalog?</td>
<td>Get ErrorTermList2</td>
</tr>
<tr>
<td>Return if a Cal Set exists</td>
<td>CSET:EXISTS?</td>
<td>Exists</td>
</tr>
<tr>
<td>Select a Cal Set by GUID</td>
<td>SENSe:CORRection:CSET:ACTivate</td>
<td>calMgr.GetCalSetByGUID</td>
</tr>
<tr>
<td>Apply a Cal Set to a channel</td>
<td>SENSe:CORRection:CSET:ACTivate</td>
<td>channel.SelectCalSet</td>
</tr>
<tr>
<td>Copy a Cal Set</td>
<td>SENSe:CORRection:CSET:COPY</td>
<td>CalSet.Copy</td>
</tr>
<tr>
<td>Function</td>
<td>Command/Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>Save a Cal Set</td>
<td>SENSe:CORRection:CSET:SAVE</td>
<td>CalSet.Save</td>
</tr>
<tr>
<td>Save Cal Sets</td>
<td>None</td>
<td>app.SaveCalSets</td>
</tr>
<tr>
<td>Automatically save to User Cal Set</td>
<td>SENSe:CORRection:PREFER:ence:CSET:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Change the Description of a Cal Set</td>
<td>SENSe:CORRection:CSET:DEscription</td>
<td>CalSet.Description</td>
</tr>
<tr>
<td>Change the Name of a Cal Set</td>
<td>SENSe:CORRection:CSET:NAME</td>
<td>calset.Name</td>
</tr>
<tr>
<td>Recall a Cal File</td>
<td>MMEMory:LOAD</td>
<td>app.Recall</td>
</tr>
<tr>
<td>Save 'in-memory' Cal Set to disk.</td>
<td>SENSe:CORRection:CSET:FLATten</td>
<td>None</td>
</tr>
<tr>
<td>Create Cal Set with De-embedded fixture removed.</td>
<td>CSET:FIXTure:DEEMbed</td>
<td>Deembed</td>
</tr>
<tr>
<td>Create Cal Set with Matching Network included.</td>
<td>CSET:FIXTure:EMBed</td>
<td>Embed</td>
</tr>
<tr>
<td>Adds stimulus data to a specific buffer.</td>
<td>None</td>
<td>PutErrorTermStimulus</td>
</tr>
<tr>
<td>Returns the stimulus values over which the specific error term was acquired.</td>
<td>None</td>
<td>GetErrorTermStimulus</td>
</tr>
<tr>
<td>Returns FOM stimulus values from a Calset.</td>
<td>SENSe:CORRection:CSET:STIMulus?</td>
<td>StimulusValues</td>
</tr>
<tr>
<td>Returns the Cal Types from the calset.</td>
<td>None</td>
<td>ContentDescriptor</td>
</tr>
<tr>
<td>Returns the properties of the calset.</td>
<td>None</td>
<td>Properties</td>
</tr>
<tr>
<td>Returns the numbers of the channels using the calset.</td>
<td>None</td>
<td>ChannelClients</td>
</tr>
<tr>
<td>Unselect Cal Set</td>
<td>SENSe:CORRection:CSET:DEACtivate</td>
<td>UnselectCalset</td>
</tr>
</tbody>
</table>

### Cal Set Items

7242
<table>
<thead>
<tr>
<th>Returns names of the items in a cal set</th>
<th>SENSe:CORRection:CSET:ITEM:CAT?</th>
<th>Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remove name-value pair from cal set</td>
<td>None</td>
<td>RemovelItem</td>
</tr>
<tr>
<td>Read the value of the Cal Set item.</td>
<td>SENSe:CORRection:CSET:ITEM[:DATA]?</td>
<td>None</td>
</tr>
<tr>
<td>Enumerate name-value pair items in the cal set.</td>
<td>None</td>
<td>EnumeratelItems</td>
</tr>
</tbody>
</table>

### Apply Cal Types

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalog ALL Cal Types for the PNA</td>
<td>SENSe:CORRection:TYPE:CATalog?</td>
<td>calMgr.GetCalTypes</td>
</tr>
<tr>
<td>Catalog Cal Types in the Cal Set</td>
<td>SENSe:CORRection:CSET:TYPE:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Is a specific Cal Type contained in the Cal Set?</td>
<td>None</td>
<td>calMgr.HasCalType</td>
</tr>
<tr>
<td>Set and return the measurement Cal Type</td>
<td>CALCulate:MEASure:CORRection:TYPE</td>
<td>meas.CalibrationTypeID</td>
</tr>
<tr>
<td>Set port to measure QSOLT reflection standards.</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Correction Settings

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turn Correction ON</td>
<td>OFF for a channel</td>
<td>SENSe:CORRection</td>
</tr>
<tr>
<td>Turn Correction ON</td>
<td>OFF for a measurement</td>
<td>CALCulate:MEASure:CORRection[:STATE]</td>
</tr>
<tr>
<td>Interpolation ON</td>
<td>OFF</td>
<td>SENSe:CORRection:INTERpolate</td>
</tr>
<tr>
<td>Returns the error correction state for the measurement</td>
<td>CALCulate:MEASure:CORRection:INDicatior?</td>
<td>ErrorCorrectionIndicator</td>
</tr>
</tbody>
</table>

### Preferences

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set default Cal Set Save behavior</td>
<td>SENSe:CORRection:PREFerence:CSET:SAVE</td>
<td>RemoteCalStoragePreference</td>
</tr>
<tr>
<td>Sets behavior for simulated cal</td>
<td>SENSe:CORRection:PREFerence:SIMCal</td>
<td>None</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>External or internal trigger during cal</td>
<td>SENSE:CORRection:PREFerence:TRIG:FREE</td>
<td>PreferInternalTriggerOnUnguidedCal</td>
</tr>
<tr>
<td>Set ECal Port Map</td>
<td>SENSE:CORRection:PREFerence:ECAL:PMAP</td>
<td>cal.ECALPortMapEx</td>
</tr>
<tr>
<td>Set default Cal Type</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Port Extensions**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extensions ON</td>
<td>OFF</td>
<td>SENSE:CORRection:EXTension</td>
</tr>
<tr>
<td>Port 1 Extensions Value</td>
<td>SENSE:CORRection:EXTension:PORT</td>
<td>portExt.Port1</td>
</tr>
<tr>
<td>Port 2 Extensions Value</td>
<td>SENSE:CORRection:EXTension:PORT</td>
<td>portExt.Port2</td>
</tr>
<tr>
<td>Set Freq 1</td>
<td>2</td>
<td>SENSE:CORRection:EXTension:PORT:FREQuency</td>
</tr>
<tr>
<td>Set Loss 1</td>
<td>2</td>
<td>SENSE:CORRection:EXTension:PORT:LOSS</td>
</tr>
<tr>
<td>Use 1</td>
<td>2</td>
<td>SENSE:CORRection:EXTension:PORT:INCLude</td>
</tr>
<tr>
<td>Set Loss at DC</td>
<td>SENSE:CORRection:EXTension:PORT:LDC</td>
<td>fix.PortLossDC</td>
</tr>
<tr>
<td>Relative Velocity</td>
<td>SENSE:CORRection:RVELocity:COAX</td>
<td>app.VelocityFactor</td>
</tr>
<tr>
<td>Port Ext in distance</td>
<td>SENSE:CORRection:EXTension:PORT:DISTance</td>
<td>PortDistance</td>
</tr>
<tr>
<td>Set distance units</td>
<td>SENSE:CORRection:EXTension:PORT:UNIT</td>
<td>PortDistanceUnit</td>
</tr>
<tr>
<td>Set Media per port</td>
<td>SENSE:CORRection:EXTension:PORT:MEDium</td>
<td>PortMedium</td>
</tr>
<tr>
<td>Set waveguide cutoff freq per port</td>
<td>SENSE:CORRection:EXTension:PORT:WGcutoff</td>
<td>PortWGcutoffFreq</td>
</tr>
<tr>
<td>Set Velocity Factor per port</td>
<td>SENSE:CORRection:EXTension:PORT:VELFactor</td>
<td>PortVelocityFactor</td>
</tr>
<tr>
<td>Couple to system Velocity Factor</td>
<td>SENSE:CORRection:EXTension:PORT:SYSVelocity</td>
<td>PortCoupleToSystemVelocity</td>
</tr>
<tr>
<td>Couple to system Media type</td>
<td>SENSE:CORRection:EXTension:PORT:SYSMedia</td>
<td>PortCoupleToSystemMedia</td>
</tr>
</tbody>
</table>

**Auto Port Extensions**
<table>
<thead>
<tr>
<th>Measure OPEN or SHORT for Auto Port Ext.</th>
<th>SENSe:CORRection:EXTension:AUTO:MEASure</th>
<th>AutoPortExtMeasure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets the frequencies used for Auto Port Ext. calculation.</td>
<td>SENSe:CORRection:EXTension:AUTO:CONFig</td>
<td>AutoPortExtConfig</td>
</tr>
<tr>
<td>Include loss correction in Auto Port Ext.?</td>
<td>SENSe:CORRection:EXTension:AUTO:LOSS</td>
<td>AutoPortExtLoss</td>
</tr>
<tr>
<td>Include DC Offset in Auto Port Ext.?</td>
<td>SENSe:CORRection:EXTension:AUTO:DCOFfset</td>
<td>AutoPortExtDCOffset</td>
</tr>
<tr>
<td>Enable specified port for Auto Port Ext.</td>
<td>SENSe:CORRection:EXTension:AUTO:PORT&lt;n&gt;</td>
<td>AutoPortExtState</td>
</tr>
<tr>
<td>Clears old port extension delay and loss data.</td>
<td>SENSe:CORRection:EXTension:AUTO:RESet</td>
<td>AutoPortExtReset</td>
</tr>
<tr>
<td>Set user span start frequency for Auto Port Ext.</td>
<td>SENSe:CORRection:EXTension:AUTO:STARt</td>
<td>AutoPortExtSearchStart</td>
</tr>
<tr>
<td>Set user span stop frequency for Auto Port Ext.</td>
<td>SENSe:CORRection:EXTension:AUTO:STOP</td>
<td>AutoPortExtSearchStop</td>
</tr>
</tbody>
</table>

**Fixturing Commands**

**See also** Ground Loop De-embedding/Embedding commands

| Turn fixturing ON and OFF | CALCulate:FSIMulator:STATe | FixturingState |
| Change order of operations | CALCulate:FSIMulator:SENDed:OORDer | None |
| 2and 4-port Extrapolate | CALCulate:FSIMulator:SNP:EXTRapolate | EnableSnPDataExtrapolation |

**2-Port Fixturing**

<p>| Port matching ON and OFF | CALCulate:FSIMulator:SENDed:PMCircuit:STATe | PortMatchingState |
| Sets Port Matching circuit model. | CALCulate:FSIMulator:SENDed:PMCircuit:PORT:TYPE | PortMatchingCktModel |</p>
<table>
<thead>
<tr>
<th>Sets Port Matching 'S2P' file name.</th>
<th>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:USER:FILename</th>
<th>strPortMatch_S2PFile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets Capacitance 'C' value.</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters:C</td>
<td>PortMatching_C</td>
</tr>
<tr>
<td>Sets Inductance 'L' value.</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters:L</td>
<td>PortMatching_L</td>
</tr>
<tr>
<td>De-embed ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:DEEMbed:STATe</td>
<td>Port2PdeembedState</td>
</tr>
<tr>
<td>Sets De-embedding circuit model.</td>
<td>CALCulate:FSIMulator:SENDed:DEEMbed:PORT</td>
<td>Port2PdeembedCktModel</td>
</tr>
<tr>
<td>Port Impedance ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:STATe</td>
<td>PortArbzState</td>
</tr>
<tr>
<td>Port Z Real</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:REAL</td>
<td>PortArbzReal</td>
</tr>
<tr>
<td>Port Z Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:IMAG</td>
<td>PortArbzImag</td>
</tr>
<tr>
<td>Port Z Real and Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:Z0</td>
<td>PortArbzZ0</td>
</tr>
</tbody>
</table>

**4-Port Network Embed/De-embed commands**

<table>
<thead>
<tr>
<th>Specifies the PNA / DUT topology</th>
<th>CALCulate:FSIMulator:EMBed:TYPE</th>
<th>Embed4PortTopology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the 4-port touchstone file</td>
<td>CALCulate:FSIMulator:EMBed:NETWork:FILename</td>
<td>Embed4PortNetworkFilename</td>
</tr>
</tbody>
</table>
| Specify PNA port connections     | CALCulate:FSIMulator:EMBed:TOPology:A:PORTs  
CALCulate:FSIMulator:EMBed:TOPology:B:PORTs  
CALCulate:FSIMulator:EMBed:TOPology:C:PORTs  
CALCulate:FSIMulator:EMBed:TOPology:D:PORTs | Embed4PortList 
SetCustomDUTTopology |
<p>| 4-port remap                     | CALCulate:FSIMulator:EMBed:NETWork&lt;n&gt;:PMAP | NetworkPortMap |</p>
<table>
<thead>
<tr>
<th><strong>Turn ON or OFF</strong></th>
<th><strong>CALCulate:FSIMulator:EMBed:STATe</strong></th>
<th><strong>Embed4PortState</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Maps the physical VNA ports to a device of balanced and single-ended logical ports for multi-port systems with greater than 4 ports</strong></td>
<td><strong>CALCulate:DTOPology</strong></td>
<td><strong>SetCustomDUTToplogy</strong></td>
</tr>
</tbody>
</table>

### Differential Port Arbitrary Impedance

<table>
<thead>
<tr>
<th><strong>Sets the impedance value</strong></th>
<th><strong>CALCulate:FSIMulator:BALun:DZConversion:BPORT:Z0</strong></th>
<th><strong>DiffZConvPortZ0</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sets real part of impedance</strong></td>
<td><strong>CALCulate:FSIMulator:BALun:DZConversion:BPORT:REAL</strong></td>
<td><strong>DiffZConvPortReal</strong></td>
</tr>
<tr>
<td><strong>Sets imaginary part of impedance</strong></td>
<td><strong>CALCulate:FSIMulator:BALun:DZConversion:BPORT:IMAG</strong></td>
<td><strong>DiffZConvPortImag</strong></td>
</tr>
<tr>
<td><strong>Turn ON or OFF</strong></td>
<td><strong>CALCulate:FSIMulator:BALun:DZConversion:STATe</strong></td>
<td><strong>DiffZConvState</strong></td>
</tr>
</tbody>
</table>

### Common Mode Port Arbitrary Impedance

<table>
<thead>
<tr>
<th><strong>Sets the impedance value</strong></th>
<th><strong>CALCulate:FSIMulator:BALun:CZConversion:BPORT:Z0</strong></th>
<th><strong>CmnModeZConvPortZ0</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sets real part of impedance</strong></td>
<td><strong>CALCulate:FSIMulator:BALun:CZConversion:BPORT:REAL</strong></td>
<td><strong>CmnModeZConvPortReal</strong></td>
</tr>
<tr>
<td><strong>Sets imaginary part of impedance</strong></td>
<td><strong>CALCulate:FSIMulator:BALun:CZConversion:BPORT:IMAG</strong></td>
<td><strong>CmnModeZConvPortImag</strong></td>
</tr>
<tr>
<td><strong>Turn ON or OFF</strong></td>
<td><strong>CALCulate:FSIMulator:BALun:CZConversion:STATe</strong></td>
<td><strong>CmnModeZConvState</strong></td>
</tr>
</tbody>
</table>

### Differential Port Matching

<table>
<thead>
<tr>
<th><strong>Sets type of circuit to embed.</strong></th>
<th><strong>CALCulate:FSIMulator:BALun:DMCircuit:BPORT</strong></th>
<th><strong>DiffPortMatchMode</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Specifies the 2-port touchstone file</strong></td>
<td><strong>CALCulate:FSIMulator:BALun:DMCircuit:BPORT:USER:FILename</strong></td>
<td><strong>DiffPortMatchUserFilename</strong></td>
</tr>
<tr>
<td>Sets Capacitance value</td>
<td>CALCulate:FSIMulator:BALun:DMCircuit:BPORt:PARameters:C</td>
<td>DiffPortMatch_C</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Sets Conductance value</td>
<td>CALCulate:FSIMulator:BALun:DMCircuit:BPORt:PARameters:G</td>
<td>DiffPortMatch_G</td>
</tr>
<tr>
<td>Sets Inductance value</td>
<td>CALCulate:FSIMulator:BALun:DMCircuit:BPORt:PARameters:L</td>
<td>DiffPortMatch_L</td>
</tr>
<tr>
<td>Sets Resistance value</td>
<td>CALCulate:FSIMulator:BALun:DMCircuit:BPORt:PARameters:R</td>
<td>DiffPortMatch_R</td>
</tr>
<tr>
<td>Turns ON/OFF</td>
<td>CALCulate:FSIMulator:BALun:DMCircuit:STATe</td>
<td>DiffPortMatchState</td>
</tr>
</tbody>
</table>

**Power Compensation**


**Remote ONLY**

<table>
<thead>
<tr>
<th>Create Cal Set with De-embedded fixture removed.</th>
<th>CSET:FIXTure:DEEMbed</th>
<th>Deembed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Cal Set with Matching network included.</td>
<td>CSET:FIXTure:EMBed</td>
<td>Embed</td>
</tr>
</tbody>
</table>

**Cal Plane Manager**

<table>
<thead>
<tr>
<th>Characterize a fixture</th>
<th>CSET:FIXTure:CHARacterize</th>
<th>CharacterizeFixture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creates a single S2P file from two existing files.</td>
<td>CSET:FIXTure:CASCade</td>
<td>CascadeS2PFiles</td>
</tr>
</tbody>
</table>

**Global Delta Match Cal**

|-------------------------------------------------|----------------------------------------|---------------------------------------|
## Manage and Modify Cal Kits

<table>
<thead>
<tr>
<th>Task</th>
<th>Command</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set a Cal Kit Active</td>
<td>SENSE:CORRection:COLLect:CKIT</td>
<td>app.CalKitType</td>
</tr>
<tr>
<td>Clear all Cal Kits from PNA</td>
<td>SENSE:CORRection:CKIT:CLEAR</td>
<td>None</td>
</tr>
<tr>
<td>Get a Handle to the Active Cal Kit</td>
<td></td>
<td>app.ActiveCalKit</td>
</tr>
<tr>
<td>Save All Cal Kits after Modifying</td>
<td></td>
<td>app.SaveKits</td>
</tr>
<tr>
<td>Load collection of Kits</td>
<td>SENSE:CORRection:CKIT:LOAD</td>
<td>None</td>
</tr>
<tr>
<td>Load (Recall) All Cal Kits</td>
<td></td>
<td>app.RecallKits</td>
</tr>
<tr>
<td>Import a specified kit.</td>
<td>SENSE:CORRection:CKIT:IMPORT</td>
<td>None</td>
</tr>
<tr>
<td>Restore Cal Kit Default</td>
<td>SENSE:CORRection:COLLect:CKIT:RESET</td>
<td>None</td>
</tr>
<tr>
<td>Restore ALL Cal Kits Default</td>
<td>SENSE:CORRection:CKIT:INITIALIZE</td>
<td>None</td>
</tr>
<tr>
<td>Build a Hybrid Cal Kit</td>
<td></td>
<td>app.BuildHybridKit</td>
</tr>
<tr>
<td>Set the Name of a Cal Kit</td>
<td>SENSE:CORRection:COLLect:CKIT:NAME</td>
<td>calKit.Name</td>
</tr>
<tr>
<td>Set a description of a Cal Kit</td>
<td>SENSE:CORRection:COLLect:CKIT:DESCRIPTION</td>
<td>None</td>
</tr>
<tr>
<td>Get the amount of installed kits</td>
<td>SENSE:CORRection:CKIT:COUNT?</td>
<td>None</td>
</tr>
<tr>
<td>Set the Port Label of a Cal Kit</td>
<td></td>
<td>calKit.Portlabel</td>
</tr>
<tr>
<td>Saves a Cal Kit to a file.</td>
<td>SENSE:CORRection:CKIT:EXPORT</td>
<td>None</td>
</tr>
</tbody>
</table>

## Modify TRL Cal Kit

<table>
<thead>
<tr>
<th>Task</th>
<th>Command</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set reference plane</td>
<td>SENSE:CORRection:COLLect:CKIT:TRLoption:RPLane</td>
<td>None</td>
</tr>
<tr>
<td>Action</td>
<td>Command</td>
<td>Function</td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
<td>----------</td>
</tr>
<tr>
<td>Set impedance standard</td>
<td>SENSE:CORRection:COLLect:CKIT:TRLoption:IMPedance</td>
<td>None</td>
</tr>
<tr>
<td>Set LRL auto-characterization</td>
<td>SENSE:CORRection:COLLect:CKIT:TRLoption:LRLChar</td>
<td>None</td>
</tr>
</tbody>
</table>

### Modify Cal Standards

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete a standard</td>
<td>SENSE:CORRection:COLLect:CKIT:STANdard:REMove</td>
<td>None</td>
</tr>
<tr>
<td>Change description of a standard</td>
<td>SENSE:CORRection:COLLect:CKIT:STANdard:SDEscription</td>
<td>None</td>
</tr>
<tr>
<td>Assign a Class to a Standard</td>
<td>SENSE:CORRection:COLLect:CKIT:ORDER1</td>
<td>calKit.StandardForClass</td>
</tr>
<tr>
<td>Set Standard Type</td>
<td>SENSE:CORRection:COLLect:CKIT:STANdard:TYPE</td>
<td>calstd.Type</td>
</tr>
<tr>
<td>Add connector family name</td>
<td>SENSE:CORRection:COLLect:CKIT:CONNector:ADD</td>
<td>None</td>
</tr>
<tr>
<td>Delete connector family name</td>
<td>SENSE:CORRection:COLLect:CKIT:CONNector:DELETE</td>
<td>None</td>
</tr>
<tr>
<td>List connector family names used in a Cal Kit</td>
<td>SENSE:CORRection:COLLect:CKIT:CONNector:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Replace connector family name</td>
<td>SENSE:CORRection:COLLect:CKIT:CONNector:FNAMe</td>
<td>None</td>
</tr>
<tr>
<td>Assign connector family name to a standard</td>
<td>SENSE:CORRection:COLLect:CKIT:CONNector:SNAMe</td>
<td>None</td>
</tr>
<tr>
<td>Set Loss</td>
<td>SENSE:CORRection:COLLect:CKIT:STANdard:LOSS</td>
<td>calstd.loss</td>
</tr>
<tr>
<td>Set Label</td>
<td>SENSE:CORRection:COLLect:CKIT:STANdard:LABel</td>
<td>calstd.Label</td>
</tr>
<tr>
<td>Set Medium (coax</td>
<td>waveguide)</td>
<td>SENSE:CORRection:COLLect:CKIT:STANdard:CHARacter</td>
</tr>
<tr>
<td>Set Capacitance (C0 to C3)</td>
<td>SENSE:CORRection:COLLect:CKIT:STANdard:C0</td>
<td>calstd.C0</td>
</tr>
<tr>
<td>Set Inductance (L0 to L3)</td>
<td>SENSE:CORRection:COLLect:CKIT:STANdard:L0</td>
<td>calstd.L0</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>Set Arbitrary Impedance (TZReal, TZImag)</td>
<td>SENSE:CORRection:COLLect:CKIT:STANdard:TZReal</td>
<td>calstd.TZReal</td>
</tr>
</tbody>
</table>

**Modify TRL Cal Kit**

<table>
<thead>
<tr>
<th>Set reference plane</th>
<th>SENSE:CORRection:COLLect:CKIT:TRLoption:RPLane</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set impedance standard</td>
<td>SENSE:CORRection:COLLect:CKIT:TRLoption:IMPedance</td>
<td>None</td>
</tr>
<tr>
<td>Set LRL auto-characterization</td>
<td>SENSE:CORRection:COLLect:CKIT:TRLoption:LRLChar</td>
<td>None</td>
</tr>
</tbody>
</table>

**Real-time Uncertainty**

**Setup Options**

<table>
<thead>
<tr>
<th>Uncertainty Options</th>
<th>SYSTEM:UNCertainty:ETERm:NOIS:ENAB</th>
<th>PortNoiseEnabled</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SYSTEM:UNCertainty:ETERm:CABLE:REPeat</td>
<td>CableRepeatabilityEnabled</td>
</tr>
<tr>
<td></td>
<td>SYSTEM:UNCertainty:ETERm:SDEFinitions</td>
<td>StandardDefinitionsEnabled</td>
</tr>
<tr>
<td></td>
<td>SYSTEM:UNCertainty:POINts:MAXimum</td>
<td>MaximumUncertaintyPoints</td>
</tr>
</tbody>
</table>

**Noise Characterization**

<table>
<thead>
<tr>
<th>Clear noise data on specified port</th>
<th>SYSTEM:UNCertainty:PORT&lt;p&gt;:NOISe:RESet</th>
<th>ResetNoise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear noise data on all ports</td>
<td>SYSTEM:UNCertainty:PORT:NOISe:RESet</td>
<td>ResetNoiseForAllPorts</td>
</tr>
<tr>
<td>Copy noise from a port to all ports</td>
<td>SYSTem:UNCertainty:PORT:NOISe:ALL:COPY</td>
<td>CopyNoiseToAllPorts</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>----------------------------------------</td>
<td>---------------------</td>
</tr>
</tbody>
</table>

### Cables Characterization

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Assign Cable to all ports</td>
<td>SYSTem:UNCertainty:PORT:CABLE:ALL</td>
<td>SelectCableForAllPorts</td>
</tr>
<tr>
<td>Assign Cable to specified port</td>
<td>SYSTem:UNCertainty:PORT&lt;p&gt;:CABLE</td>
<td>Cable</td>
</tr>
<tr>
<td>Reset repeatability</td>
<td>SYSTem:UNCertainty:CABL:REP:RES</td>
<td>ResetRepeatability</td>
</tr>
<tr>
<td>Start Cable char</td>
<td>SENS:CORR:COLL:GUID:UNCertainty:CHAR:CABLE</td>
<td>InitiateCableCharacterization</td>
</tr>
</tbody>
</table>

### Uncertainty workspace

<table>
<thead>
<tr>
<th>Load workspace</th>
<th>SYSTem:UNCertainty:LOAD</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save workspace</td>
<td>SYSTem:UNCertainty:STORe</td>
<td>Save</td>
</tr>
</tbody>
</table>

### Enabling a Guided Calibration to include Uncertainties

<table>
<thead>
<tr>
<th>Checkbox on Guided Cal Select Ports page</th>
<th>SENSE:CORRection:COLectL:GUIDed:UNCertainty</th>
<th>UncertaintyEnabled</th>
</tr>
</thead>
</table>

### Trace Properties

<table>
<thead>
<tr>
<th>CALCulate:MEASure:UNCertainty:DISPLAY:TYPE</th>
<th>DisplayType</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:MEASure:UNCertainty:DISPLAY:CFACtor</td>
<td>CoverageFactor</td>
</tr>
<tr>
<td>CALCulate:MEASure:UNCertainty:MODE:NOISe</td>
<td>MeasurementNoiseUncertainty</td>
</tr>
</tbody>
</table>
Apply to all traces
None
Add Trace
None
Save uncertainty data
CALCulate:MEASure:UNCertainty:SAVE
None
None
CableRepeatabilityUncertainty
CALCulate:MEASure:UNCertainty:MODE:ETE Rm
ErrorTermUncertainty

Multiple Power Sensors
See commands to configure a Power Meter as Receiver (PMAR)

Enable multiple sensors
SENSe:CORRection:COLLect:GUIDed:PSENsor:MULTiple
UseMultipleSensors

Add sensors
Add

Assign power sensor name
Name

Remove sensors
Remove

Read the number of configured sensors
SENSe:CORRection:COLLect:GUIDed:PSENsor:MULTiple:COUNt?
Count
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Set connector type</td>
<td>SENSE:CORRection:COLLect:GUIDed:PSEN sor:MULTiple:CONNector</td>
<td>PowerSensorConnectorType</td>
</tr>
</tbody>
</table>

### Source Power Calibration

<table>
<thead>
<tr>
<th>Copy Source Power cal to another channel</th>
<th>SYSTem:MACRo:COPY:CHANnel:SOURce</th>
<th>ApplySourcePowerCorrectionTo</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPIB Power Meter Address</td>
<td>SYSTem:COMMunicate:GPIB:PMETer:ADDRess</td>
<td>pwrCal.PowerMeterGPIBAddress</td>
</tr>
<tr>
<td>Set source power cal method</td>
<td>SOURce:POWER:CORRection:COLLect:ACQ</td>
<td>SetCallInfoEx Method</td>
</tr>
<tr>
<td>Turn correction ON</td>
<td>OFF</td>
<td>SOURce:POWER:CORRection</td>
</tr>
<tr>
<td>Applies correction values after completing a source power cal acquisition sweep.</td>
<td>SOURce:POWER:CORRection:COLLect:SAVE</td>
<td>ApplyPowerCorrectionValuesEx</td>
</tr>
<tr>
<td>Optionally do reference receiver cal.</td>
<td>None</td>
<td>PowerAcquisitionDevice</td>
</tr>
<tr>
<td>Returns the currently-selected power sensor channel (A or B) for</td>
<td>None</td>
<td>PowerAcquisitionDevice</td>
</tr>
<tr>
<td>Task Description</td>
<td>Command</td>
<td>Method/Function</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------------------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>Use at a specific frequency.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set power level</td>
<td><code>SOURce:POWER:CORRection:LEV</code></td>
<td>SetCallInfoEx Method</td>
</tr>
<tr>
<td>Set power offset</td>
<td><code>SOURce:POWER:CORRection:OFFSet</code></td>
<td>SourcePowerCalPowerOffset</td>
</tr>
<tr>
<td>Set settling tolerance</td>
<td><code>SOURce:POWER:CORRection:COLLect:AVERage:NTOLerance</code></td>
<td>ReadingsTolerance</td>
</tr>
<tr>
<td>Set max readings for settling</td>
<td><code>SOURce:POWER:CORRection:COLLect:AVERage:COUNt</code></td>
<td>ReadingsPerPoint</td>
</tr>
<tr>
<td>Set accuracy tolerance</td>
<td><code>SOURce:POWER:CORRection:COLLect:ITERation:NTOLerance</code></td>
<td>IterationsTolerance</td>
</tr>
<tr>
<td>Set max readings for accuracy</td>
<td><code>SOURce:POWER:CORRection:COLLect:ITERation:COUNt</code></td>
<td>MaximumIterationsPerPoint</td>
</tr>
<tr>
<td>Turn ON</td>
<td>OFF display of readings</td>
<td><code>SOURce:POWER:CORRection:COLLect:DISPlay</code></td>
</tr>
<tr>
<td>Acquire receiver-only readings</td>
<td><code>SOURce:POWER:CORRection:COLLect:ACQuire:REC</code></td>
<td>AcquirePowerReadingsEx</td>
</tr>
<tr>
<td>Initiates a source power cal acquisition.</td>
<td><code>SOURce:POWER:CORRection:COLLect:ACQuire</code></td>
<td>AcquirePowerReadingsEx</td>
</tr>
<tr>
<td>Aborts a source power cal acquisition sweep that is currently in progress.</td>
<td><code>SOURce:POWER:CORRection:COLLect:ABORt</code></td>
<td>AbortPowerAcquisition</td>
</tr>
<tr>
<td>Launches the Power Meter Settings dialog on the PNA.</td>
<td>None</td>
<td>LaunchPowerMeterSettingsDialog</td>
</tr>
<tr>
<td>Frequency checking (ON</td>
<td>OFF)</td>
<td><code>SOURce:POWER:CORRection:COLLect:FCheck</code></td>
</tr>
<tr>
<td>Operation</td>
<td>Command(s)</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Check test port power</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Calibrate the source at multiple power levels.</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Specifies if the source power cal in the calset linked to a measurement cal should be enabled or disabled with that cal</td>
<td>None</td>
<td>PreferSourcePowerCalFromCalset</td>
</tr>
<tr>
<td>Enable/disable use of error messages during a source calibration if calibration fails to achieve desired power level at the power sensor</td>
<td>SOURce:POWer:CORRection:COLLect:WARN</td>
<td>None</td>
</tr>
</tbody>
</table>

**Power Meter/Sensor settings**

*See commands to configure a Power Meter as Receiver (PMAR)*

*See commands to configure multiple power sensors for guided Power Cal*

<table>
<thead>
<tr>
<th>Setting</th>
<th>Command(s)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the type of power sensor to be used</td>
<td>SYSTem:COMMunicate:PSENsor</td>
<td>Path</td>
</tr>
<tr>
<td>Specifies the location of the power sensor to be used.</td>
<td>SYSTem:COMMunicate:PSENsor</td>
<td>Locator</td>
</tr>
<tr>
<td>Returns the ID string of connected USB power meters / sensors.</td>
<td>SYSTem:COMMunicate:USB:PMETer:CATalog?</td>
<td>USBPowerMeterCatalog</td>
</tr>
<tr>
<td>Pwr meter Max Readings for settling</td>
<td>SOURce:POWer:CORRection:COLLect:AVERage:COUNt</td>
<td>ReadingsPerPoint</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>Pwr meter settling tolerance</td>
<td><code>SOURce:POWer:CORRection:COLLect:AVERage:NTOLerance</code></td>
<td>ReadingsTolerance</td>
</tr>
<tr>
<td>Minimum Frequency</td>
<td><code>SOURce:POWer:CORRection:COLLect:&lt;-&gt;SENsor</code></td>
<td>MinimumFrequency</td>
</tr>
<tr>
<td>Maximum Frequency</td>
<td><code>SOURce:POWer:CORRection:COLLect:&lt;-&gt;SENsor</code></td>
<td>MaximumFrequency</td>
</tr>
<tr>
<td>Power meter channel</td>
<td>None</td>
<td>PowerMeterChannel</td>
</tr>
<tr>
<td>Set sensor cal factor</td>
<td><code>SOURce:POWer:CORRection:COLLect:&lt;-&gt;SENsor:RCFactor</code></td>
<td>ReferenceCalFactor</td>
</tr>
<tr>
<td>Set table type</td>
<td><code>SOURce:POWer:CORRection:COLLect:TABLE &lt;&gt;SENsor</code></td>
<td>None</td>
</tr>
<tr>
<td>Read/Write cal data</td>
<td><code>SOURce:POWer:CORRection:DATA</code></td>
<td>See Data</td>
</tr>
<tr>
<td>Use Loss table?</td>
<td><code>SOURce:POWer:CORRection:COLLect:TABLE:LOSS</code></td>
<td>UsePowerLossSegments</td>
</tr>
<tr>
<td>Cal Factor Table</td>
<td><code>SOURce:POWer:CORRection:COLLect:TABLE:SELect</code></td>
<td>CalFactorSegments Collection</td>
</tr>
<tr>
<td>Read number of segments in table</td>
<td><code>SOURce:POWer:CORRection:COLLect:TABLE:POINts?</code></td>
<td>Count</td>
</tr>
<tr>
<td>Segment number</td>
<td>None</td>
<td>SegmentNumber</td>
</tr>
<tr>
<td>Add segment</td>
<td>None</td>
<td>Add</td>
</tr>
<tr>
<td>Cal factor of the segment</td>
<td><code>SOURce:POWer:CORRection:COLLect:TABLE:DATA</code></td>
<td>CalFactor</td>
</tr>
<tr>
<td>Frequency of the segment</td>
<td><code>SOURce:POWer:CORRection:COLLect:TABLE:FREQuency</code></td>
<td>Frequency</td>
</tr>
<tr>
<td>Power Loss Table</td>
<td><code>SOURce:POWer:CORRection:COLLect:TABLE:SELect</code></td>
<td>PowerLossSegments Collection</td>
</tr>
<tr>
<td>Read number of segments in table</td>
<td><code>SOURce:POWer:CORRection:COLLect:TABLE:POINts?</code></td>
<td>Count</td>
</tr>
<tr>
<td>Segment number</td>
<td>None</td>
<td>SegmentNumber</td>
</tr>
<tr>
<td>Add segment</td>
<td>None</td>
<td>Add</td>
</tr>
<tr>
<td>Frequency</td>
<td><code>SOURce:POWer:CORRection:COLLect:TABLE:FREQuency</code></td>
<td>Frequency</td>
</tr>
<tr>
<td>Loss value</td>
<td><code>SOURce:POWer:CORRection:COLLect:TABLE:DATA</code></td>
<td>Loss</td>
</tr>
<tr>
<td>Receiver Cal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Action</td>
<td>Command</td>
<td>Function</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>Set offset from test port power</td>
<td><code>SENSe:CORRection:RPOWer:OFFSet[:AMPLitude]</code></td>
<td>DoReceiverPowerCal</td>
</tr>
<tr>
<td>Set cal method to receiver cal</td>
<td><code>SENSe:CORRection:COLLect:METHod RPOWer</code></td>
<td>DoReceiverPowerCal</td>
</tr>
<tr>
<td>Take measurement</td>
<td><code>SENSe:CORRection:COLLect[:ACQuire] POWer</code></td>
<td>None</td>
</tr>
<tr>
<td>Turn receiver cal ON</td>
<td>OFF</td>
<td>Error Correction</td>
</tr>
<tr>
<td>Do interpolation</td>
<td>`SENSe:CORRection:INTerpolate[:STATE] ON</td>
<td>OFF`</td>
</tr>
</tbody>
</table>

### CalPod

<table>
<thead>
<tr>
<th>Command used to send other commands as arguments</th>
<th><code>CONTrol:CALPod:COMmand</code></th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start the CalPod software</td>
<td><code>Calpod:LAUNch</code></td>
<td>None</td>
</tr>
<tr>
<td>Assign Calpod serial number to a port.</td>
<td><code>Calpod:ENABle</code></td>
<td>None</td>
</tr>
<tr>
<td>Unassign Calpod serial number from a port.</td>
<td><code>Calpod:Disable</code></td>
<td>None</td>
</tr>
<tr>
<td>Initialize the selected channel</td>
<td><code>Calpod:INITialize:ACTive</code></td>
<td>None</td>
</tr>
<tr>
<td>Initialize ALL channels</td>
<td><code>Calpod:INITialize:ALL</code></td>
<td>None</td>
</tr>
<tr>
<td>Recorrect the selected channel</td>
<td><code>Calpod:Recorrect:ACTive</code></td>
<td>None</td>
</tr>
<tr>
<td>Recorrect ALL channels</td>
<td><code>Calpod:Recorrect:ALL</code></td>
<td>None</td>
</tr>
<tr>
<td>Show refresh dialog</td>
<td><code>Calpod:SHOW</code></td>
<td>None</td>
</tr>
<tr>
<td>Hide refresh dialog</td>
<td><code>Calpod:HIDE</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets impedance state</td>
<td><code>Calpod:STATe</code></td>
<td>None</td>
</tr>
<tr>
<td>Read Calpod temperature</td>
<td><code>Calpod:TEMP?</code></td>
<td>None</td>
</tr>
</tbody>
</table>

### Custom Cal Window

7258
| Turn ON | OFF Custom Cal window. | SENSe:CORRection:COLLect:DISPlay:WINDow | DisplayNAWindowDuringCalAcquisition |
| Show NO Custom Cal windows. | SENSe:CORRection:COLLect:DISPlay:WINDow:AOFF | DisplayOnlyCalWindowDuringCalAcquisition |
| Specify channel to sweep before Cal acquisition. | SENSe:CORRection:COLLect:SWEep:CHANnel | AllowChannelToSweepDuringCalAcquisition |
| Sweep NO channel before Cal acquisition. | SENSe:CORRection:COLLect:SWEep:CHANnel:AOFF | SweepOnlyCalChannelDuringCalAcquisition |
| Preview sweep before remote Cal acquisition. | SENSe:CORRection:COLLect:GUIDed:PACQuire | SetupMeasurementsForStep |

**Retrieve and Put Calibration Data**

| Retrieve Cal Data from the PNA | SENSe:CORRection:CSET:DATA | see Data Topic |
| Put Cal Data in the PNA | SENSe:CORRection:CSET:DATA | see Data Topic |

**Automatic Fixture Removal (AFR)**

<p>| Turns ON of OFF AFR mode conversion. | AFR:ADVanced:MCONversion[:STATe] | None |
| Resets the AFR configuration. | AFR:ADVanced:RESet | None |
| Gets or sets the manual start time in AFR configuration. | AFR:ADVanced:TIME:STARt | None |
| Gets or sets the manual stop time in AFR configuration. | AFR:ADVanced:TIME:STOP | None |
| Gets or sets the manual window type in AFR configuration. | AFR:ADVanced:WINDow:COEFFicient | None |</p>
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turns ON or OFF the manual window coefficient of the AFR.</td>
<td>AFR:ADVanced:WINDow:MANual[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Selects whether the fixture is band limited or not.</td>
<td>AFR:FIXTure:BLIMited[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Selects whether to use DUT correction or not when the characterization fixture is not equal to the DUT measurement fixture.</td>
<td>AFR:FIXTure:CDUT[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Selects Fixture Length A not equal to B correction.</td>
<td>AFR:FIXTure:CLENgth[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Selects Fixture Match A not equal to B correction.</td>
<td>AFR:FIXTure:CMATch[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Describes the fixture inputs (single ended or differential).</td>
<td>AFR:FIXTure:INPuts</td>
<td>None</td>
</tr>
<tr>
<td>Selects the number of fixtures to be characterized.</td>
<td>AFR:FIXTure:MEASurement</td>
<td>None</td>
</tr>
<tr>
<td>Refreshes preview data.</td>
<td>AFR:FIXTure:PREView</td>
<td>None</td>
</tr>
<tr>
<td>Reads the impedance profile of the calculated fixture model.</td>
<td>AFR:FIXTure:PREView:DATA[:IMPedance]?</td>
<td>None</td>
</tr>
<tr>
<td>Reads the impedance profile of the calculated fixture model at a specified position.</td>
<td>AFR:FIXTure:PREView:DATA[:IMPedance]:MARKer:Y?</td>
<td>None</td>
</tr>
<tr>
<td>Chooses the calibration reference Z0 after fixture removal.</td>
<td>AFR:FIXTure:REFZ</td>
<td>None</td>
</tr>
<tr>
<td>Sets &quot;System Z0&quot; to Calibration Reference Z0.</td>
<td>AFR:FIXTure:SET:SYSZ[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Specifies whether thru's are used in case of multi-port fixtures.</td>
<td>AFR:FIXTure:USE:THRUs[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Restores the default AFR settings.</td>
<td>AFR:INITialize</td>
<td>None</td>
</tr>
<tr>
<td>Specifies the file paths of saved fixture data.</td>
<td>AFR:SAVE:FILename</td>
<td>None</td>
</tr>
<tr>
<td>Specifies whether the port impedances are normalized in saving the AFR fixture files.</td>
<td>AFR:SAVE:IMPedance:NORMalize[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Assigns the ports for saved fixture data in several formats.</td>
<td>AFR:SAVE:PORTs</td>
<td>None</td>
</tr>
<tr>
<td>Sets the file type to save fixture data.</td>
<td>AFR:SAVE:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Task Description</td>
<td>Command</td>
<td>Note</td>
</tr>
<tr>
<td>------------------------------------------------------------</td>
<td>--------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Selects all OPEN standards.</td>
<td><code>AFR:STANdard:ALLOpen[:STATE]</code></td>
<td>None</td>
</tr>
<tr>
<td>Selects all SHORT standards.</td>
<td><code>AFR:STANdard:ALLShort[:STATE]</code></td>
<td>None</td>
</tr>
<tr>
<td>Reads the impedance profile of the measured standard.</td>
<td><code>AFR:STANdard:DATA[:IMPedance]?</code></td>
<td>None</td>
</tr>
<tr>
<td>Reads the impedance of the measured standard at a specified position.</td>
<td><code>AFR:STANdard:DATA[:IMPedance]:MARKer:Y?</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the fixture length for the selected fixture (for 1X AFR only).</td>
<td><code>AFR:STANdard:EDIT:FLENgth</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the gate position for the selected fixture.</td>
<td><code>AFR:STANdard:EDIT:GATE</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the impedance for the selected term.</td>
<td><code>AFR:STANdard:EDIT:IMPedance</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the impedance method.</td>
<td><code>AFR:STANdard:EDIT:IMPedance:METHod</code></td>
<td>None</td>
</tr>
<tr>
<td>Loads the calibration standards data from a file.</td>
<td><code>AFR:STANdard:LOAD</code></td>
<td>None</td>
</tr>
<tr>
<td>Measures calibration standard.</td>
<td><code>AFR:STANdard:MEASure</code></td>
<td>None</td>
</tr>
<tr>
<td>Specifies fixture thru settings.</td>
<td><code>AFR:STANdard:THRU</code></td>
<td>None</td>
</tr>
<tr>
<td>Chooses the calibration standards.</td>
<td><code>AFR:STANdard:USE</code></td>
<td>None</td>
</tr>
</tbody>
</table>
## CF_Avg BW Commands

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Averaging</td>
<td>On/Off</td>
<td>SENSE:AVERage[:STATe]</td>
<td>Averaging</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SENSE:AVERage:COUNt</td>
<td>AveragingFactor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(STATus:OPERation:AVERaging)</td>
<td></td>
</tr>
<tr>
<td>Averaging Restart</td>
<td></td>
<td>SENSE:AVERage:CLEAR</td>
<td>AveragingRestart</td>
</tr>
<tr>
<td>Average Type</td>
<td></td>
<td>SENSE:AVERage:MODE</td>
<td>AverageMode</td>
</tr>
<tr>
<td>IF Bandwidth</td>
<td></td>
<td>SENSE:BANDwidth</td>
<td>BWIDth[:RESolution]</td>
</tr>
<tr>
<td>Vector Avg</td>
<td></td>
<td>SENSE:SA:COHerence:VECtor:AVERage:VALUE</td>
<td></td>
</tr>
<tr>
<td>(Spectrum Analyzer</td>
<td></td>
<td>SENSE:SA:COHerence:VECtor:AVERage:STATE</td>
<td></td>
</tr>
<tr>
<td>Measurement Class only)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Noise BW</td>
<td></td>
<td>SENSE:SA:BANDwidth:NOISe</td>
<td>None</td>
</tr>
<tr>
<td>(Modulation Distortion</td>
<td></td>
<td>SENSE:SA:BANDwidth:NOISe:AUTO</td>
<td>None</td>
</tr>
<tr>
<td>Measurement Class only)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Detector Type</td>
<td></td>
<td>SENSE:SA:DETector:FUNCtion</td>
<td>DetectorFunction</td>
</tr>
</tbody>
</table>

7262
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Command</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main Tone IFBW</td>
<td></td>
<td>SENSe:IMD:IFBWidth:MAIN</td>
<td></td>
</tr>
<tr>
<td>IM Tone IFBW</td>
<td></td>
<td>SENSe:IMD:IFBWidth:IMTone</td>
<td></td>
</tr>
<tr>
<td>LF Auto BW</td>
<td>ON/OFF</td>
<td>SENSe:BANDwidth</td>
<td>BWIDth:TRACk</td>
</tr>
<tr>
<td>Noise Avg</td>
<td></td>
<td>SENSe:NOISe:AVERage</td>
<td></td>
</tr>
<tr>
<td>Noise BW</td>
<td>720,000 kHz</td>
<td>SENSe:NOISe:BWIDth</td>
<td></td>
</tr>
<tr>
<td>RBW Ratio</td>
<td></td>
<td>SENSe:PN:BWIDth[:RESolution]:RATio</td>
<td></td>
</tr>
</tbody>
</table>

None
(Phase Noise Measurement  
Class only)

### Smoothing Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smoothing</td>
<td>ON/OFF</td>
<td>CALCulate:MEASure:SMOothing[:STATe]</td>
<td>Smoothing</td>
</tr>
<tr>
<td>Smooth Percent</td>
<td></td>
<td>CALCulate:MEASure:SMOothing:APERture</td>
<td>SmoothingAperture</td>
</tr>
<tr>
<td>Smooth Points</td>
<td></td>
<td>CALCulate:MEASure:SMOothing:POINts</td>
<td>None</td>
</tr>
</tbody>
</table>

### Delay Aperture Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aperture Percent</td>
<td></td>
<td>CALCulate:MEASure:GDELay:PERCent</td>
<td>Percent</td>
</tr>
<tr>
<td>Aperture Points</td>
<td></td>
<td>CALCulate:MEASure:GDELay:POINts</td>
<td>Points</td>
</tr>
<tr>
<td>Aperture Freq</td>
<td></td>
<td>CALCulate:MEASure:GDELay:FREQuency</td>
<td>Frequency</td>
</tr>
</tbody>
</table>
CF_Cal Commands

The Cal softkeys vary depending on which measurement class is currently active. Click on a measurement class link below to view the corresponding softkeys.

- Standard
- Active Hot Parameters
- Gain Compression and Gain Compression Converters
- Differential I/Q
- IM Spectrum and IM Spectrum Converters
- Swept IMD and Swept IMD Converters
- Modulation Distortion and Modulation Distortion Converters
- Noise Figure Cold Source and Noise Figure Converters
- Phase Noise
- Scalar Mixer/Converter + Phase and Vector Mixer/Converter
- Spectrum Analyzer
### CF_Cal Commands - DIQ

**Cal Commands - Differential I/Q Measurement Class**

Click [here](#) to view links to Cal commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
<td><strong>Sub-item</strong></td>
<td><strong>SCPI</strong></td>
<td><strong>COM</strong></td>
</tr>
<tr>
<td><strong>Smart Cal...</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Correction</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Src Power Correct</strong></td>
<td><strong>ON/OFF</strong></td>
<td><strong>SOURce:POWER:CORRection[:STATe]</strong></td>
<td><strong>SourcePowerCorrection</strong></td>
</tr>
<tr>
<td><strong>Interpolation</strong></td>
<td><strong>ON/OFF</strong></td>
<td><strong>SENSe:CORRection:INTerpolate[:STATe]</strong></td>
<td><strong>InterpolateCorrection</strong></td>
</tr>
<tr>
<td><strong>Properties..</strong></td>
<td></td>
<td><strong>None</strong></td>
<td><strong>Properties</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cal Sets &amp; Cal Kits Tab Commands</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
<td><strong>Sub-item</strong></td>
<td><strong>SCPI</strong></td>
<td><strong>COM</strong></td>
</tr>
<tr>
<td>Cal Set...</td>
<td>ON/OFF</td>
<td>DISP:TOOL:CSET[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Cal Set Viewer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cal Kit...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cal Pod...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Fixtures Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply Fixtures</td>
<td>ON/OFF</td>
<td>CALC:FSIMulator:STATE</td>
<td>FixturingState</td>
</tr>
<tr>
<td>Power Comp...</td>
<td>Port N</td>
<td>CALC:FSIM:SEND:POW:PORT:COMP</td>
<td>EnablePowerCompensation</td>
</tr>
<tr>
<td>Fixture Setup</td>
<td>Change order of operations</td>
<td>CALC:FSIMulator:SENDed:OORDER</td>
<td>None</td>
</tr>
<tr>
<td>2 and 4-port Extrapolate</td>
<td></td>
<td>CALC:FSIMulator:SNP:EXTRapolate</td>
<td>EnableSnPDataExtrapolation</td>
</tr>
<tr>
<td>2-Port Fixturing</td>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------</td>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>Port matching ON and OFF</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:STATe</td>
<td>PortMatchingState</td>
</tr>
<tr>
<td>Reverse ports</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:DEEM:PORT&lt;n&gt;:SNP:REVere</td>
<td>Reverse2PortAdapter</td>
</tr>
<tr>
<td>Sets Port Matching circuit model</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:TYPE</td>
<td>PortMatchingCktModel</td>
</tr>
<tr>
<td>Sets Port Matching 'S2P' file name</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:USER:FILen ame</td>
<td>strPortMatch_S2PFile</td>
</tr>
<tr>
<td>Sets Capacitance 'C' value</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters: C</td>
<td>PortMatching_C</td>
</tr>
<tr>
<td>Sets Inductance 'L' value</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters: L</td>
<td>PortMatching_L</td>
</tr>
<tr>
<td>Sets Resistance 'R' value</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters: R</td>
<td>PortMatching_R</td>
</tr>
<tr>
<td>De-embed ON and OFF</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:DEEMbed:STATe</td>
<td>Port2PdeembedState</td>
</tr>
<tr>
<td>Sets De-embedding circuit model</td>
<td>CALCulate:FSIMulator:SENDed:DEEMbed:PORT</td>
<td>Port2PdeembedCktModel</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------</td>
<td>--------------------------</td>
<td></td>
</tr>
<tr>
<td>Port Impedance ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:STATE</td>
<td>PortArbzState</td>
<td></td>
</tr>
<tr>
<td>Port Z Real</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:REAL</td>
<td>PortArbzReal</td>
<td></td>
</tr>
<tr>
<td>Port Z Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:IMAG</td>
<td>PortArbzImag</td>
<td></td>
</tr>
<tr>
<td>Port Z Real and Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:Z0</td>
<td>PortArbzZ0</td>
<td></td>
</tr>
<tr>
<td>Remote ONLY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create Cal Set with De-embeded fixture removed</td>
<td>CSET:FIXTure:DEEMbed</td>
<td>Deembed</td>
<td></td>
</tr>
<tr>
<td>Create Cal Set with</td>
<td>CSET:FIXTure:EMBed</td>
<td>Embed</td>
<td></td>
</tr>
</tbody>
</table>
| Matchin
g network
included | | |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal Plane Manager...</td>
<td>Characterize a fixture</td>
<td>CSET:FIXTure:CHARacterize</td>
</tr>
<tr>
<td></td>
<td>Creates a single S2P file from two existing files</td>
<td>CSET:FIXTure:CASCade</td>
</tr>
<tr>
<td>Auto Fixture Removal...</td>
<td></td>
<td>CSET:FIXTure:DEEMbed</td>
</tr>
</tbody>
</table>
### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smart Cal...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other Cals</td>
<td>Cal All...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source Power</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cal...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Src Power</td>
<td>ON/OFF</td>
<td>SOURce:POWER:CORRection[:STATe]</td>
<td>SourcePowerCorrection</td>
</tr>
<tr>
<td>Correct</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interpolation</td>
<td>ON/OFF</td>
<td>SENSE:CORRection:INTERpolate[:STATe]</td>
<td>InterpolateCorrection</td>
</tr>
<tr>
<td>Correction</td>
<td></td>
<td>CALCulate:MEASure:CORRection:TYPE</td>
<td>meas.CalibrationTypeID</td>
</tr>
<tr>
<td>Methods...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Port Extension Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>Port N</td>
<td>SENSE:CORRection:EXTension:PORT</td>
<td>portExt.Port1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>portExt.Port2</td>
</tr>
<tr>
<td>Port Extension</td>
<td>ON/OFF</td>
<td>SENSE:CORRection:EXTension</td>
<td>portExtension.State</td>
</tr>
<tr>
<td>Time</td>
<td></td>
<td>SENSE:CORRection:EXTension:PORT:TIME</td>
<td>PortDelay</td>
</tr>
<tr>
<td>Distance</td>
<td></td>
<td>SENSE:CORRection:EXTension:PORT:DISTance</td>
<td>PortDistance</td>
</tr>
<tr>
<td>Velocity Factor</td>
<td></td>
<td>SENSE:CORRection:RVELocity:COAX</td>
<td>app.VelocityFactor</td>
</tr>
<tr>
<td>DC Loss</td>
<td></td>
<td>SENSE:CORRection:EXTension:PORT:LDC</td>
<td>fix.PortLossDC</td>
</tr>
<tr>
<td>Auto Port Extension...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Cal Sets & Cal Kits Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal Set...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cal Set Viewer</td>
<td>ON/OFF</td>
<td>DISPlay:TOOL:CSET[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Cal Kit...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cal Pod...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Fixtures Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply Fixtures</td>
<td>ON/OFF</td>
<td>CALCulate:FSIMulator:STATe</td>
<td>FixturingState</td>
</tr>
<tr>
<td>Power Comp...</td>
<td>Port N</td>
<td>CALC:FSIM:SEND:POW:PORT:COMP</td>
<td>EnablePowerCompensation</td>
</tr>
</tbody>
</table>

7273
<table>
<thead>
<tr>
<th>Fixture Setup</th>
<th>Change order of operations</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:OORDer</td>
<td>None</td>
</tr>
<tr>
<td>2 and 4-port Extrapolate</td>
<td></td>
<td>CALCulate:FSIMulator:SNP:EXTRapolate</td>
<td>EnableSnPDataExtrapolation</td>
</tr>
<tr>
<td>2-Port Fixturing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port matching ON and OFF</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:STATe</td>
<td>PortMatchingState</td>
</tr>
<tr>
<td>Sets Port Matching circuit model</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:TYPE</td>
<td>PortMatchingCktModel</td>
</tr>
<tr>
<td>Sets Port Matching 'S2P' file name</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:USER:FILename</td>
<td>strPortMatch_S2PFile</td>
</tr>
<tr>
<td>Sets Capacitance 'C' value</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters:C</td>
<td>PortMatching_C</td>
</tr>
<tr>
<td>Sets Inductance 'L' value</td>
<td></td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters:L</td>
<td>PortMatching_L</td>
</tr>
<tr>
<td>Sets Resistance 'R' value</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters:R</td>
<td>PortMatching_R</td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------------------------------------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>De-embed ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:DEEMbed:STATe</td>
<td>Port2PdeembedState</td>
<td></td>
</tr>
<tr>
<td>Sets De-embedding circuit model</td>
<td>CALCulate:FSIMulator:SENDed:DEEMbed:PORT</td>
<td>Port2PdeembedCktModel</td>
<td></td>
</tr>
<tr>
<td>Port Impedance ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:STATe</td>
<td>PortArbzState</td>
<td></td>
</tr>
<tr>
<td>Port Z Real</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:REAL</td>
<td>PortArbzReal</td>
<td></td>
</tr>
<tr>
<td>Port Z Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:IMAG</td>
<td>PortArbzImag</td>
<td></td>
</tr>
<tr>
<td>Port Z Real and Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:Z0</td>
<td>PortArbzZ0</td>
<td></td>
</tr>
<tr>
<td>Remote ONLY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create Cal Set with De-embeded fixture</td>
<td>CSET:FIXTure:DEEMbed</td>
<td>Deembed</td>
<td></td>
</tr>
<tr>
<td>Removed</td>
<td>Create Cal Set with Matching network included</td>
<td>CSET:FIXTure:EMBed</td>
<td>Embed</td>
</tr>
<tr>
<td>---------</td>
<td>---------------------------------------------</td>
<td>--------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Cal Plane Manager...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### CF_Cal Commands - HotS22

#### Cal Commands - Active Hot Parameters

Click [here](#) to view links to Cal commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cal All...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Src Power Correct</td>
<td>ON/OFF</td>
<td>SOURce:POWer:CORRection[:STATe]</td>
<td>SourcePowerCorrection</td>
<td></td>
</tr>
<tr>
<td>Interpolation</td>
<td>ON/OFF</td>
<td>SENSE:CORRection:INTerpolate[:STATe]</td>
<td>InterpolateCorrection</td>
<td></td>
</tr>
<tr>
<td>Correction Properties...</td>
<td>None</td>
<td></td>
<td>Properties</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cal Sets &amp; Cal Kits Tab Commands</th>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cal Set...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cal Set Viewer</td>
<td>ON/OFF</td>
<td>DISPlay:TOOL:CSET[:STATe]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Cal Kit...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECal...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cal Pod...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Fixtures Tab Commands</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
<td></td>
</tr>
<tr>
<td><strong>Apply Fixtures</strong></td>
<td>ON/OFF</td>
<td>CALCulate:FSIMulator:STATe</td>
<td><strong>FixturingState</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Fixture Setup</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Cal Plane Manager...</strong></td>
<td>Characterize a fixture</td>
<td>CSET:FIXTure:CHARacterize</td>
<td>CharacterizeFixture</td>
<td></td>
</tr>
<tr>
<td>Creates a single S2P file from two existing files</td>
<td>CSET:FIXTure:CASCade</td>
<td>CascadeS2PFiles</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## CF_Cal Commands - IMD_IMDX

### Cal Commands - Swept IMD and Swept IMD Converters Measurement Classes

Click [here](#) to view links to Cal commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td>Smart Cal...</td>
</tr>
<tr>
<td>Other Cals</td>
</tr>
<tr>
<td><strong>Correction</strong></td>
</tr>
<tr>
<td>Src Power Correct</td>
</tr>
<tr>
<td>Interpolation</td>
</tr>
<tr>
<td>Properties..</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cal Sets &amp; Cal Kits Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>-------------</td>
</tr>
<tr>
<td>Cal Set...</td>
</tr>
<tr>
<td>Cal Set Viewer</td>
</tr>
<tr>
<td>Cal Kit...</td>
</tr>
<tr>
<td>ECal</td>
</tr>
<tr>
<td>Cal Pod...</td>
</tr>
</tbody>
</table>

**Fixtures Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply Fixtures</td>
<td>ON/OFF</td>
<td><code>CALCulate:FSIMulator:STATe</code></td>
<td>FixturingState</td>
</tr>
<tr>
<td>Power Comp...</td>
<td>Port N</td>
<td><code>CALC:FSIM:SEND:POW:PORT:COMP</code></td>
<td>EnablePowerCompensation</td>
</tr>
<tr>
<td>Fixture Setup</td>
<td>Change order of operations</td>
<td><code>CALCulate:FSIMulator:SENDed:OORDer</code></td>
<td>None</td>
</tr>
<tr>
<td>2 and 4-port Extrapolate</td>
<td>CALCulate:FSIMulator:SNP:EXTRapolate</td>
<td>EnableSnPDataExtrapolation</td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------</td>
<td>-----------------------------</td>
<td></td>
</tr>
<tr>
<td>2-Port Fixturing</td>
<td>2-Port Fixturing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port matching ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:STATe</td>
<td>PortMatchingState</td>
<td></td>
</tr>
<tr>
<td>Sets Port Matching circuit model</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:TYPE</td>
<td>PortMatchingCktModel</td>
<td></td>
</tr>
<tr>
<td>Sets Port Matching 'S2P' file name</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:USER:FILename</td>
<td>strPortMatch_S2PFile</td>
<td></td>
</tr>
<tr>
<td>Sets Capacitance 'C' value</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters:C</td>
<td>PortMatching_C</td>
<td></td>
</tr>
<tr>
<td>Sets Inductance 'L' value</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters:L</td>
<td>PortMatching_L</td>
<td></td>
</tr>
<tr>
<td>Sets Resistance 'R' value</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters:R</td>
<td>PortMatching_R</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Result</td>
<td></td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>-----------------</td>
<td></td>
</tr>
<tr>
<td>De-embed ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:DEEmbed:STATe</td>
<td>Port2PdeembedState</td>
<td></td>
</tr>
<tr>
<td>Sets De-embedding circuit model</td>
<td>CALCulate:FSIMulator:SENDed:DEEmbed:PORT</td>
<td>Port2PdeembedCktModel</td>
<td></td>
</tr>
<tr>
<td>Port Impedance ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:STATe</td>
<td>PortArbzState</td>
<td></td>
</tr>
<tr>
<td>Port Z Real</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:REAL</td>
<td>PortArbzReal</td>
<td></td>
</tr>
<tr>
<td>Port Z Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:IMAG</td>
<td>PortArbzImag</td>
<td></td>
</tr>
<tr>
<td>Port Z Real and Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:Z0</td>
<td>PortArbzZ0</td>
<td></td>
</tr>
<tr>
<td>Remote ONLY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create Cal Set with De-embeded fixture removed</td>
<td>CSET:FIXTure:DEEmbed</td>
<td>Deembed</td>
<td></td>
</tr>
<tr>
<td>Create Cal Set with Matching network included</td>
<td>CSET:FIXTure:EMBed</td>
<td>Embed</td>
<td></td>
</tr>
<tr>
<td>Cal Plane Manager...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
| | | |

| | | |
### CF_Cal Commands - IMS_IMSX

#### Cal Commands - IM Spectrum and IM Spectrum Converters Measurement Classes

Click [here](#) to view links to Cal commands for all Measurement Classes.

#### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal All...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Power Correct</th>
<th>ON/OFF</th>
<th>SOURce:POWer:CORRection[:STATe]</th>
<th>SourcePowerCorrection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interpolation</td>
<td>ON/OFF</td>
<td>SENSE:CORRection:INTERpolate[:STATe]</td>
<td>InterpolateCorrection</td>
</tr>
<tr>
<td>Properties..</td>
<td>None</td>
<td>None</td>
<td>Properties</td>
</tr>
</tbody>
</table>

#### Cal Sets & Cal Kits Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

7284
<table>
<thead>
<tr>
<th>Cal Set...</th>
<th>Cal Kit...</th>
<th>ECal</th>
<th>Cal Pod...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal Set Viewer</td>
<td>ON/OFF</td>
<td>DISPlay:TOOL:CSET[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Cal Kit...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cal Pod...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Fixtures Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply Fixtures</td>
<td>ON/OFF</td>
<td>CALCulate:FSIMulator:STATe</td>
<td>FixturingState</td>
</tr>
<tr>
<td>Power Comp...</td>
<td>Port N</td>
<td>CALC:FSIM:SEND:POW:PORT:COMP</td>
<td>EnablePowerCompensation</td>
</tr>
<tr>
<td>Fixture Setup</td>
<td>Change order of operations</td>
<td>CALCulate:FSIMulator:SENDed:OORDER</td>
<td>None</td>
</tr>
<tr>
<td>2 and 4-port Extrapolate</td>
<td></td>
<td>CALCulate:FSIMulator:SNP:EXTRapolate</td>
<td>EnableSnPDataExtrapolation</td>
</tr>
<tr>
<td>2-Port Fixturing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------</td>
<td>-----------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Port matching ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:STATe</td>
<td>PortMatchingState</td>
<td></td>
</tr>
<tr>
<td>Sets Port Matching circuit model</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:TYPE</td>
<td>PortMatchingCktModel</td>
<td></td>
</tr>
<tr>
<td>Sets Port Matching 'S2P' file name</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:USER:FILENAME</td>
<td>strPortMatch_S2PFile</td>
<td></td>
</tr>
<tr>
<td>Sets Capacitance 'C' value</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARAMETERS:C</td>
<td>PortMatching_C</td>
<td></td>
</tr>
<tr>
<td>Sets Inductance 'L' value</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARAMETERS:L</td>
<td>PortMatching_L</td>
<td></td>
</tr>
<tr>
<td>Sets Resistance 'R' value</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARAMETERS:R</td>
<td>PortMatching_R</td>
<td></td>
</tr>
<tr>
<td>De-embed ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:DEEMbed:STATe</td>
<td>Port2PdeembedState</td>
<td></td>
</tr>
<tr>
<td>Sets De-embedding circuit model</td>
<td>CALCulate:FSIMulator:SENDed:DEEMbed:PORT</td>
<td>Port2PdeembedCktModel</td>
<td></td>
</tr>
<tr>
<td>Port Impedance ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:STATE</td>
<td>PortArbzState</td>
<td></td>
</tr>
<tr>
<td>Port Z Real</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:REAL</td>
<td>PortArbzReal</td>
<td></td>
</tr>
<tr>
<td>Port Z Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:IMAG</td>
<td>PortArbzImag</td>
<td></td>
</tr>
<tr>
<td>Port Z Real and Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:Z0</td>
<td>PortArbzZ0</td>
<td></td>
</tr>
<tr>
<td>Remote ONLY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create Cal Set with Deembeded fixture removed</td>
<td>CSET:FIXTure:DEEMbed</td>
<td>Deembed</td>
<td></td>
</tr>
<tr>
<td>Create Cal Set with</td>
<td>CSET:FIXTure:EMBed</td>
<td>Embed</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Matchin g network included</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------</td>
<td>---------------------</td>
<td></td>
</tr>
<tr>
<td>Cal Plane Manager...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CF_Cal Commands - MOD

Cal Commands - Modulation Distortion and Modulation Distortion Converters Measurement Class

Only the Main Cal commands corresponding to the Modulation Distortion and Modulation Distortion Converters measurement class are documented here. All other commands are identical to the Cal commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Cal commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>Cal All...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Other Cals</th>
<th>Src Mod Cal...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal Type - Power</td>
<td></td>
</tr>
<tr>
<td>Power</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:ENABLE</td>
</tr>
<tr>
<td>Cal Port</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:RECeiver</td>
</tr>
<tr>
<td>Cal Span</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:SPAN</td>
</tr>
<tr>
<td>Max Iterations</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:ITERations</td>
</tr>
<tr>
<td>Desired Tolerance</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:TOLerance</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>Cal Type - Equalization</td>
<td></td>
</tr>
<tr>
<td>Cal Port</td>
<td>SOURce:MODulation:CORRection:COLLection:EOualizatio:RECeiver SOURce:MODulation:CORRection:COLLection:FLATn ess:RECeiver</td>
</tr>
<tr>
<td>Cal Type - LO Feedthru</td>
<td></td>
</tr>
<tr>
<td>LO Feedthru</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FTHRu:ENABLe</td>
</tr>
<tr>
<td>Cal Port: Cal Port</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FRu:RECEiver</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>Cal Span: Cal Span</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FRu:SPAN</td>
</tr>
<tr>
<td>Max Iterations: Max Iterations</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FRu:ITERations</td>
</tr>
<tr>
<td>Desired Tolerance: Desired Tolerance</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FRu:TOLERance</td>
</tr>
<tr>
<td>Cal Type - Distortion</td>
<td></td>
</tr>
<tr>
<td>Distortion: Distortion</td>
<td>SOURce:MODulation:CORRection:COLLection:DISTortion:ENABLE</td>
</tr>
<tr>
<td>Cal Port: Cal Port</td>
<td>SOURce:MODulation:CORRection:COLLection:DISTortion:RECEiver</td>
</tr>
<tr>
<td>Cal Span: Cal Span</td>
<td>SOURce:MODulation:CORRection:COLLection:DISTortion:SPAN</td>
</tr>
<tr>
<td>Max Iterations: Max Iterations</td>
<td>SOURce:MODulation:CORRection:COLLection:DISTortion:ITERations</td>
</tr>
<tr>
<td>Desired Tolerance: Desired Tolerance</td>
<td>SOURce:MODulation:CORRection:COLLection:DISTortion:TOLERance</td>
</tr>
<tr>
<td>Cal Type - NPR Notch</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Command Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NPR Notch</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:ENABle</td>
</tr>
<tr>
<td>Cal Port</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:RECeiver</td>
</tr>
<tr>
<td>Cal Span</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:SPAN</td>
</tr>
<tr>
<td>Max Iterations</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:ITERations</td>
</tr>
<tr>
<td>Desired Tolerance</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:TOLerance</td>
</tr>
<tr>
<td>Cal Type - ACP</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command Details</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Guard Band</td>
<td>SOURc:MODulation:CORRection:COLLection:ACP:LOWer:GBAND</td>
</tr>
<tr>
<td></td>
<td>SOURc:MODulation:CORRection:COLLection:ACP:UPPer:GBAND</td>
</tr>
<tr>
<td>Max Iterations</td>
<td>SOURc:MODulation:CORRection:COLLection:ACP:ITERations</td>
</tr>
<tr>
<td></td>
<td>SOURc:MODulation:CORRection:COLLection:ACP:LOWer:ITERations</td>
</tr>
<tr>
<td></td>
<td>SOURc:MODulation:CORRection:COLLection:ACP:UPPer:ITERations</td>
</tr>
<tr>
<td>Desired Tolerance</td>
<td>SOURc:MODulation:CORRection:COLLection:ACP:TOLerance</td>
</tr>
<tr>
<td></td>
<td>SOURc:MODulation:CORRection:COLLection:ACP:LOWer:TOLERance</td>
</tr>
<tr>
<td></td>
<td>SOURc:MODulation:CORRection:COLLection:ACP:UPPer:TOLERance</td>
</tr>
<tr>
<td>Save Cal in File</td>
<td>SOURc:MODulation:SAVE</td>
</tr>
<tr>
<td>File Properties...</td>
<td>SOURc:MODulation:FILE:CORRection:CATalog?</td>
</tr>
<tr>
<td></td>
<td>SOURc:MODulation:FILE:CORRection:DELete?</td>
</tr>
<tr>
<td></td>
<td>SOURc:MODulation:FILE:CORRection:FREQuency?</td>
</tr>
<tr>
<td></td>
<td>SOURc:MODulation:FILE:CORRection:POWer?</td>
</tr>
<tr>
<td></td>
<td>SOURc:MODulation:CORRection:COLLection:APPend</td>
</tr>
<tr>
<td>Mod Cal Details...</td>
<td></td>
</tr>
<tr>
<td>Enable Faster Cal with</td>
<td>SOURc:MODulation:CORRection:COLLection:FAST:ENABLE</td>
</tr>
<tr>
<td>Reduced Accuracy</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>---</td>
</tr>
<tr>
<td>Use Previous Cal as Initial Value</td>
<td>None</td>
</tr>
<tr>
<td>RF Power for calibration - Fixed</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:TYPE FIXed</td>
</tr>
<tr>
<td>RF Power for calibration - Swept</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:TYPE SWEpt</td>
</tr>
<tr>
<td>RF Carrier for calibration - Fixed</td>
<td>SOURce:MODulation:CORRection:COLLection:FREQuency:TYPE FIXed</td>
</tr>
<tr>
<td>RF Carrier for</td>
<td>SOURce:MODulation:CORRection:COLLection:FREQuency:TYPE SWEpt</td>
</tr>
<tr>
<td>Calibration</td>
<td>SOURce:MODulation:CORRection:COLLection:FREQuency:STARt</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------</td>
</tr>
<tr>
<td>- Swept</td>
<td>SOURce:MODulation:CORRection:COLLection:FREQuency:STOP</td>
</tr>
<tr>
<td></td>
<td>SOURce:MODulation:CORRection:COLLection:FREQuency:POINts</td>
</tr>
</tbody>
</table>

**Receiver IF Cal...**

<table>
<thead>
<tr>
<th>Calibrate IF...</th>
<th>SENSE&lt;cnun&gt;-.DISTortion:CORRection:COLLect:IF:ACQuire</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[SYNChronous</td>
</tr>
</tbody>
</table>

**Correction**

<table>
<thead>
<tr>
<th>Src Mod Correct</th>
<th>SOURce:MODulation:CORRection:[STATE]</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON/OFF</td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Interpolation</th>
<th>SENSE:CORRection:INTerpolate:[STATE]</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON/OFF</td>
<td>InterpolateCorrection</td>
</tr>
</tbody>
</table>

**Correction Methods...**

<table>
<thead>
<tr>
<th>Correction Properties. ..</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Properties</td>
</tr>
</tbody>
</table>
# CF_Cal Commands - NF_NFX

**Cal Commands - Noise Figure Cold Source and Noise Figure Converters Measurement Classes**

Click [here](#) to view links to Cal commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td>Smart Cal...</td>
</tr>
<tr>
<td><strong>Other Cals</strong></td>
</tr>
<tr>
<td>Source Power Cal...</td>
</tr>
<tr>
<td><strong>Correction</strong></td>
</tr>
<tr>
<td>Src Power Correct</td>
</tr>
<tr>
<td>Interpolation</td>
</tr>
<tr>
<td>Correction Methods...</td>
</tr>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>--------------</td>
</tr>
<tr>
<td>Cal Set...</td>
</tr>
<tr>
<td>Cal Set Viewer ON/OFF</td>
</tr>
<tr>
<td>Cal Kit...</td>
</tr>
<tr>
<td>ECal</td>
</tr>
<tr>
<td>Cal Pod...</td>
</tr>
</tbody>
</table>

**Fixtures Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply Fixtures ON/OFF</td>
<td></td>
<td>CALCulate:FSIMulator:STATe</td>
<td>FixturingState</td>
</tr>
<tr>
<td>Power Comp... Port N</td>
<td></td>
<td>CALC:FSIM:SEND:POW:PORT:COMP</td>
<td>EnablePowerCompensatio</td>
</tr>
</tbody>
</table>

Properties

- None
<table>
<thead>
<tr>
<th>Fixture Setup</th>
<th>Change order of operations</th>
<th>CALCulate:FSIMulator:SENDed:OORDer</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 and 4-port Extrapolate</td>
<td>CALCulate:FSIMulator:SNP:EXTRapolate</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>2-Port Fixturing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port matching ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:STATe</td>
<td>PortMatchingState</td>
<td></td>
</tr>
<tr>
<td>Sets Port Matching circuit model</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:TYPE</td>
<td>PortMatchingCktModel</td>
<td></td>
</tr>
<tr>
<td>Sets Port Matching 'S2P' file name</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:USER:FILename</td>
<td>strPortMatch_S2PFile</td>
<td></td>
</tr>
<tr>
<td>Sets Capacitance 'C' value</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters: C</td>
<td>PortMatching_C</td>
<td></td>
</tr>
<tr>
<td>Sets Inductance 'L' value</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARameters: L</td>
<td>PortMatching_L</td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PortMatching_R</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>De-embed ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:DEEMbed:STATe</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port2PdeembedState</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets De-embedding circuit model</td>
<td>CALCulate:FSIMulator:SENDed:DEEMbed:PORT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port2PdeembedCktModel</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>strPort2Pdeembed_S2PFile</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port Impedance ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:STATe</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PortArbzState</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port Z Real</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:REAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PortArbzReal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port Z Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:IMAG</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PortArbzImag</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port Z Real and Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:Z0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PortArbzZ0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Remote ONLY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create Cal Set with De-embeded fixture</td>
<td>CSET:FIXTure:DEEMbed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Deembed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CSET:FIXTure:EMBed</td>
<td>Embed</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>create cal set with matching network included</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cal plane manager...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

removed
**CF_Cal Commands - PN**

**Cal Commands - Phase Noise Measurement Class**

Only the Main Cal commands corresponding to the Phase Noise measurement class are documented here. All other commands are identical to the Cal commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click [here](#) to view links to Cal commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal All...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Src Power Correct</td>
<td>ON/OFF</td>
<td></td>
<td>SOURce:POWer:CORRection[:STATe]</td>
<td>SourcePowerCorrection</td>
</tr>
<tr>
<td>Interpolation</td>
<td>ON/OFF</td>
<td></td>
<td>SENSe:CORRection:INTerpolate[:STATe]</td>
<td>InterpolateCorrection</td>
</tr>
<tr>
<td>Correction Methods...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction Properties...</td>
<td></td>
<td>None</td>
<td></td>
<td>Properties</td>
</tr>
</tbody>
</table>
CF_Cal Commands - SA

Cal Commands - Spectrum Analyzer Measurement Class

Only the Main and Fixtures Cal commands corresponding to the Spectrum Analyzer measurement class are documented here. All other commands are identical to the Cal commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Cal commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main</th>
<th>Port Extension</th>
<th>Cal Sets &amp; Cal Kits</th>
<th>Fixtures</th>
</tr>
</thead>
</table>

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal All...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Src Power Correct</td>
<td>ON/OFF</td>
<td>SOURce:POWER:CORRection[:STATe]</td>
<td>SourcePowerCorrection</td>
</tr>
<tr>
<td>Interpolation</td>
<td>ON/OFF</td>
<td>SENSE:CORRection:INTERpolate[:STATe]</td>
<td>InterpolateCorrection</td>
</tr>
<tr>
<td>Correction Methods...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction Properties..</td>
<td>None</td>
<td></td>
<td>Properties</td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------</td>
<td>-------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Apply Fixtures</td>
<td>ON/OFF</td>
<td>CALCulate:FSIMulator:STATe</td>
<td>FixturingState</td>
</tr>
<tr>
<td>Power Comp...</td>
<td>Port N</td>
<td>CALC:FSIM:SEND:POW:PORT:COMP</td>
<td>EnablePowerCompensation</td>
</tr>
<tr>
<td>Fixture Setup</td>
<td>Change order of operations</td>
<td>CALCulate:FSIMulator:SENDed:OORDER</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>2 and 4-port Extrapolate</td>
<td>CALCulate:FSIMulator:SNP:EXTRapolate</td>
<td>EnableSnPDataExtrapolation</td>
</tr>
</tbody>
</table>

### 2-Port Fixuring

<table>
<thead>
<tr>
<th>Port matching ON and OFF</th>
<th>CALCulate:FSIMulator:SENDed:PMCircuit:STATe</th>
<th>PortMatchingState</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets Port Matching circuit model</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:TYPE</td>
<td>PortMatchingCktModel</td>
</tr>
<tr>
<td>Port Description</td>
<td>Command</td>
<td>Variable Name</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>Sets Port Matching 'S2P' file name</td>
<td><code>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:USER:FILENAME</code></td>
<td><code>strPortMatch_S2PFile</code></td>
</tr>
<tr>
<td>Sets Capacitance 'C' value</td>
<td><code>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARAMeters:C</code></td>
<td><code>PortMatching_C</code></td>
</tr>
<tr>
<td>Sets Inductance 'L' value</td>
<td><code>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARAMeters:L</code></td>
<td><code>PortMatching_L</code></td>
</tr>
<tr>
<td>De-embed ON and OFF</td>
<td><code>CALCulate:FSIMulator:SENDed:DEEMbed:STATE</code></td>
<td><code>Port2PdeembedState</code></td>
</tr>
<tr>
<td>Sets De-embedding circuit model</td>
<td><code>CALCulate:FSIMulator:SENDed:DEEMbed:PORT</code></td>
<td><code>Port2PdeembedCktModel</code></td>
</tr>
<tr>
<td>Port Impedance ON and OFF</td>
<td><code>CALCulate:FSIMulator:SENDed:ZCONversion:STATE</code></td>
<td><code>PortArbzState</code></td>
</tr>
<tr>
<td>Port Z Real</td>
<td><code>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:REAL</code></td>
<td><code>PortArbzReal</code></td>
</tr>
<tr>
<td>Port Z Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:IMAG</td>
<td>PortArbzImag</td>
</tr>
<tr>
<td>Port Z Real and Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:Z0</td>
<td>PortArbzZ0</td>
</tr>
<tr>
<td>Remote ONLY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create Cal Set with De-embedde d fixture remove d</td>
<td>CSET:FIXTure:DEEMbed</td>
<td>Deembed</td>
</tr>
<tr>
<td>Create Cal Set with Matching network included</td>
<td>CSET:FIXTure:EMBed</td>
<td>Embed</td>
</tr>
<tr>
<td>Cal Plane Manager...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Auto Fixture Removal...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**CF_Cal Commands - SMC_VMC**

*Cal Commands - Scalar Mixer/Converter + Phase and Vector Mixer/Converter Measurement Classes*

Click [here](#) to view links to Cal commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td><strong>Smart Cal...</strong></td>
</tr>
<tr>
<td><strong>Other Cals</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Correction</strong></td>
</tr>
<tr>
<td><strong>Src Power Correct</strong></td>
</tr>
<tr>
<td><strong>Interpolation</strong></td>
</tr>
<tr>
<td>Correction Methods...</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>Properties...</td>
</tr>
</tbody>
</table>

### Cal Sets & Cal Kits Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal Set...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cal Set Viewer</td>
<td>ON/OFF</td>
<td>DISPlay:TOOL:CSET[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Cal Kit...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cal Pod...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Fixtures Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply Fixtures</td>
<td>ON/OFF</td>
<td>CALCulate:FSIMulator:STATe</td>
<td>FixturingState</td>
</tr>
<tr>
<td>Power Comp...</td>
<td>Port N</td>
<td>CALC:FSIM:SEND:POW:PORT:COMP</td>
<td>EnablePowerCompensation</td>
</tr>
<tr>
<td>Fixture Setup</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>Cal Plane Manager...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# CF_Cal Commands - Standard

## Cal Commands - Standard Measurement Class

Click [here](#) to view links to Cal commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td><strong>Basic Cal...</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Other Cals</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Use Smart Cal Order</td>
</tr>
<tr>
<td>---------------------</td>
</tr>
<tr>
<td>Independent Calibration Channels</td>
</tr>
<tr>
<td>Include Power Calibration</td>
</tr>
<tr>
<td>Set the IFBW</td>
</tr>
<tr>
<td>Set the power level</td>
</tr>
<tr>
<td>Set the power offset</td>
</tr>
<tr>
<td>Set the receiver atten</td>
</tr>
<tr>
<td>Set the User Calset Prefix</td>
</tr>
<tr>
<td>Set Path Configuration</td>
</tr>
<tr>
<td>------------------------</td>
</tr>
<tr>
<td>Read unique property values</td>
</tr>
<tr>
<td>Set property name/value</td>
</tr>
<tr>
<td>Read primary Cal channel</td>
</tr>
<tr>
<td>Returns all guided calibration channels</td>
</tr>
<tr>
<td>Get Guided Cal handle</td>
</tr>
<tr>
<td>For each channel, sets the ports to be calibrated</td>
</tr>
<tr>
<td>Returns a final list of ports to be calibrated</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td><strong>Independent Power Calibration</strong></td>
</tr>
<tr>
<td>Add a power cal range</td>
</tr>
<tr>
<td>Query how many ranges are included</td>
</tr>
<tr>
<td>Set number of points</td>
</tr>
<tr>
<td>Set start frequency</td>
</tr>
<tr>
<td>Action</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Set stop frequency</td>
</tr>
<tr>
<td>Smart Cal...</td>
</tr>
<tr>
<td>Initiate a Guided Cal</td>
</tr>
<tr>
<td>Select a Connector Type</td>
</tr>
<tr>
<td>Select a Cal Kit</td>
</tr>
<tr>
<td>Set cal method for each port pair</td>
</tr>
<tr>
<td>Set Thru Method for each port pair</td>
</tr>
<tr>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Set Thru Port Pairs</td>
</tr>
<tr>
<td>Return Number of Steps in a Cal</td>
</tr>
<tr>
<td>Return a Description of a Cal Step</td>
</tr>
<tr>
<td>Save Cal</td>
</tr>
<tr>
<td><strong>E Cal...</strong></td>
</tr>
<tr>
<td>Specify Module and Characterization</td>
</tr>
<tr>
<td>Task</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Do ECAL 1-Port</td>
</tr>
<tr>
<td>Do ECAL 2-Port</td>
</tr>
<tr>
<td>Get list of ECal Modules attached to PNA</td>
</tr>
<tr>
<td>Perform Module Orientation during calibration</td>
</tr>
<tr>
<td>Maps ECAL Module to PNA Ports</td>
</tr>
<tr>
<td>Action</td>
</tr>
<tr>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>Perform ECal Isolation</td>
</tr>
<tr>
<td>Increment Avg for ECal Isolation</td>
</tr>
<tr>
<td><strong>Response Cal...</strong></td>
</tr>
<tr>
<td>Launch Cal Wizard</td>
</tr>
<tr>
<td>Set Cal Type</td>
</tr>
<tr>
<td>Select a Cal Kit</td>
</tr>
<tr>
<td>Get a Handle to the Active Cal Kit</td>
</tr>
<tr>
<td>Simultaneous 2-Port Calibration</td>
</tr>
<tr>
<td><strong>Acquisition Direction</strong></td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
<tr>
<td><strong>Measure a Standard</strong></td>
</tr>
<tr>
<td><strong>Calculate Errors</strong></td>
</tr>
<tr>
<td><strong>Do Isolation</strong></td>
</tr>
<tr>
<td><strong>Perform and apply Response (Normalization)</strong> cal</td>
</tr>
</tbody>
</table>

Source Power Cal...

<table>
<thead>
<tr>
<th><strong>Copy Source Power cal to another channel</strong></th>
<th>SYSTem:MACRo:COPY:CHANnel:SOURce</th>
<th>ApplySourcePowerCorrectionTo</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GPIB Power Meter Address</strong></td>
<td>SYSTem:COMMunicate:PSENsor</td>
<td>pwrCal.PowerMeterGPIBAaddress</td>
</tr>
<tr>
<td>Set source power cal method</td>
<td>SOURce:POWer:CORRection:COLLect:ACQuire</td>
<td>SetCalInfoEx Method</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Turn correction ON</td>
<td>OFF</td>
<td>SOURce:POWer:CORRection</td>
</tr>
<tr>
<td>Applies correction values after completing a source power cal acquisition sweep Optionally do reference receiver cal</td>
<td>SOURce:POWer:CORRection:COLLect:SAVE</td>
<td>ApplyPowerCorrectionValuesEx</td>
</tr>
<tr>
<td>Returns the currently-selected power sensor channel (A or B) for use at a specific frequency</td>
<td>None</td>
<td>PowerAcquisitionDevice</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Method</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>Set power level</td>
<td>SOURc:e:POWer:CORRection:LEVel</td>
<td>SetCalInfoEx Method</td>
</tr>
<tr>
<td>Set power offset</td>
<td>SOURc:e:POWer:CORRection:OFFSet</td>
<td>SourcePowerCalPowerOffs</td>
</tr>
<tr>
<td>Set settling tolerance</td>
<td>SOURc:e:POWer:CORRection:COLLect:AVERage:NTOLera</td>
<td>ReadingsTolerance</td>
</tr>
<tr>
<td>Set max readings for settling</td>
<td>SOURc:e:POWer:CORRection:COLLect:AVERage:COUNt</td>
<td>ReadingsPerPoint</td>
</tr>
<tr>
<td>Set accuracy tolerance</td>
<td>SOURc:e:POWer:CORRection:COLLect:ITERation:NTOLera</td>
<td>IterationsTolerance</td>
</tr>
<tr>
<td>Set max readings for accuracy</td>
<td>SOURc:e:POWer:CORRection:COLLect:ITERation:COUNt</td>
<td>MaximumIterationsPerPoint</td>
</tr>
<tr>
<td>Turn ON</td>
<td>OFF display of readings</td>
<td>SOURc:e:POWer:CORRection:COLLect:DISPlay</td>
</tr>
<tr>
<td>Action Description</td>
<td>Command</td>
<td>Function</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>Acquire receiver-only readings</td>
<td>SOURce:POWer:CORRection:COLLect:ACQuire</td>
<td>AcquirePowerReadingsEx</td>
</tr>
<tr>
<td>Initiates a source power cal acquisition</td>
<td>SOURce:POWer:CORRection:COLLect:ACQuire</td>
<td>AcquirePowerReadingsEx</td>
</tr>
<tr>
<td>Aborts a source power cal acquisition sweep that is currently in progress</td>
<td>SOURce:POWer:CORRection:COLLect:ABORt</td>
<td>AbortPowerAcquisition</td>
</tr>
<tr>
<td>Launches the Power Meter Settings dialog on the PNA</td>
<td>None</td>
<td>LaunchPowerMeterSettingsDialog</td>
</tr>
<tr>
<td>Frequency checking (ON</td>
<td>OFF)</td>
<td>SOURce:POWer:CORRection:COLLect:FCheck</td>
</tr>
<tr>
<td><strong>Check test port power</strong></td>
<td>None</td>
<td><strong>Calibrate the source at multiple power levels</strong></td>
</tr>
<tr>
<td>---------------------------</td>
<td>------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td><strong>Correction</strong></td>
<td></td>
<td><strong>Channel Correction On</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Channel Correction Off</strong></td>
</tr>
<tr>
<td><strong>Cal Set...</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Create a Cal Set</strong></td>
<td><strong>SENSe:CORRection:CSET:CREate</strong></td>
<td><strong>calMgr.CreateCalSet</strong></td>
</tr>
<tr>
<td><strong>Delete a Cal Set</strong></td>
<td><strong>CSET:DEL</strong></td>
<td><strong>calMgr.DeleteCalSet</strong></td>
</tr>
<tr>
<td><strong>List Cal Sets</strong></td>
<td><strong>CSET:CATalog?</strong></td>
<td><strong>calMgr.GetCalSetCatalog</strong></td>
</tr>
<tr>
<td><strong>List Cal Sets in PNA</strong></td>
<td>None</td>
<td><strong>EnumerateCalSets</strong></td>
</tr>
<tr>
<td>Action</td>
<td>Command</td>
<td>Method</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Get Cal Set Information</td>
<td>None</td>
<td>calMgr.GetCalSetUsageInfo</td>
</tr>
<tr>
<td>List Cal Set Error Terms</td>
<td>SENSE:CORRection:CSET:ETERm:CATalog?</td>
<td>Get ErrorTermList2</td>
</tr>
<tr>
<td>Return if a Cal Set exists</td>
<td>CSET:EXISTS?</td>
<td>Exists</td>
</tr>
<tr>
<td>Select a Cal Set by GUID</td>
<td>SENSE:CORRection:CSET:ACTivate</td>
<td>calMgr.GetCalSetByGUID</td>
</tr>
<tr>
<td>Apply a Cal Set to a channel</td>
<td>SENSE:CORRection:CSET:ACTivate</td>
<td>channel.SelectCalSet</td>
</tr>
<tr>
<td>Copy a Cal Set</td>
<td>SENSE:CORRection:CSET:COPY</td>
<td>CalSet.Copy</td>
</tr>
<tr>
<td>Save a Cal Set</td>
<td>SENSE:CORRection:CSET:SAVE</td>
<td>CalSet.Save</td>
</tr>
<tr>
<td>Save Cal Sets</td>
<td>None</td>
<td>app.SaveCalSets</td>
</tr>
<tr>
<td>Automatically save to User Cal Set</td>
<td>SENSE:CORRection:PREFERENCE:CSET:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Action</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>Change the Description of a Cal Set</td>
<td>SENSE:CORRection:CSET:DESCription</td>
<td>CalSet.Description</td>
</tr>
<tr>
<td>Change the Name of a Cal Set</td>
<td>SENSE:CORRection:CSET:NAME</td>
<td>calset.Name</td>
</tr>
<tr>
<td>Recall a Cal File</td>
<td>MMEMory:LOAD</td>
<td>app.Recall</td>
</tr>
<tr>
<td>Save 'in-memory' Cal Set to disk.</td>
<td>SENSE:CORRection:CSET:FLATten</td>
<td>None</td>
</tr>
<tr>
<td>Create Cal Set with De-embedded fixture removed</td>
<td>CSET:FIXTure:DEEMbed</td>
<td>Deembed</td>
</tr>
<tr>
<td>Create Cal Set with Matching Network included</td>
<td>CSET:FIXTure:EMBed</td>
<td>Embed</td>
</tr>
<tr>
<td>Adds stimulus data to a specific buffer</td>
<td>None</td>
<td>PutErrorTermStimulus</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td>Command</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td>---------</td>
</tr>
<tr>
<td>Returns the stimulus values over which the specific error term was acquired</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Returns FOM stimulus values from a Calset</td>
<td>SENSE:CORRection:CSET:STIMulus?</td>
<td></td>
</tr>
<tr>
<td>Returns the Cal Types from the calset</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Returns the properties of the calset</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Returns the numbers of the channels using the calset</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Unselect Cal Set</td>
<td>SENSE:CORRection:CSET:DEACtivate</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Setting</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>Factory Cal ON/OFF</td>
<td></td>
<td>SYSTEM:FCORrection:CHANnel:COUPler[:STATe]</td>
</tr>
<tr>
<td>Src Power Correct ON/OFF</td>
<td></td>
<td>SOURce:POWer:CORRection[:STATe]</td>
</tr>
<tr>
<td>Interpolation ON/OFF</td>
<td></td>
<td>SENSE:CORRection:INTerpolate[:STATe]</td>
</tr>
<tr>
<td>Correction Methods ON/OFF</td>
<td>Power Correction Type</td>
<td>SENSE:CORRection:WAVE[:METHod]</td>
</tr>
<tr>
<td>Enable Port Subset Correction</td>
<td></td>
<td>SENSE:CORRection:METHods:PORT:SUBSet[:STATe]</td>
</tr>
<tr>
<td>Clear All</td>
<td></td>
<td>SENSE:CORRection:METHods:PORT:SUBSet:RESet</td>
</tr>
<tr>
<td>Correction Properties...</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>
# Port Extension Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>Port N</td>
<td><code>SENSe:CORRection:EXTension:PORT</code></td>
<td><code>portExt.Port1</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>portExt.Port2</code></td>
</tr>
<tr>
<td>Port Extension</td>
<td>ON/OFF</td>
<td><code>SENSe:CORRection:EXTension</code></td>
<td><code>portExtension.State</code></td>
</tr>
<tr>
<td>Time</td>
<td></td>
<td><code>SENSe:CORRection:EXTension:PORT:TIME</code></td>
<td><code>PortDelay</code></td>
</tr>
<tr>
<td>Distance</td>
<td></td>
<td><code>SENSe:CORRection:EXTension:PORT:DISTance</code></td>
<td><code>PortDistance</code></td>
</tr>
<tr>
<td>Velocity Factor</td>
<td></td>
<td><code>SENSe:CORRection:RVELocity:COAX</code></td>
<td><code>app.VelocityFactor</code></td>
</tr>
<tr>
<td>DC Loss</td>
<td></td>
<td><code>SENSe:CORRection:EXTension:PORT:LDC</code></td>
<td><code>fix.PortLossDC</code></td>
</tr>
<tr>
<td>Extensions ON/OFF</td>
<td></td>
<td><code>SENSe:CORRection:EXTension</code></td>
<td><code>portExtension.State</code></td>
</tr>
<tr>
<td>Port 1 Extensions</td>
<td>SENSE:CORRection:EXTension:PORT</td>
<td>portExt.Port1</td>
<td></td>
</tr>
<tr>
<td>Port 2 Extensions</td>
<td>SENSE:CORRection:EXTension:PORT</td>
<td>portExt.Port2</td>
<td></td>
</tr>
<tr>
<td>Set Freq 1</td>
<td>2</td>
<td>SENSE:CORRection:EXTension:PORT:FREQuency</td>
<td>Fix.PortFreq1</td>
</tr>
<tr>
<td>Set Loss 1</td>
<td>2</td>
<td>SENSE:CORRection:EXTension:PORT:LOSS</td>
<td>fix.PortLoss1</td>
</tr>
<tr>
<td>Use 1</td>
<td>2</td>
<td>SENSE:CORRection:EXTension:PORT:INCLude</td>
<td>fix.PortExtUse1</td>
</tr>
<tr>
<td>Set Loss at DC</td>
<td>SENSE:CORRection:EXTension:PORT:LDC</td>
<td>fix.PortLossDC</td>
<td></td>
</tr>
<tr>
<td>Relative Velocity</td>
<td>SENSE:CORRection:RVELocity:COAX</td>
<td>app.VelocityFactor</td>
<td></td>
</tr>
<tr>
<td>Port Ext in distance</td>
<td>SENSE:CORRection:EXTension:PORT:DISTance</td>
<td>PortDistance</td>
<td></td>
</tr>
<tr>
<td>Set distance units</td>
<td>SENSE:CORRection:EXTension:PORT:UNIT</td>
<td>PortDistanceUnit</td>
<td></td>
</tr>
<tr>
<td>Action Description</td>
<td>Command Path</td>
<td>Result Path</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Set Media per port</td>
<td>SENSE:CORRection:EXTension:PORT:MEDium</td>
<td>PortMedium</td>
<td></td>
</tr>
<tr>
<td>Set waveguide cutoff freq per port</td>
<td>SENSE:CORRection:EXTension:PORT:WGCuttof</td>
<td>PortWGCuttofFreq</td>
<td></td>
</tr>
<tr>
<td>Set Velocity Factor per port</td>
<td>SENSE:CORRection:EXTension:PORT:VELFactor</td>
<td>PortVelocityFactor</td>
<td></td>
</tr>
<tr>
<td>Couple to system Velocity Factor</td>
<td>SENSE:CORRection:EXTension:PORT:SYSVelocity</td>
<td>PortCoupleToSystemVelocity</td>
<td></td>
</tr>
<tr>
<td>Couple to system Media type</td>
<td>SENSE:CORRection:EXTension:PORT:SYSMedia</td>
<td>PortCoupleToSystemMedia</td>
<td></td>
</tr>
<tr>
<td>Auto Port Extension ...</td>
<td>Measure OPEN or SHORT for Auto Port Ext</td>
<td>AutoPortExtMeasure</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sets the frequencies used for Auto Port Ext. calculation</td>
<td>AutoPortExtConfig</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Function</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>-------------------------</td>
<td></td>
</tr>
<tr>
<td>Include loss correction in Auto Port Ext</td>
<td><code>SENSe:CORRection:EXTension:AUTO:LOSS</code></td>
<td><code>AutoPortExtLoss</code></td>
<td></td>
</tr>
<tr>
<td>Include DC Offset in Auto Port Ext</td>
<td><code>SENSe:CORRection:EXTension:AUTO:DCOffset</code></td>
<td><code>AutoPortExtDCOffset</code></td>
<td></td>
</tr>
<tr>
<td>Enable specified port for Auto Port Ext</td>
<td><code>SENSe:CORRection:EXTension:AUTO:PORT&lt;n&gt;</code></td>
<td><code>AutoPortExtState</code></td>
<td></td>
</tr>
<tr>
<td>Clears old port extension delay and loss data</td>
<td><code>SENSe:CORRection:EXTension:AUTO:RESet</code></td>
<td><code>AutoPortExtReset</code></td>
<td></td>
</tr>
<tr>
<td>Set user span start frequency for Auto Port Ext</td>
<td><code>SENSe:CORRection:EXTension:AUTO:START</code></td>
<td><code>AutoPortExtSearchStart</code></td>
<td></td>
</tr>
<tr>
<td>Set user span stop frequency for Auto Port Ext</td>
<td><code>SENSe:CORRection:EXTension:AUTO:STOP</code></td>
<td><code>AutoPortExtSearchStop</code></td>
<td></td>
</tr>
</tbody>
</table>

**Cal Sets & Cal Kits Tab Commands**
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal Set…</td>
<td>Cal Set Viewer</td>
<td>ON/OFF</td>
<td>DISPlay:TOOL:CSET[:STATe]</td>
</tr>
<tr>
<td>Cal Kit…</td>
<td>Set a Cal Kit Active</td>
<td>SENSE:CORRection:COLLect:CKIT</td>
<td>app.CalKitType</td>
</tr>
<tr>
<td>Cal Kit…</td>
<td>Clear all Cal Kits from PNA</td>
<td>SENSE:CORRection:CKIT:CLEar</td>
<td>None</td>
</tr>
<tr>
<td>Cal Kit…</td>
<td>Get a Handle to the Active Cal Kit</td>
<td>None</td>
<td>app.ActiveCalKit</td>
</tr>
<tr>
<td>Cal Kit…</td>
<td>Save All Cal Kits after Modifying</td>
<td>None</td>
<td>app.SaveKits</td>
</tr>
<tr>
<td>Cal Kit…</td>
<td>Load collection of Kits</td>
<td>SENSE:CORRection:CKIT:LOAD</td>
<td>None</td>
</tr>
<tr>
<td>Cal Kit…</td>
<td>Load (Recall) All Cal Kits</td>
<td>None</td>
<td>app.RecallKits</td>
</tr>
<tr>
<td>Action</td>
<td>Command</td>
<td>Parameter</td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------------------</td>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>Import a specified kit</td>
<td><code>SENSe:CORRection:CKIT:IMPort</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Restore Cal Kit Default</td>
<td><code>SENSe:CORRection:CKIT:INITialize</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Restore ALL Cal Kits Default</td>
<td><code>SENSe:CORRection:CKIT:INITialize</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Build a Hybrid Cal Kit</td>
<td>None</td>
<td><code>app.BuildHybridKit</code></td>
<td></td>
</tr>
<tr>
<td>Set the Name of a Cal Kit</td>
<td><code>SENSe:CORRection:COLLect:CKIT:NAME</code></td>
<td><code>calKit.Name</code></td>
<td></td>
</tr>
<tr>
<td>Set a description of a Cal Kit</td>
<td><code>SENSe:CORRection:COLLect:CKIT:DESCRIPTION</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Get the amount of installed kits</td>
<td><code>SENSe:CORRection:CKIT:COUNt?</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Set the Port Label of a Cal Kit</td>
<td>None</td>
<td><code>calKit.Portlabel</code></td>
<td></td>
</tr>
<tr>
<td>Saves a Cal Kit to a file</td>
<td><code>SENSe:CORRection:CKIT:EXPort</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Command used to send other commands as arguments</td>
<td>CONTrol:CALPod:COMMannd</td>
<td>None</td>
</tr>
<tr>
<td>---------</td>
<td>-------------------------------------------------</td>
<td>--------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Start the CalPod software</td>
<td>Calpod:LAUNch</td>
<td>Calpod:ENABLe</td>
<td>None</td>
</tr>
<tr>
<td>Assign Calpod serial number to a port.</td>
<td>Calpod: ENABLE</td>
<td>Calpod: Disable</td>
<td>None</td>
</tr>
<tr>
<td>Unassign Calpod serial number from a port.</td>
<td>Calpod:INITialize:ACTive</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Command Line</td>
<td>Value</td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>Initialize ALL channels</td>
<td>Calpod:INITialize:ALL</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Recorrect the selected channel</td>
<td>Calpod:Recorrect:ACTive</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Recorrect ALL channels</td>
<td>Calpod:Recorrect:ALL</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Show refresh dialog</td>
<td>Calpod:SHOW</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Hide refresh dialog</td>
<td>Calpod:HIDE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets impedance state</td>
<td>Calpod:STATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Read Calpod temperature</td>
<td>Calpod:TEMP?</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Trace Type</td>
<td>CALCulate:UNCertainty:DISPLAY:TYPE</td>
<td>DisplayType</td>
<td></td>
</tr>
<tr>
<td>Uncertainty Setup...</td>
<td>CALCulate:UNCertainty:DISPlay:CFACtor</td>
<td>CoverageFactor</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>Coverage factor value</td>
<td>CALCulate:UNCertainty:MODE:NOISe</td>
<td>MeasurementNoiseUncertainty</td>
<td></td>
</tr>
<tr>
<td>Noise contribution</td>
<td>CALCulate:UNCertainty:MODE:CABLE:REPeat</td>
<td>CableRepeatabilityUncertainty</td>
<td></td>
</tr>
<tr>
<td>Cable/connection repeatability contribution</td>
<td>CALCulate:UNCertainty:MODE:ETERm</td>
<td>ErrorTermUncertainty</td>
<td></td>
</tr>
<tr>
<td>Error term uncertainties</td>
<td>CALCulate:UNCertainty:SAVE</td>
<td>WriteUncertaintyFile</td>
<td></td>
</tr>
<tr>
<td>Save uncertainty data</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Add Trace</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Apply to all traces</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Fixtures Tab Commands**
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fixture Simulator</strong></td>
<td><strong>Draft and Fixture Simulator Active</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Copy draft fixture to active fixture</td>
<td>CALCulate:FSIMulator:DRAFT:APPLY</td>
<td>CALCulate:FSIMulator:APPLY</td>
<td>None</td>
</tr>
<tr>
<td>Discards changes on scratch/draft fixture</td>
<td>CALCulate:FSIMulator:DRAFT:DISCard</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Saves SNP file corresponding to the entire scratch fixture</td>
<td>CALCulate:FSIMulator:DRAFT:SAVE</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Saves SNP file corresponding to the entire active fixture</td>
<td>CALCulate:FSIMulator:SAVE</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Loads an active fixture topology file</td>
<td>CALCulate:FSIMulator:TOPology:LOAD</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Saves an active fixture topology file</td>
<td>CALCulate:FSIMulator:TOPology:SAVE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets and returns on power compensation for the active fixture</td>
<td>CALCulate:FSIMulator:POWer:PORT:COMPensate[:STATe]</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Section State Switches**

| Turns port extensions ON and OFF for all ports | CALCulate:FSIMulator:DRAFt:SECTion:EXTension:ENABLe | None |
| Turns all circuits ON | CALCulate:FSIMulator:DRAFt:SECTion:CIRCuit:ENABLe | None |
| and OFF for all ports | **CALCulate:FSIMulator:DRAFT:SECTION:ZCONversion:ENABle**  
**CALCulate:FSIMulator:SECTION:ZCONversion:ENABle?** | None |
|---|---|---|
| Turns all ZConversion ON and OFF (both SE and BAL) | **CALCulate:FSIMulator:DRAFT:SECTION:ZCONversion:ENABle**  
**CALCulate:FSIMulator:SECTION:ZCONversion:ENABle?** | None |
| Circuit Section | **CALCulate:FSIMulator:DRAFT:CIRCuit:CATalog?**  
**CALCulate:FSIMulator:CIRCuit:CATalog?** | None |
| Returns a comma-separated list of circuit numbers created in the draft circuit | **CALCulate:FSIMulator:DRAFT:CIRCuit:NEXT?** | None |
| Returns the next free circuit number than be used to create a new circuit block | | |
| Create a block of the specified block type with the specified fixture port count | CALCulate::FSIMulator:DRAFt:CIRCuit:ADD | None |
| Deletes the specified circuit | CALCulate::FSIMulator:DRAFt:CIRCuit:DELeTe | None |
| Sets the VNA ports for the specified circuit | CALCulate::FSIMulator:DRAFt:CIRCuit:VNA:PORTs, CALCulate::FSIMulator:CIRCuit:VNA:PORTs? | None |
| Sets the Device ports for the specified circuit | CALCulate::FSIMulator:DRAFt:CIRCuit:DEVice:PORTs, CALCulate::FSIMulator:CIRCuit:DEVice:PORTs? | None |
| Sets the resistance circuit element | CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:R  
CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:R2  
CALCulate:FSIMulator:CIRCuit:PARameter:R?  
CALCulate:FSIMulator:CIRCuit:PARameter:R2? | None |
|-----------------------------------|---------------------------------------------------|------|
| Sets the inductance circuit element | CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:L  
CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:L2  
CALCulate:FSIMulator:CIRCuit:PARameter:L?  
CALCulate:FSIMulator:CIRCuit:PARameter:L2? | None |
| Sets the capacitance circuit element | CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:C  
CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:C2  
CALCulate:FSIMulator:CIRCuit:PARameter:C?  
CALCulate:FSIMulator:CIRCuit:PARameter:C2? | None |
| Sets the conductance circuit element | CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:G  
CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:G2  
CALCulate:FSIMulator:CIRCuit:PARameter:G?  
CALCulate:FSIMulator:CIRCuit:PARameter:G2? | None |
| Sets the impedance (Z0) for the ideal block (scalar value). | CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:Z0  
CALCulate:FSIMulator:CIRCuit:PARameter:Z0? | None |
<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets the R value at output of transformer</td>
<td>CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:ROUT</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:CIRCuit:PARameter:ROUT?</td>
<td></td>
</tr>
<tr>
<td>(real only)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets the R value at input of transformer</td>
<td>CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:RIN</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:CIRCuit:PARameter:RIN?</td>
<td></td>
</tr>
<tr>
<td>(real only)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets port extension delay in time</td>
<td>CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:DELay</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:CIRCuit:PARameter:DELay?</td>
<td></td>
</tr>
<tr>
<td>Sets the loss at DC</td>
<td>CALCulate:FSIMulator:DRAFt:CIRCuit:PARameter:LOSS:VALUE</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:CIRCuit:PARameter:LOSS:VALue?</td>
<td></td>
</tr>
<tr>
<td>Sets the filename</td>
<td>CALCulate:FSIMulator:DRAFt:CIRCuit:FILE</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:CIRCuit:FILE?</td>
<td></td>
</tr>
<tr>
<td>Sets whether or not extrapolation is allowed</td>
<td>CALCulate:FSIMulator:DRAFt:CIRCuit:FILE:EXTRapolate</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:CIRCuit:FILE:EXTRapolate?</td>
<td></td>
</tr>
</tbody>
</table>
| embedded or de-embedded? | CALCulate:FSIMulator:DRAFt:CIRCuit:EMBED:TYPE  
CALCulate:FSIMulator:CIRCuit:EMBED:TYPE? | None |
|-------------------------|-----------------------------------------------|------|
| Turns the circuit ON and OFF | CALCulate:FSIMulator:DRAFt:CIRCuit:STATe  
CALCulate:FSIMulator:CIRCuit:STATe? | None |
| Impedance Conversion | CALCulate:FSIMulator:DRAFt:ZCONversion:DIFFerential:STATe  
CALCulate:FSIMulator:ZCONversion:DIFFerential:STATe? | None |
| Sets the differential port impedance conversion function ON/OFF | CALCulate:FSIMulator:DRAFt:ZCONversion:DIFFerential:BPORt:COMPLex  
CALCulate:FSIMulator:ZCONversion:DIFFerential:BPORt:COMPLex?  
CALCulate:FSIMulator:ZCONversion:DIFFerential:LPORt:COMPLex  
CALCulate:FSIMulator:ZCONversion:DIFFerential:LPORt:COMPLex? | None |
| Sets the real impedance value for differential port impedance | CALCulate:FSIMulator:DRAFt:ZCONversion:DIFferential:BPORT:SCALar  
CALCulate:FSIMulator:DRAFt:ZCONversion:DIFferential:LPORt:SCALar  
CALCulate:FSIMulator:ZCONversion:DIFferential:BPORT:S  
CALCulate:FSIMulator:ZCONversion:DIFferential:LPORt:S  
CALCulate:FSIMulator:ZCONversion:DIFferential:BPORT:SCLar?  
CALCulate:FSIMulator:ZCONversion:DIFferential:LPORt:SCLar? | None |
| Sets the common mode port impedance conversion function ON/OFF | CALCulate:FSIMulator:DRAFt:ZCONversion:COMMONmode:STATe  
CALCulate:FSIMulator:ZCONversion:COMMONmode:STATe? | None |
| Sets the complex impedance value for common mode port impedance | CALCulate:FSIMulator:DRAFt:ZCONversion:COMMONmode:BPORT:COMPLex  
CALCulate:FSIMulator:DRAFt:ZCONversion:COMMONmode:LPORt:COMPLex  
CALCulate:FSIMulator:ZCONversion:COMMONmode:BPORT:COMPLex?  
CALCulate:FSIMulator:ZCONversion:COMMONmode:LPORt:COMPLex? | None |
| Sets the real impedance value for common mode port impedance | CALCulate:FSIMulator:DRAFT:ZCONversion:COMMONmode:BPORt:SCALar  
CALCulate:FSIMulator:ZCONversion:COMMONmode:BPORt:SCALar?  
CALCulate:FSIMulator:ZCONversion:COMMONmode:LPORT:SCALar? | None |
| Sets the single-ended ZCONV ON/OFF | CALCulate:FSIMulator:DRAFT:ZCONversion:SENDed:PORT:STATe | None |
| Sets the complex impedance value for SE port impedance | CALCulate:FSIMulator:DRAFT:ZCONversion:SENDed:PORT:COMPLex  
CALCulate:FSIMulator:ZCONversion:SENDed:PORT:COMPlex? | None |
| Sets the real impedance value for SE port impedance | CALCulate:FSIMulator:DRAFT:ZCONversion:SENDed:PORT:SCALar  

Port Extension
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets media type</td>
<td>CALCulate:FSIMulator:DRAFt:EXTension:PORT:MEDium</td>
<td>None</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:EXTension:PORT:MEDium?</td>
<td></td>
</tr>
<tr>
<td>Sets waveguide cutoff frequency</td>
<td>CALCulate:FSIMulator:DRAFt:EXTension:PORT:WAVEguide:FCUToff</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:EXTension:PORT:WAVEguide:FCUToff?</td>
<td></td>
</tr>
<tr>
<td>Sets couple to system media</td>
<td>CALCulate:FSIMulator:DRAFt:EXTension:PORT:WAVEguide:COUPLE</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:EXTension:PORT:WAVEguide:COUPLE?</td>
<td></td>
</tr>
<tr>
<td>Sets velocity factor</td>
<td>CALCulate:FSIMulator:DRAFt:EXTension:PORT:VELocity:FACtor</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:EXTension:PORT:VELocity:FACtor?</td>
<td></td>
</tr>
<tr>
<td>Sets coupling to system velocity factor</td>
<td>CALCulate:FSIMulator:DRAFt:EXTension:PORT:VELocity:COUPLE</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:EXTension:PORT:VELocity:COUPLE?</td>
<td></td>
</tr>
<tr>
<td>Sets the port loss at DC</td>
<td>CALCulate:FSIMulator:DRAFt:EXTension:PORT:DCLoss:VALue</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:EXTension:PORT:DCLoss:VALue?</td>
<td></td>
</tr>
<tr>
<td>Action</td>
<td>Command Details</td>
<td>Result</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Turns port extension ON/OFF</td>
<td>CALCulate:FSIMulator:DRAFT:EXTension:PORT[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:EXTension:PORT[:STATe]?</td>
<td></td>
</tr>
<tr>
<td>Moves the port extension block to the right-most side of the circuit sections</td>
<td>CALCulate:FSIMulator:DRAFT:EXTension:PORT:END</td>
<td>None</td>
</tr>
<tr>
<td>Apply Fixtures</td>
<td>ON/OFF</td>
<td>FixturingState</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:STATe</td>
<td></td>
</tr>
<tr>
<td>Power Comp...</td>
<td>Port N</td>
<td>EnablePowerCompensation</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:POWER:COMPensate:MODE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:POWER:PORT:COMPensate:STATe</td>
<td></td>
</tr>
<tr>
<td>Fixture Setup</td>
<td>Change order of operations</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:SENDed:OORDER</td>
<td></td>
</tr>
<tr>
<td>2 and 4-port Extrapolate</td>
<td>CALCulate:FSIMulator:SNP:EXTRapolate</td>
<td>EnableSnPDataExtrapolation</td>
</tr>
<tr>
<td>2-Port Fixturing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port matching ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:STATE</td>
<td>PortMatchingState</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Sets Port Matching circuit model</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:TYPE</td>
<td>PortMatchingCktModel</td>
</tr>
<tr>
<td>Sets Port Matching 'S2P' file name</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:USER:FILename</td>
<td>strPortMatch_S2PFile</td>
</tr>
<tr>
<td>Sets Capacitance 'C' value</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARAMeters:C</td>
<td>PortMatching_C</td>
</tr>
<tr>
<td>Sets Inductance 'L' value</td>
<td>CALCulate:FSIMulator:SENDed:PMCircuit:PORT:PARAMeters:L</td>
<td>PortMatching_L</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>De-embed ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:DEEMbed:STATe</td>
<td>Port2PdeembedState</td>
</tr>
<tr>
<td>Sets De-embedding circuit model</td>
<td>CALCulate:FSIMulator:SENDed:DEEMbed:PORT</td>
<td>Port2PdeembedCktModel</td>
</tr>
<tr>
<td>Sets De-embedding 'S2P' file name</td>
<td>CALCulate:FSIMulator:SENDed:DEEM:PORT:USER:FI LEName</td>
<td>strPort2Pdeembed_S2PFile</td>
</tr>
<tr>
<td>Port Impedance ON and OFF</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:STATe</td>
<td>PortArbzState</td>
</tr>
<tr>
<td>Port Z Real</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:REAL</td>
<td>PortArbzReal</td>
</tr>
<tr>
<td>Port Z Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:IMAG</td>
<td>PortArbzImag</td>
</tr>
<tr>
<td>Port Z Real and Imag</td>
<td>CALCulate:FSIMulator:SENDed:ZCONversion:PORT:Z0</td>
<td>PortArbzZ0</td>
</tr>
<tr>
<td>4-Port Network Embed/De-embed commands</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specifies the PNA / DUT topology</td>
<td>\texttt{CALCulate:FSIMulator:EMBed:TYPE}</td>
<td>Embed4PortTopology</td>
</tr>
<tr>
<td>Specifies the 4-port touchstone file</td>
<td>\texttt{CALCulate:FSIMulator:EMBed:NETWork:FILENAME}</td>
<td>Embed4PortNetworkFilename</td>
</tr>
<tr>
<td>4-port remap</td>
<td>\texttt{CALCulate:FSIMulator:EMBed:NETWork&lt;n&gt;:PMAP}</td>
<td>NetworkPortMap</td>
</tr>
<tr>
<td>Turn ON or OFF</td>
<td>\texttt{CALCulate:FSIMulator:EMBed:STATE}</td>
<td>Embed4PortState</td>
</tr>
</tbody>
</table>

### Differential Port Arbitrary Impedance
<table>
<thead>
<tr>
<th>Sets the impedance value</th>
<th>\texttt{CALCulate:FSIMulator:BAunL:DZConversion:BPORt:Z_0}</th>
<th>\texttt{DiffZConvPortZ0}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets real part of impedance</td>
<td>\texttt{CALCulate:FSIMulator:BAunL:DZConversion:BPORt:REAL}</td>
<td>\texttt{DiffZConvPortReal}</td>
</tr>
<tr>
<td>Sets imaginary part of impedance</td>
<td>\texttt{CALCulate:FSIMulator:BAunL:DZConversion:BPORt:IMAG}</td>
<td>\texttt{DiffZConvPortImag}</td>
</tr>
<tr>
<td>Turn ON or OFF</td>
<td>\texttt{CALCulate:FSIMulator:BAunL:DZConversion:STATe}</td>
<td>\texttt{DiffZConvState}</td>
</tr>
</tbody>
</table>

### Common Mode Port Arbitrary Impedance

<table>
<thead>
<tr>
<th>Sets the impedance value</th>
<th>\texttt{CALCulate:FSIMulator:BAunL:CZConversion:BPORt:Z_0}</th>
<th>\texttt{CmnModeZConvPortZ0}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets real part of impedance</td>
<td>\texttt{CALCulate:FSIMulator:BAunL:CZConversion:BPORt:REAL}</td>
<td>\texttt{CmnModeZConvPortReal}</td>
</tr>
<tr>
<td>Operation</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>Sets imaginary part of impedance</td>
<td>CALCulate:FSIMulator:BALun:CZConversion:BPORt:IMAG</td>
<td>CmnModeZConvPortimag</td>
</tr>
<tr>
<td>Turn ON or OFF</td>
<td>CALCulate:FSIMulator:BALun:CZConversion:STATE</td>
<td>CmnModeZConvState</td>
</tr>
<tr>
<td><strong>Differential Port Matching</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets type of circuit to embed</td>
<td>CALCulate:FSIMulator:BALun:DMCircuit:BPORt</td>
<td>DiffPortMatchMode</td>
</tr>
<tr>
<td>Specifies the 2-port touchstone file</td>
<td>CALCulate:FSIMulator:BALun:DMCircuit:BPORt:USER:FILENAME</td>
<td>DiffPortMatchUserFilename</td>
</tr>
<tr>
<td>Sets Capacitance value</td>
<td>CALCulate:FSIMulator:BALun:DMCircuit:BPORt:PARAMeters:C</td>
<td>DiffPortMatch_C</td>
</tr>
<tr>
<td>Sets Conductance value</td>
<td>CALCulate:FSIMulator:BALun:DMCircuit:BPORt:PARAMeters:G</td>
<td>DiffPortMatch_G</td>
</tr>
<tr>
<td>Sets</td>
<td>Inductance value</td>
<td>CALCulate:FSIMulator:BA Lun:DM Circuit:BPORt:PARa meters:L</td>
</tr>
<tr>
<td>-------</td>
<td>------------------</td>
<td>-------------------------------------------------------------</td>
</tr>
<tr>
<td>Sets</td>
<td>Resistance value</td>
<td>CALCulate:FSIMulator:BA Lun:DM Circuit:BPORt:PARa meters:R</td>
</tr>
<tr>
<td>Turns</td>
<td>ON/OFF</td>
<td>CALCulate:FSIMulator:BA Lun:DM Circuit:STATe</td>
</tr>
<tr>
<td>Remote ONLY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create Cal Set with De-embedded fixture removed</td>
<td>CSET:FIXTure:DEEMbed</td>
<td>Deembed</td>
</tr>
<tr>
<td>Create Cal Set with Matching network included</td>
<td>CSET:FIXTure:EMBed</td>
<td>Embed</td>
</tr>
<tr>
<td>Cal Plane Manager.</td>
<td>Characterize a fixture</td>
<td>CSET:FIXTure:CHARacterize</td>
</tr>
<tr>
<td>..</td>
<td>Creates a single S2P file from two existing files</td>
<td>CSET:FIXTure:CASCade</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Auto Fixture Removal. ..</td>
<td></td>
<td>CSET:FIXTure:DEEMbed</td>
</tr>
</tbody>
</table>
## CF_Channel Commands

### Channel 1 - 8 Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel 1</td>
<td>On/Off</td>
<td>None</td>
<td>Add</td>
</tr>
<tr>
<td>Channel 2</td>
<td>On/Off</td>
<td>None</td>
<td>Add</td>
</tr>
<tr>
<td>Channel 3</td>
<td>On/Off</td>
<td>None</td>
<td>Add</td>
</tr>
<tr>
<td>Channel 4</td>
<td>On/Off</td>
<td>None</td>
<td>Add</td>
</tr>
<tr>
<td>Channel 5</td>
<td>On/Off</td>
<td>None</td>
<td>Add</td>
</tr>
<tr>
<td>Channel 6</td>
<td>On/Off</td>
<td>None</td>
<td>Add</td>
</tr>
<tr>
<td>Channel 7</td>
<td>On/Off</td>
<td>None</td>
<td>Add</td>
</tr>
<tr>
<td>Channel 8</td>
<td>On/Off</td>
<td>None</td>
<td>Add</td>
</tr>
</tbody>
</table>

### Channel Setup Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>ChN</td>
<td>SENSE:CLASs:NAME?</td>
<td>ActiveChannel</td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------</td>
<td>---------------------------</td>
<td></td>
</tr>
<tr>
<td>Add Channel</td>
<td>New Trace + Channel</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>New Trace + Channel + Window</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Copy Channel</td>
<td>Copy to Active Window</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Copy to New Window</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Copy Channel</td>
<td>SYSTem:MACRo:COPY:CHANnel[:TO]</td>
<td>CopyToChannel</td>
<td></td>
</tr>
<tr>
<td>Delete Channel</td>
<td>ChN</td>
<td>SYSTem:CHANnels:DELeete</td>
<td>RemoveChannelNumber</td>
</tr>
</tbody>
</table>
## CF_Display Commands

### Window 1-8 Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Window 1</td>
<td>On/Off</td>
<td>DISPLAY:WINDow[:STATE]</td>
<td>Add</td>
</tr>
<tr>
<td>Window 2</td>
<td>On/Off</td>
<td>DISPLAY:WINDow[:STATE]</td>
<td>Add</td>
</tr>
<tr>
<td>Window 3</td>
<td>On/Off</td>
<td>DISPLAY:WINDow[:STATE]</td>
<td>Add</td>
</tr>
<tr>
<td>Window 4</td>
<td>On/Off</td>
<td>DISPLAY:WINDow[:STATE]</td>
<td>Add</td>
</tr>
<tr>
<td>Window 5</td>
<td>On/Off</td>
<td>DISPLAY:WINDow[:STATE]</td>
<td>Add</td>
</tr>
<tr>
<td>Window 6</td>
<td>On/Off</td>
<td>DISPLAY:WINDow[:STATE]</td>
<td>Add</td>
</tr>
<tr>
<td>Window 7</td>
<td>On/Off</td>
<td>DISPLAY:WINDow[:STATE]</td>
<td>Add</td>
</tr>
<tr>
<td>Window 8</td>
<td>On/Off</td>
<td>DISPLAY:WINDow[:STATE]</td>
<td>Add</td>
</tr>
</tbody>
</table>

### Window Setup Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>WinN</td>
<td>DISPLAY:WINDow:TRACe:SELect</td>
<td>ActivateWindow</td>
</tr>
<tr>
<td>Window</td>
<td>Enable</td>
<td>DISPlay::WINDow::TIITLE[:STATe]</td>
<td>TitleState</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
<td>---------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Title</td>
<td></td>
<td>DISPlay::WINDow::TIITLE[:DATA]</td>
<td>Title</td>
</tr>
<tr>
<td>Add Window</td>
<td>New Window</td>
<td>DISPlay::WINDow[:STATe]</td>
<td>Add</td>
</tr>
<tr>
<td></td>
<td>New Trace + Window</td>
<td>DISPlay::WINDow::TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td></td>
<td>None</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Delete Window</td>
<td>WinN</td>
<td>DISPlay::WINDow[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Move Window...</td>
<td></td>
<td>DISPlay::WINDow::TRACe::MOVE</td>
<td>Move</td>
</tr>
<tr>
<td>Window Layout</td>
<td>1 Window</td>
<td>DISPlay::ARRange</td>
<td>ArrangeWindows</td>
</tr>
<tr>
<td></td>
<td>2 Windows</td>
<td>DISPlay::ARRange</td>
<td>ArrangeWindows</td>
</tr>
<tr>
<td></td>
<td>3 Windows</td>
<td>DISPlay::ARRange</td>
<td>ArrangeWindows</td>
</tr>
<tr>
<td></td>
<td>4 Windows</td>
<td>DISPlay::ARRange</td>
<td>ArrangeWindows</td>
</tr>
<tr>
<td>1 Trace per Window</td>
<td>DISPlay:ARRange</td>
<td>ArrangeWindows</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>1 Channel per Window</td>
<td>DISPlay:ARRange</td>
<td>ArrangeWindows</td>
<td></td>
</tr>
<tr>
<td>Tile Windows</td>
<td>DISPlay:ARRange</td>
<td>ArrangeWindows</td>
<td></td>
</tr>
</tbody>
</table>

**Sheet Setup Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>Sheet N</td>
<td>DISPlay:SHEet:STATe</td>
<td>None</td>
</tr>
<tr>
<td>Sheet Title...</td>
<td></td>
<td>DISPlay:SHEet:TITle:DATA</td>
<td>None</td>
</tr>
<tr>
<td>Add Sheet</td>
<td>New Sheet</td>
<td>DISPlay:SHEet:STATe</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>New Trace + Sheet</td>
<td>DISPlay:WINDow:FEED</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>New Trace + Channel + Sheet</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Delete Sheet</td>
<td>Sheet N</td>
<td>DISPlay:SHEet:STATe</td>
<td>None</td>
</tr>
<tr>
<td>Sheet Layout</td>
<td>1 Sheet</td>
<td>DISPlay:SHEet:ARRange</td>
<td>None</td>
</tr>
<tr>
<td>1 Trace per Sheet</td>
<td>DISPlay:SHEet:ARRange</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>1 Channel per Sheet</td>
<td>DISPlay:SHEet:ARRange</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>1 Window per Sheet</td>
<td>DISPlay:SHEet:ARRange</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Display Setup Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trace Maximize</td>
<td>ON/OFF</td>
<td>DISPlay:TMAX</td>
<td>TraceMax</td>
</tr>
<tr>
<td>Window Max</td>
<td>ON/OFF</td>
<td>DISPlay:WINDow:SIZe</td>
<td>WindowState</td>
</tr>
<tr>
<td>Show Table</td>
<td>None</td>
<td>DISPlay:WINDow:TABLe</td>
<td>ShowTable</td>
</tr>
<tr>
<td>Marker</td>
<td>DISPlay:WINDow:TABLe</td>
<td>ShowTable</td>
<td></td>
</tr>
<tr>
<td>Limit</td>
<td>DISPlay:WINDow:TABLe</td>
<td>ShowTable</td>
<td></td>
</tr>
<tr>
<td>Ripple</td>
<td>DISPlay:WINDow:TABLe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Segment</td>
<td>Display:Window:Table</td>
<td>ShowTable</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------------------------------------------</td>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>Distortion</td>
<td>Display:Window:Table</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Spurious</td>
<td>Display:Window:Table:SPURious:ENABLE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Integrated Noise</td>
<td>Display:Window:Table:INoise:ENABLE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Spot Noise</td>
<td>Display:Window:Table:SNOise:ENABLE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Customize Display...</td>
<td>Display:Window:ANNoteation:Y[:STATE]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Trace Status</td>
<td>Display:Window:ANNoteation:TRACe[:STATE]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Y-axis Labels</td>
<td>Display:Window:ANNoteation:Y[:STATE]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>X-axis Labels</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Trace Annotation for All Traces</td>
<td>Display:Window:ANNoteation:TRACe:SCOPE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Markers</td>
<td>Display:Window:ANNoteation:MARKer[:STATE]</td>
<td>MarkerReadout</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------------</td>
<td>---------------------</td>
<td></td>
</tr>
<tr>
<td>Large Readout</td>
<td><code>DISPlay:WINDow:ANNotation:MARKer:SIZE</code></td>
<td>MarkerReadoutSize</td>
<td></td>
</tr>
<tr>
<td>Active Trace Only</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Readouts Per Trace</td>
<td><code>DISPlay:WINDow:ANNotation:MARKer:NUMBer</code></td>
<td>MarkerReadoutsPerTrace</td>
<td></td>
</tr>
<tr>
<td>Symbols Above Trace</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Independent Position Per Trace</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Active TrN</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Marker Colors...</td>
<td>DISPlay:COLor:TRACe:MARKer</td>
<td>Markers</td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>N Trace: Markers</td>
<td>DISPlay:COLor:TRACe:MARKer</td>
<td>Markers</td>
<td></td>
</tr>
<tr>
<td>N Trace: Memory Markers</td>
<td>DISPlay:COLor:TRACe:MMARker</td>
<td>MemoryMarkers</td>
<td></td>
</tr>
<tr>
<td>Change Color...</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Reset Color</td>
<td>DISPlay:COLor:RESet</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Save Theme...</td>
<td>DISPlay:COLor:STORE</td>
<td>StoreTheme</td>
<td></td>
</tr>
<tr>
<td>Recall Theme...</td>
<td>DISPlay:COLor:LOAD</td>
<td>LoadTheme</td>
<td></td>
</tr>
<tr>
<td>Reset Theme...</td>
<td>DISPlay:COLor:RESet</td>
<td>ResetTheme</td>
<td></td>
</tr>
</tbody>
</table>

**Grid & Table**

<table>
<thead>
<tr>
<th>Grid Lines - Solid</th>
<th>Dotted</th>
<th>DISPlay:WINDow:TRACe:GRATicule:GRID:LTYPe</th>
<th>GridLineType</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Property</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------</td>
<td>----------</td>
<td></td>
</tr>
<tr>
<td>Y-axis Divisions - 2 to 30</td>
<td>DISPlay:WINDow:TRACe:Y[:SCALE]:PDIVision</td>
<td>YScale</td>
<td></td>
</tr>
<tr>
<td>Show Table - None, Marker,</td>
<td>DISPlay:WINDow:TABLe</td>
<td>ShowTable</td>
<td></td>
</tr>
<tr>
<td>Limit, Ripple, and Segment</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Toolbars</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>DISPlay:TOOLbar:ENTRY[:STATE]</td>
<td>ShowToolbar</td>
<td></td>
</tr>
<tr>
<td>Hardkey</td>
<td>DISPlay:TOOLbar:KEYS[:STATE]</td>
<td>ShowToolbar</td>
<td></td>
</tr>
<tr>
<td>Port Extensions</td>
<td>DISPlay:TOOLbar:EXTensions[:STATE]</td>
<td>ShowToolbar</td>
<td></td>
</tr>
<tr>
<td>Transform</td>
<td>DISPlay:TOOLbar:TRANSform[:STATE]</td>
<td>ShowToolbar</td>
<td></td>
</tr>
<tr>
<td>Marker</td>
<td>DISPlay:TOOLbar:MARKer[:STATE]</td>
<td>ShowToolbar</td>
<td></td>
</tr>
<tr>
<td>Cal Set Viewer</td>
<td>DISPlay:TOOLbar:CSET[:STATE]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Main Title Bar</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Menu Bar</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Active Entry Toolbar</td>
<td>DISPLAY:TOOLbar:ENTRY[:STATe]</td>
<td>ShowToolbar</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Status Bar</td>
<td>DISPLAY:ANNOTation[:STATus]</td>
<td>ShowStatusBar</td>
<td></td>
</tr>
<tr>
<td>- 2 Bars</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>- Clock</td>
<td>SYSTem:CLOCk[:STATe]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Display Sheet Tabs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Top</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Bottom</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Tools</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tool</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Active Entry</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Colors</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>DISPLAY:COLOR:RESet</td>
<td>ResetTheme</td>
<td></td>
</tr>
<tr>
<td>Light Gray</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>Dark Gray</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Blue</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Display Colors...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Print Colors...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Background</td>
<td>Display:COLor:BACKground</td>
<td>Background</td>
<td></td>
</tr>
<tr>
<td>Active Background</td>
<td>Display:COLor:ABACKground</td>
<td>ActiveBackground</td>
<td></td>
</tr>
<tr>
<td>Grid</td>
<td>Display:COLor:GRAT2</td>
<td>Grid</td>
<td></td>
</tr>
<tr>
<td>Active Labels, Grid Frame</td>
<td>Display:COLor:GRAT1</td>
<td>ActiveLabels</td>
<td></td>
</tr>
<tr>
<td>Inactive Window Labels</td>
<td>Display:COLor:ILABEL</td>
<td>InactiveLabels</td>
<td></td>
</tr>
<tr>
<td>Failed Trace</td>
<td>Display:COLor:LIM1</td>
<td>FailedTraces</td>
<td></td>
</tr>
<tr>
<td>N Trace: Data and Limits</td>
<td>DISPlay:COLor:TRACe:DATA</td>
<td>DataAndLimits</td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td>N Trace: Memory</td>
<td>DISPlay:COLor:TRACe:MEMory</td>
<td>Memory</td>
<td></td>
</tr>
<tr>
<td>N Trace: Markers</td>
<td>DISPlay:COLor:TRACe:MARKer</td>
<td>Markers</td>
<td></td>
</tr>
<tr>
<td>N Trace: Memory Markers</td>
<td>DISPlay:COLor:TRACe:MMARker</td>
<td>MemoryMarkers</td>
<td></td>
</tr>
<tr>
<td>Change Color...</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Reset Color</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Save Theme...</td>
<td>DISPlay:COLor:STORe</td>
<td>StoreTheme</td>
<td></td>
</tr>
<tr>
<td>Recall Theme...</td>
<td>DISPlay:COLor:LOAD</td>
<td>LoadTheme</td>
<td></td>
</tr>
<tr>
<td>Reset Theme...</td>
<td>DISPlay:COLor:RESet</td>
<td>ResetTheme</td>
<td></td>
</tr>
<tr>
<td>Touchscreen</td>
<td>ON/OFF</td>
<td>SYSTEM:TOUCHscreen[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td>-----------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Calibration Touchscreen</td>
<td>ON/OFF</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Display Update</td>
<td>ON/OFF</td>
<td>DISPlay:UPDate[:STATE] DISPlay:UPDate:IMMediate</td>
<td>None</td>
</tr>
</tbody>
</table>
# CF_Format Commands

## Format 1 Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log Mag</td>
<td>On/Off</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Lin Mag</td>
<td>On/Off</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Phase</td>
<td>On/Off</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Delay</td>
<td>On/Off</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Smith</td>
<td>On/Off</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Polar</td>
<td>On/Off</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>SWR</td>
<td>On/Off</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Group Delay Aperture...</td>
<td>Points</td>
<td>CALCulate:MEASure:GDElay:POINts</td>
<td>Points</td>
</tr>
<tr>
<td></td>
<td>Percent of Span</td>
<td>CALCulate:MEASure:GDElay:PERCent</td>
<td>Percent</td>
</tr>
<tr>
<td></td>
<td>Frequency</td>
<td>CALCulate:MEASure:GDElay:FREQuency</td>
<td>Frequency</td>
</tr>
</tbody>
</table>

## Format 2 Tab Commands
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real</td>
<td>On/Off</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Imaginary</td>
<td>On/Off</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Unwrapped Phase</td>
<td>On/Off</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Positive Phase</td>
<td>On/Off</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Inverted Smith</td>
<td>On/Off</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Temperature</td>
<td>Kelvin (K)</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td></td>
<td>Fahrenheit (° F)</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td></td>
<td>Celsius (° C)</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
</tbody>
</table>
CF_Freq Commands

The Freq softkeys vary depending on which measurement class is currently active. Click on a measurement class link below to view the corresponding softkeys.

Standard
Active (Hot) Parameters
Gain Compression
Differential I/Q
IM Spectrum and IM Spectrum Converters
Swept IMD and Swept IMD Converters
Modulation Distortion and Modulation Distortion Converters
Noise Figure Cold Source
Phase Noise
Gain Compression Converters and Noise Figure Converters
Scalar Mixer/Converter + Phase and Vector Mixer/Converter
Spectrum Analyzer
## CF_Freq Commands - DIQ

Freq Commands - Differential I/Q Measurement Class

Click [here](#) to view links to Freq commands for all Measurement Classes.

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIQ Setup...</td>
<td>Measurement Set Up</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Frequency Range</td>
<td>SENSE:DIQ:FREQ:RANGE:STARt</td>
<td>RangeStartFrequency</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:DIQ:FREQ:RANGE:STOP</td>
<td>RangeStopFrequency</td>
<td></td>
</tr>
<tr>
<td>Add Source...</td>
<td>SYSTem:CONFigure:EDEVice:ADD</td>
<td>Add (External Device)</td>
<td></td>
</tr>
<tr>
<td>Remove</td>
<td>SYSTem:CONFigure:EDEVice:REMove</td>
<td>Remove</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>SYSTem:CONFigure:EDEVice:ADD</td>
<td>Add (External Device)</td>
<td></td>
</tr>
<tr>
<td>Device Type</td>
<td>SYSTem:CONFigure:EDEVice:DTYPe</td>
<td>DeviceType</td>
<td></td>
</tr>
<tr>
<td>Driver</td>
<td>SYSTem:CONFigure:EDEVice:DRIVer</td>
<td>Driver</td>
<td></td>
</tr>
<tr>
<td>Active - Show in UI</td>
<td>SYSTem:CONFigure:EDEVice:STATe</td>
<td>Active</td>
<td></td>
</tr>
<tr>
<td>Power On (All Channels)</td>
<td>OUTPut[:STATe]</td>
<td>SourcePowerState</td>
<td></td>
</tr>
</tbody>
</table>
# CFFreq Commands - GCA

## Freq Commands - Gain Compression Measurement Class

Click [here](#) to view links to Freq commands for all Measurement Classes.

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td></td>
<td>SENSe:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td></td>
<td>SENSe:FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>Center</td>
<td></td>
<td>SENSe:FREQuency:CENTer</td>
<td>CenterFrequency</td>
</tr>
<tr>
<td>Span</td>
<td></td>
<td>SENSe:FREQuency:SPAN</td>
<td>FrequencySpan</td>
</tr>
<tr>
<td>Step</td>
<td></td>
<td>SENSe:FREQuency:STEP</td>
<td>FrequencyStep</td>
</tr>
<tr>
<td>Frequency Offset...</td>
<td>Frequency Offset (ON/OFF)</td>
<td>SENSe:FOM[:STATe]</td>
<td>State</td>
</tr>
<tr>
<td>Mode - Coupled and Un-coupled</td>
<td>SENSe:FOM:RANGe:COUPled</td>
<td>Coupled</td>
<td></td>
</tr>
<tr>
<td>Sweep Type</td>
<td>SENSe:FOM:RANGe:SWEep:TYPE</td>
<td>SweepType</td>
<td></td>
</tr>
<tr>
<td>Settings</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td></td>
<td>SENSe:FOM:RANGe:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Annotation - Primary, Source, and Receivers</td>
<td>SENSe:FOM:RANGe:NAME?</td>
<td>Name</td>
<td></td>
</tr>
<tr>
<td>X-Axis Point Spacing</td>
<td>SENSe:FOM:RANGe:SEGMen:SWEP:POINts</td>
<td>NumberOfPoints</td>
<td></td>
</tr>
<tr>
<td>GCA Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# CF_Freq Commands - GCX_NFX

_Freq Commands - Gain Compression Converters and Noise Figure Converters Measurement Classes_

Click [here](#) to view links to Freq commands for all Measurement Classes.

<table>
<thead>
<tr>
<th><strong>Main Tab Commands</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
<td><strong>Sub-item</strong></td>
</tr>
<tr>
<td>Start...</td>
<td>Input</td>
</tr>
<tr>
<td>Calculate Input and Output frequencies</td>
<td></td>
</tr>
<tr>
<td>Input to swept or fixed</td>
<td></td>
</tr>
<tr>
<td>Input start frequency</td>
<td></td>
</tr>
<tr>
<td>Input stop frequency</td>
<td></td>
</tr>
<tr>
<td>Input power level</td>
<td></td>
</tr>
<tr>
<td>Input fixed frequency</td>
<td></td>
</tr>
<tr>
<td>LO1</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>LO freq fixed or swept</td>
<td>SENSe:MIXer:LO:FREQuency:MODE</td>
</tr>
<tr>
<td>LO fixed frequency</td>
<td>SENSe:MIXer:LO:FREQuency:FIXed</td>
</tr>
<tr>
<td>LO start frequency</td>
<td>SENSe:MIXer:LO:FREQuency:STARt</td>
</tr>
<tr>
<td>LO stop frequency</td>
<td>SENSe:MIXer:LO:FREQuency:STOP</td>
</tr>
<tr>
<td>Input Greater / Less that LO</td>
<td>SENSe:MIXer:LO:FREQuency:ILTI</td>
</tr>
<tr>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>Sideband (high or low)</td>
<td>SENSe:MIXer:OUTPut:FREQuency:SIDeband</td>
</tr>
<tr>
<td>Output start frequency</td>
<td>SENSe:MIXer:OUTPut:FREQuency:STARt</td>
</tr>
<tr>
<td>Output stop frequency</td>
<td>SENSe:MIXer:OUTPut:FREQuency:STOP</td>
</tr>
<tr>
<td>Output to swept or fixed</td>
<td>SENSe:MIXer:OUTPut:FREQuency:MODE</td>
</tr>
<tr>
<td></td>
<td>Output fixed frequency</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><strong>Stop...</strong></td>
<td>Input</td>
</tr>
<tr>
<td></td>
<td>LO1</td>
</tr>
<tr>
<td></td>
<td>Output</td>
</tr>
<tr>
<td><strong>Center...</strong></td>
<td>Input</td>
</tr>
<tr>
<td></td>
<td>LO1</td>
</tr>
<tr>
<td></td>
<td>Output</td>
</tr>
<tr>
<td><strong>Span...</strong></td>
<td>Input</td>
</tr>
<tr>
<td></td>
<td>LO1</td>
</tr>
<tr>
<td></td>
<td>Output</td>
</tr>
<tr>
<td><strong>CW...</strong></td>
<td>Sweep Type</td>
</tr>
<tr>
<td></td>
<td>Data Acquisition Mode</td>
</tr>
<tr>
<td></td>
<td>(Gain Compression</td>
</tr>
<tr>
<td></td>
<td>Converters</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td><strong>Measurement Class only</strong></td>
<td></td>
</tr>
<tr>
<td>X-Axis Display (Noise Figure Converters Measurement Class only)</td>
<td>CALCulate:MEASURE:MIXer:XAXis</td>
</tr>
<tr>
<td>Sweep Settings</td>
<td></td>
</tr>
<tr>
<td>GCX Setup... (Gain Compression Converters Measurement Class only)</td>
<td></td>
</tr>
<tr>
<td>NFX Setup... (Noise Figure Converters Measurement Class only)</td>
<td></td>
</tr>
</tbody>
</table>
## CF_Freq Commands - HotS22

**Freq Commands - Active (Hot) Parameters Measurement Class**

Click [here](#) to view links to Freq commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td></td>
<td>SENSE: FREQuency: START</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td></td>
<td>SENSE: FREQuency: STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>Center</td>
<td></td>
<td>SENSE: FREQuency: CENTER</td>
<td>CenterFrequency</td>
</tr>
<tr>
<td>Span</td>
<td></td>
<td>SENSE: FREQuency: SPAN</td>
<td>FrequencySpan</td>
</tr>
<tr>
<td>Step</td>
<td></td>
<td>SENSE: SWEep: STEP</td>
<td>FrequencyStep</td>
</tr>
<tr>
<td>CW</td>
<td></td>
<td>SENSE: FREQuency: CW</td>
<td>CWFrequency</td>
</tr>
<tr>
<td>Frequency Offset...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AHP Setup...</td>
<td>Sweep</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sweep Type</td>
<td>SENSE: ACTive: SWEep: TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Source</td>
<td>Command</td>
<td>Value</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------</td>
<td>----------------------------</td>
<td></td>
</tr>
<tr>
<td>Start Frequency</td>
<td>SENSE:FREQuency:STARt</td>
<td>StartFrequency</td>
<td></td>
</tr>
<tr>
<td>Center Frequency</td>
<td>SENSE:FREQuency:CENTer</td>
<td>CenterFrequency</td>
<td></td>
</tr>
<tr>
<td>Stop Frequency</td>
<td>SENSE:FREQuency:STOP</td>
<td>StopFrequency</td>
<td></td>
</tr>
<tr>
<td>Frequency Span</td>
<td>SENSE:FREQuency:SPAN</td>
<td>FrequencySpan</td>
<td></td>
</tr>
<tr>
<td>Frequency Sweep</td>
<td>SENSE:ACTive:SWEep:TYPE</td>
<td>SweepType</td>
<td></td>
</tr>
<tr>
<td>Number of Frequencies</td>
<td>SENSE:SWEep:STEP</td>
<td>FrequencyStep</td>
<td></td>
</tr>
<tr>
<td>Power Level</td>
<td>SENSE:ACTive:DISplay:TRACe:IPWer</td>
<td>DisplayInputPower</td>
<td></td>
</tr>
<tr>
<td>Start Power</td>
<td>SENSE:ACTive:SWEep:POWer:STARt</td>
<td>StartPowerIn3DSweep</td>
<td></td>
</tr>
<tr>
<td>Stop Power</td>
<td>SENSE:ACTive:SWEep:POWer:STOP</td>
<td>StopPowerIn3DSweep</td>
<td></td>
</tr>
<tr>
<td>Number Of Powers</td>
<td>SENSE:ACTive:SWEep:POWer:STEP</td>
<td>PowerStepsIn3DSweep</td>
<td></td>
</tr>
<tr>
<td>CW Frequency</td>
<td>SENSE:FREQuency:CW</td>
<td>CWFrequency</td>
<td></td>
</tr>
<tr>
<td>Extraction</td>
<td>IF Bandwidth</td>
<td>IFBandwidth</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>--------------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:BWIDth:RESolution</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Absolute Power</td>
<td>SENSE:ACTive:TTONe:MODE</td>
<td>ExtractionToneMode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:ACTive:TTONe:ABSolute</td>
<td>AbsoluteExtractionToneLevel</td>
<td></td>
</tr>
<tr>
<td>Relative to Input Power</td>
<td>SENSE:ACTive:TTONe:MODE</td>
<td>ExtractionToneMode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:ACTive:TTONe:RELative</td>
<td>RelativeExtractionToneLevel</td>
<td></td>
</tr>
<tr>
<td>Number of Phases</td>
<td>SENSE:ACTive:SWEep:PHASE:POINt</td>
<td>PhaseSweepPoints</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DC Sources...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RF Path</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power On - all channels</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Source</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source Port</td>
<td>None</td>
<td>SelectPort</td>
<td></td>
</tr>
<tr>
<td>Source Attenuator</td>
<td>SOURce:POWer&lt;port&gt;:ATTenuation</td>
<td>Attenuator</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>Receiver A Attenuator</td>
<td>SENSe:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
<td></td>
</tr>
<tr>
<td>ALC Hardware</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
<td></td>
</tr>
<tr>
<td>Receiver Leveling</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver</td>
<td>State</td>
<td></td>
</tr>
<tr>
<td>Extraction</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extraction Port</td>
<td>None</td>
<td>SelectPort</td>
<td></td>
</tr>
<tr>
<td>Source Attenuator</td>
<td>SOURce:POWer&lt;port&gt;:ATTenuation</td>
<td>Attenuator</td>
<td></td>
</tr>
<tr>
<td>Receiver C Attenuator</td>
<td>SENSe:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
<td></td>
</tr>
<tr>
<td>ALC Hardware</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
<td></td>
</tr>
<tr>
<td>Receiver Leveling</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver</td>
<td>State</td>
<td></td>
</tr>
<tr>
<td>RF Path Config...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X-axis</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Value</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------------------------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td>Active</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
<td></td>
</tr>
<tr>
<td>Select all traces</td>
<td>DISPlay:WINDow:TRACe:SELect</td>
<td>View</td>
<td></td>
</tr>
<tr>
<td>X-axis display</td>
<td>SENSE:ACTive:SWEep:TYPE</td>
<td>SweepType</td>
<td></td>
</tr>
<tr>
<td>Fixed Parameters</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input Power</td>
<td>SENSE:ACTive:DISPlay:TRACe:IPWer</td>
<td>DisplayInputPower</td>
<td></td>
</tr>
<tr>
<td>Frequency</td>
<td>SENSE:FREQuency:CENTer</td>
<td>CenterFrequency</td>
<td></td>
</tr>
<tr>
<td>Enable Interpolation</td>
<td>SENSE:ACTive:DISP:INTERpolate[:STATe]</td>
<td>DisplayInterpolationState</td>
<td></td>
</tr>
<tr>
<td>Name (DC source)</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
## CF_Freq Commands - IMD_IMDX

Freq Commands - Swept IMD and Swept IMD Converters Measurement Classes

Click [here](#) to view links to Freq commands for all Measurement Classes.

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Fc</td>
<td></td>
<td><code>SENSe:IMD:FREQuency:FCENter:STARt</code></td>
<td><code>FrequencyCenterStart</code></td>
</tr>
<tr>
<td>Stop Fc</td>
<td></td>
<td><code>SENSe:IMD:FREQuency:FCENter:STOP</code></td>
<td><code>FrequencyCenterStop</code></td>
</tr>
<tr>
<td>Center Fc</td>
<td></td>
<td><code>SENSe:IMD:FREQuency:FCENter</code></td>
<td><code>FrequencyCenter</code></td>
</tr>
<tr>
<td>Span Fc</td>
<td></td>
<td><code>SENSe:IMD:FREQuency:FCENter:SPAN</code></td>
<td><code>FrequencyCenterSpan</code></td>
</tr>
<tr>
<td>Fixed DeltaF</td>
<td></td>
<td><code>SENSe:IMD:FREQuency:DFRequency</code></td>
<td><code>DeltaFrequency</code></td>
</tr>
<tr>
<td>IMD Setup...</td>
<td>(Swept IMD Measurement Class only)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMDX Setup...</td>
<td>(Swept IMD Converters Measurement Class only)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# CF_Freq Commands - IMS.IMSX

**Freq Commands - IM Spectrum and IM Spectrum Converters Measurement Classes**

Click [here](#) to view links to Freq commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fc</strong></td>
<td>SENSE:IMS:STIMulus:FCEnter</td>
<td>FrequencyCenter</td>
</tr>
<tr>
<td><strong>DeltaF</strong></td>
<td>SENSE:IMS:STIMulus:DFrequency</td>
<td>DeltaFrequency</td>
</tr>
<tr>
<td><strong>F1</strong></td>
<td>SENSE:IMS:STIMulus:F1FRequency</td>
<td>F1Frequency</td>
</tr>
<tr>
<td><strong>F2</strong></td>
<td>SENSE:IMS:STIMulus:F2FRequency</td>
<td>F2Frequency</td>
</tr>
<tr>
<td><strong>Start Spectrum</strong></td>
<td>SENSE:IMS:RESPonse:STARt</td>
<td>SpectrumStartFrequency</td>
</tr>
<tr>
<td><strong>Stop Spectrum</strong></td>
<td>SENSE:IMS:RESPonse:STOP</td>
<td>SpectrumStopFrequency</td>
</tr>
<tr>
<td><strong>Center Spectrum</strong></td>
<td>SENSE:IMS:RESPonse:CENTer</td>
<td>SpectrumCenterFrequency</td>
</tr>
<tr>
<td><strong>Span Spectrum</strong></td>
<td>SENSE:IMS:RESPonse:SPAN</td>
<td>SpectrumSpanFrequency</td>
</tr>
</tbody>
</table>
# CF_Freq Commands - MOD

**Freq Commands - Modulation Distortion and Modulation Distortion Converters Measurement Class**

Click [here](#) to view links to Freq commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td></td>
<td>SENSE: FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td></td>
<td>SENSE: FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>Center</td>
<td></td>
<td>SENSE: FREQuency:CENTer</td>
<td>CenterFrequency</td>
</tr>
<tr>
<td>Span</td>
<td></td>
<td>SENSE: FREQuency:SPAN</td>
<td>FrequencySpan</td>
</tr>
<tr>
<td>Span All Meas Bands</td>
<td></td>
<td>SENSE: DISTortion:FREQuency:TUNE:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>MOD Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MODX Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mod Source Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>Fixed Carrier</td>
</tr>
<tr>
<td>Start Carrier</td>
</tr>
<tr>
<td>Stop Carrier</td>
</tr>
<tr>
<td>Source Modulation</td>
</tr>
<tr>
<td>Pulse Setup</td>
</tr>
<tr>
<td>MOD Setup</td>
</tr>
<tr>
<td>MODX Setup</td>
</tr>
</tbody>
</table>
## CF_Freq Commands - NF

Freq Commands - Noise Figure Cold Source Measurement Class

Click [here](#) to view links to Freq commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Start</strong></td>
<td></td>
<td>SENSE:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td><strong>Stop</strong></td>
<td></td>
<td>SENSE:FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td><strong>Center</strong></td>
<td></td>
<td>SENSE:FREQuency:CENTer</td>
<td>CenterFrequency</td>
</tr>
<tr>
<td><strong>Span</strong></td>
<td></td>
<td>SENSE:FREQuency:SPAN</td>
<td>FrequencySpan</td>
</tr>
<tr>
<td><strong>Step</strong></td>
<td></td>
<td>SENSE:FREQuency:STEP</td>
<td>FrequencyStep</td>
</tr>
<tr>
<td><strong>Frequency</strong></td>
<td><strong>Offset...</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Mode</strong></td>
<td><strong>Coupled and Un-coupled</strong></td>
<td>SENSE:FOM:RANGe:COUPled</td>
<td>Coupled</td>
</tr>
<tr>
<td><strong>Sweep Type</strong></td>
<td></td>
<td>SENSE:FOM:RANGe:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td><strong>Settings</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Start</strong></td>
<td></td>
<td>SENSE:FOM:RANGe:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td></td>
<td>SENSE:FOM:RANGE:FREQuency:STOP</td>
<td>StopFrequency</td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------------------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>Annotation - Primary, Source, and Receivers</td>
<td>SENSE:FOM:RANGE:NAME?</td>
<td>Name</td>
<td></td>
</tr>
<tr>
<td>X-Axis Point Spacing</td>
<td>SENSE:FOM:RANGE:SEGment:SWEep:POINts</td>
<td>NumberOfPoints</td>
<td></td>
</tr>
<tr>
<td>NF Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CF_Freq Commands - PN
Freq Commands - Phase Noise Measurement Class

Click [here](#) to view links to Freq commands for all Measurement Classes.

## Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Offset</td>
<td></td>
<td><code>SENSe:FREQuency:STARt</code></td>
<td><code>StartFrequency</code></td>
</tr>
<tr>
<td>Stop Offset</td>
<td></td>
<td><code>SENSe:FREQuency:STOP</code></td>
<td><code>StopFrequency</code></td>
</tr>
<tr>
<td>Carrier</td>
<td></td>
<td><code>SENSe:PN:SWEep:CARRier:FREQuency</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td>Source Frequency...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phase Noise Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# CF_FREQ Commands - SA

Freq Commands - Spectrum Analyzer Measurement Class

Click [here](#) to view links to Freq commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Start</strong></td>
<td>SENSE:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td><strong>Stop</strong></td>
<td>SENSE:FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td><strong>Center</strong></td>
<td>SENSE:FREQuency:CENTer</td>
<td>CenterFrequency</td>
</tr>
<tr>
<td><strong>Span</strong></td>
<td>SENSE:FREQuency:SPAN</td>
<td>FrequencySpan</td>
</tr>
<tr>
<td><strong>Step</strong></td>
<td>SENSE:SWEep:STEP</td>
<td>FrequencyStep</td>
</tr>
<tr>
<td><strong>Full Span</strong></td>
<td>SENSE:FREQuency:SPAN:FULL</td>
<td>FrequencySpanFull</td>
</tr>
<tr>
<td><strong>CF Step</strong></td>
<td>SENSE:FREQuency:CENTer:STEP:SIZE</td>
<td>CenterFrequencyStepSize</td>
</tr>
<tr>
<td><strong>Auto Tune</strong></td>
<td>SENSE:SA:COHerence:DISTortion:FREQuency:TUNE:IMMediate</td>
<td>None</td>
</tr>
</tbody>
</table>
### Source Frequency Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI Command</th>
<th>COM Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1...</td>
<td></td>
<td>SENSe:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
</tr>
<tr>
<td>Port 2...</td>
<td></td>
<td>SENSe:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
</tr>
<tr>
<td>Port 3...</td>
<td></td>
<td>SENSe:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
</tr>
<tr>
<td>Port 4...</td>
<td></td>
<td>SENSe:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
</tr>
</tbody>
</table>

### Source IQ Mod.

**MXG** (this is the name assigned to the source)
# CF_Freq Commands - SMC_VMC

**Freq Commands - Scalar Mixer/Converter + Phase and Vector Mixer/Converter Measurement Classes**

Click [here](#) to view links to Freq commands for all Measurement Classes.

## Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start...</td>
<td><strong>Input</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Calculate Input and Output frequencies</td>
<td>SENSe:MIXer:CALCulate</td>
<td>Calculate</td>
<td></td>
</tr>
<tr>
<td>Input to swept or fixed</td>
<td>SENSe:MIXer:INPut:FREQuency:MODE</td>
<td>InputRangeMode</td>
<td></td>
</tr>
<tr>
<td>Input start frequency</td>
<td>SENSe:MIXer:INPut:FREQuency:STARt</td>
<td>InputStartFrequency</td>
<td></td>
</tr>
<tr>
<td>Input stop frequency</td>
<td>SENSe:MIXer:INPut:FREQuency:STOP</td>
<td>InputStopFrequency</td>
<td></td>
</tr>
<tr>
<td>Input power level</td>
<td>SENSe:MIXer:INPut:POWer</td>
<td>InputPower</td>
<td></td>
</tr>
<tr>
<td>Input fixed frequency</td>
<td>SENSE:MIXer:INPut:FREQuency:FIXed</td>
<td>InputFixedFrequency</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------</td>
<td>---------------------</td>
<td></td>
</tr>
<tr>
<td>LO1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LO freq fixed or swept</td>
<td>SENSE:MIXer:LO:FREQuency:MODE</td>
<td>LORangeMode</td>
<td></td>
</tr>
<tr>
<td>LO fixed frequency</td>
<td>SENSE:MIXer:LO:FREQuency:FIXed</td>
<td>LOFixedFrequency</td>
<td></td>
</tr>
<tr>
<td>LO start frequency</td>
<td>SENSE:MIXer:LO:FREQuency:STARt</td>
<td>LOStartFrequency</td>
<td></td>
</tr>
<tr>
<td>LO stop frequency</td>
<td>SENSE:MIXer:LO:FREQuency:STOP</td>
<td>LOStopFrequency</td>
<td></td>
</tr>
<tr>
<td>Input Greater / Less that LO</td>
<td>SENSE:MIXer:LO:FREQuency:ILTI</td>
<td>IsInputGreaterThanLO</td>
<td></td>
</tr>
<tr>
<td>Output</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sideband (high or low)</td>
<td>SENSE:MIXer:OUTPut:FREQuency:SIDeband</td>
<td>OutputSideband</td>
<td></td>
</tr>
<tr>
<td>Setting</td>
<td>Command</td>
<td>Output Field</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>--------------------------------------------</td>
<td>-----------------------</td>
<td></td>
</tr>
<tr>
<td>Output start frequency</td>
<td>SENSe:MIXer:OUTPut:FREQuency:STARt</td>
<td>OutputStartFrequency</td>
<td></td>
</tr>
<tr>
<td>Output stop frequency</td>
<td>SENSe:MIXer:OUTPut:FREQuency:STOP</td>
<td>OutputStopFrequency</td>
<td></td>
</tr>
<tr>
<td>Output to swept or fixed</td>
<td>SENSe:MIXer:OUTPut:FREQuency:MODE</td>
<td>OutputRangeMode</td>
<td></td>
</tr>
<tr>
<td>Output fixed frequency</td>
<td>SENSe:MIXer:OUTPut:FREQuency:FIXed</td>
<td>OutputFixedFrequency</td>
<td></td>
</tr>
<tr>
<td>Stop...</td>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LO1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Center...</td>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LO1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Span...</td>
<td>Input</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LO1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>-----</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CW...</td>
<td></td>
<td>Sweep</td>
<td></td>
</tr>
<tr>
<td>SMC Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Scalar Mixer/Converter + Phase Measurement Class only)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VMC Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Vector Mixer/Converter Measurement Class only)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# CF_Freq Commands - Standard

*Freq Commands - Standard Measurement Class*

Click [here](#) to view links to Freq commands for all Measurement Classes.

## Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td></td>
<td>SENSE: FREQuency: STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td></td>
<td>SENSE: FREQuency: STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>Center</td>
<td></td>
<td>SENSE: FREQuency: CENTER</td>
<td>CenterFrequency</td>
</tr>
<tr>
<td>Span</td>
<td></td>
<td>SENSE: FREQuency: SPAN</td>
<td>FrequencySpan</td>
</tr>
<tr>
<td>Step</td>
<td></td>
<td>SENSE: SWEep: STEP</td>
<td>FrequencyStep</td>
</tr>
<tr>
<td>CW</td>
<td></td>
<td>SENSE: FREQuency: CW</td>
<td>CWFrequency</td>
</tr>
<tr>
<td>Frequency Offset...</td>
<td>Frequency Offset (ON/OFF)</td>
<td>SENSE: FOM[:STATe]</td>
<td>State</td>
</tr>
<tr>
<td></td>
<td>Mode - Coupled and Un-coupled</td>
<td>SENSE: FOM: RANGe: COUPlled</td>
<td>Coupled</td>
</tr>
<tr>
<td></td>
<td>Sweep Type</td>
<td>SENSE: FOM: RANGe: SWEep: TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Settings</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>SENSE:FORM:RANGE:FREQuency:START</td>
<td>StartFrequency</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>----------------------------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>Stop</td>
<td>SENSE:FORM:RANGE:FREQuency:STOP</td>
<td>StopFrequency</td>
<td></td>
</tr>
<tr>
<td>Annotation - Primary, Source, and Receivers</td>
<td>SENSE:FORM:RANGE:NAME?</td>
<td>Name</td>
<td></td>
</tr>
<tr>
<td>X-Axis Point Spacing</td>
<td>SENSE:FORM:RANGE:SEGMENT:SWEEP:POINts</td>
<td>NumberOfPoints</td>
<td></td>
</tr>
</tbody>
</table>
CF_Hardkeys
# CF_Macro Commands

## Key Setup Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Macro Setup...</td>
<td>Execute Macro</td>
<td><strong>SYSTem:SHORtcut:EXECute</strong></td>
<td><strong>app.ExecuteShortcut</strong></td>
</tr>
<tr>
<td></td>
<td>Delete Macro</td>
<td><strong>SYSTem:SHORtcut:DELete</strong></td>
<td><strong>app.DeleteShortCut</strong></td>
</tr>
<tr>
<td></td>
<td>Write macro path, argument, and title</td>
<td><strong>SYSTem:SHORtcut:PATH</strong>&lt;br&gt;<strong>SYSTem:SHORtcut:ARGuments</strong></td>
<td><strong>app.PutShortcut</strong></td>
</tr>
<tr>
<td></td>
<td>Read macro path, argument, and title</td>
<td><strong>SYSTem:SHORtcut:TITLe</strong></td>
<td><strong>app.GetShortcut</strong></td>
</tr>
<tr>
<td>Clear Favorites</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
# CF_Markers Commands

## Marker 1-7 Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marker 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
<tr>
<td>Marker 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
<tr>
<td>Marker 3</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
<tr>
<td>Marker 5</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
<tr>
<td>Marker 6</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
<tr>
<td>Marker 7</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
</tbody>
</table>

## Marker 8-15 Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marker 8</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
<tr>
<td>Marker 9</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
<tr>
<td>Marker</td>
<td>On/Off</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>Marker 10</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
<tr>
<td>Marker 11</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
<tr>
<td>Marker 12</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
<tr>
<td>Marker 13</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
<tr>
<td>Marker 14</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
<tr>
<td>Marker 15</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>MarkerState</td>
</tr>
</tbody>
</table>

**Marker Setup Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delta</td>
<td>ON/OFF</td>
<td>CALCulate:MEASure:MARKer:DELTa</td>
<td>DeltaMarker</td>
</tr>
<tr>
<td>Discrete</td>
<td>ON/OFF</td>
<td>CALCulate:MEASure:MARKer:DISCrete</td>
<td>Interpolated</td>
</tr>
<tr>
<td>Type</td>
<td>NORMAL</td>
<td>CALCulate:MEASure:MARKer:TYPE</td>
<td>Type</td>
</tr>
<tr>
<td></td>
<td>FIXED</td>
<td>CALCulate:MEASure:MARKer:TYPE</td>
<td>Type</td>
</tr>
<tr>
<td>Format</td>
<td>Trace</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td></td>
<td>Default</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Format</td>
<td>CALCapulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>Log Phase</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>Lin Phase</td>
<td>CALCapulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>Re / Im</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>R+jX</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>G+jB</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>Log Magnitude</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>Linear Magnitude</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>Phase</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>SWR</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>Delay</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>Real</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>Imaginary</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>Kelvin</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>----------------------------------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td>Fahrenheit</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>Celsius</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>Format</td>
<td></td>
</tr>
<tr>
<td>Coupled</td>
<td>Off</td>
<td>CoupledMarkers</td>
<td></td>
</tr>
<tr>
<td>Channel</td>
<td>CALCulate:MEASure:MARKer:COUPling[:STATE]</td>
<td>CoupledMarkersMethod</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>CALCulate:MEASure:MARKer:COUPling:METHOD</td>
<td>CoupledMarkersMethod</td>
<td></td>
</tr>
<tr>
<td>Marker Display...</td>
<td>Marker Readout</td>
<td>MarkerReadout</td>
<td></td>
</tr>
<tr>
<td>Large Readout</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>MarkerReadoutSize</td>
<td></td>
</tr>
<tr>
<td>Readouts Per Trace</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>MarkerReadoutsPerTrace</td>
<td></td>
</tr>
<tr>
<td>Symbol - Triangle, Flag, and Line</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>MarkerSymbol</td>
<td></td>
</tr>
<tr>
<td>Decimal Places - Stimulus</td>
<td>CALCulate:MEASure:MARKer:FORMat</td>
<td>MarkerReadoutStimulusPlaces</td>
<td></td>
</tr>
<tr>
<td>and Response</td>
<td>DISPlay:WINDow:ANNote:MARKer:XPOSition</td>
<td>MarkerReadoutXPosition</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------</td>
<td>------------------------</td>
<td></td>
</tr>
<tr>
<td>Readout Position - X and Y</td>
<td>DISPlay:WINDow:ANNote:MARKer:YPOSition</td>
<td>MarkerReadoutYPosition</td>
<td></td>
</tr>
<tr>
<td>Marker Colors...</td>
<td>DISPlay:COLor:TRACe:MARKer</td>
<td>Markers</td>
<td></td>
</tr>
<tr>
<td>Marker Table</td>
<td>DISPlay:WINDow:TABLE</td>
<td>MarkerReadout</td>
<td></td>
</tr>
<tr>
<td>All Off</td>
<td>CALCulate:MEASure:MARKer:AOFF</td>
<td>DeleteAllMarkers</td>
<td></td>
</tr>
</tbody>
</table>

**Marker -> Functions Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marker -&gt; Start</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>SetStart</td>
<td></td>
</tr>
<tr>
<td>Marker -&gt; Stop</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>SetStop</td>
<td></td>
</tr>
<tr>
<td>Marker -&gt; Center</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>SetCenter</td>
<td></td>
</tr>
<tr>
<td>Marker -&gt; Span</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Marker -&gt; Ref Level</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>SetReferenceLevel</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------</td>
<td>------------------</td>
<td></td>
</tr>
<tr>
<td>Marker -&gt; Delay</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>SetElectricalDelay</td>
<td></td>
</tr>
<tr>
<td>Marker -&gt; CW Freq</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>SetCW</td>
<td></td>
</tr>
<tr>
<td>Marker -&gt; IM Spectrum (Swept IMD and Swept IMD Converters Measurement Classes only)</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Marker -&gt; SA</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>toSA</td>
<td></td>
</tr>
</tbody>
</table>

**SA Analysis Tab Commands**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Commands</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CALCulate:SA:MARKer:BDENSITY:DATA?</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:MEASure:SA:MARKer:BDENSITY:NOISE[:STATE]</td>
<td></td>
</tr>
<tr>
<td>Band Span</td>
<td>CALCulate:MEASure:SA:MARKer:BNOise:SPAN</td>
<td>BandnoiseSpan</td>
</tr>
<tr>
<td>Band Span</td>
<td>CALCulate:MEASure:SA:MARKer:BPOwer:SPAN</td>
<td>BandpowerSpan</td>
</tr>
<tr>
<td>Equivalent Span</td>
<td>CALCulate:MEASure:SA:MARKer:BDENsity:EQSPan</td>
<td>BandDensityEQSPan</td>
</tr>
<tr>
<td>Equivalent Span</td>
<td>CALCulate:MEASure:SA:MARKer:BDENsity:EQSPan</td>
<td>BandDensityEQSPan</td>
</tr>
<tr>
<td>Density BW</td>
<td>CALCulate:SA:MARKer:BDENsity:BW</td>
<td>BandDensityBW</td>
</tr>
<tr>
<td>Occupied BW</td>
<td>CALCulate:MEASure:SA:MARKer:OCCBand:[STATE]</td>
<td>OccupiedBandState</td>
</tr>
<tr>
<td>Occupied BW</td>
<td>CALCulate:MEASure:SA:MARKer:OCCBand:CENTer?</td>
<td>OccupiedBandCenter</td>
</tr>
<tr>
<td>Occupied BW</td>
<td>CALCulate:MEASure:SA:MARKer:OCCBand:POWer?</td>
<td>OccupiedBandPowerdBm</td>
</tr>
<tr>
<td>Occupied BW</td>
<td>CALCulate:MEASure:SA:MARKer:OCCBand:SPAN</td>
<td>OccupiedBandSpan</td>
</tr>
<tr>
<td>Band Span</td>
<td>CALCulate:MEASure:SA:MARKer:BAND:IBW</td>
<td>None</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Off</td>
<td>CALCulate:MEASure:SA:MARKer:BAND:FUNCTION</td>
<td>None</td>
</tr>
<tr>
<td>Band Density</td>
<td>CALCulate:MEASure:SA:MARKer:BAND:FUNCTION</td>
<td>None</td>
</tr>
<tr>
<td>Band Power</td>
<td>CALCulate:MEASure:SA:MARKer:BAND:FUNCTION</td>
<td>None</td>
</tr>
<tr>
<td>Marker Noise</td>
<td>CALCulate:MEASure:SA:MARKer:BAND:FUNCTION</td>
<td>None</td>
</tr>
<tr>
<td>OBW Percent</td>
<td>CALCulate:MEASure:SA:MARKer:OCCBand:PERCent</td>
<td>OccupiedBandPercent</td>
</tr>
<tr>
<td></td>
<td>CALCulate:MEASure:SA:MARKer:OCCBand:SPAN?</td>
<td></td>
</tr>
</tbody>
</table>
# CF_Math Commands

## Memory Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data -&gt; Memory</td>
<td></td>
<td>CALCulate:MEASure:MATH:MEMorize</td>
<td>DataToMemory</td>
</tr>
<tr>
<td>Normalize</td>
<td></td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Data Math</td>
<td>Off</td>
<td>CALCulate:MEASure:MATH:FUNCtion</td>
<td>TraceMath</td>
</tr>
<tr>
<td>Data / Memory</td>
<td></td>
<td>CALCulate:MEASure:MATH:FUNCtion</td>
<td>TraceMath</td>
</tr>
<tr>
<td>Data * Memory</td>
<td></td>
<td>CALCulate:MEASure:MATH:FUNCtion</td>
<td>TraceMath</td>
</tr>
<tr>
<td>Data - Memory</td>
<td></td>
<td>CALCulate:MEASure:MATH:FUNCtion</td>
<td>TraceMath</td>
</tr>
<tr>
<td>Data + Memory</td>
<td></td>
<td>CALCulate:MEASure:MATH:FUNCtion</td>
<td>TraceMath</td>
</tr>
<tr>
<td>Display</td>
<td>Data Trace</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td></td>
<td>Memory Trace</td>
<td>DISPlay:WINDow:TRACe:MEMory</td>
<td>View</td>
</tr>
<tr>
<td>Data and Memory</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DISPlay:WINDow:TRACe:MEMory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>View</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8510 Mode</td>
<td>ON/OFF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>None</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interpolate</td>
<td>ON/OFF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALCulate:MEASure:MATH:INTERpolate[:STATe]</td>
<td>InterpolateMemoryIsDefault</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analysis Tab Commands</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCPI</td>
<td>COM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conversions</td>
<td>Off</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALCulate:MEASure:CONVersion:FUNCTION</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Z-Reflect</td>
<td>CALCulate:MEASure:CONVersion:FUNCTION</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Z-Transmit</td>
<td>CALCulate:MEASure:CONVersion:FUNCTION</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Z-Trans-Shunt</td>
<td>CALCulate:MEASure:CONVersion:FUNCTION</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Y-Reflect</td>
<td>CALCulate:MEASure:CONVersion:FUNCTION</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Y-Transmit</td>
<td>CALCulate:MEASure:CONVersion:FUNCTION</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Y-Trans-Shunt</td>
<td>CALCulate:MEASure:CONVersion:FUNCTION</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Equation Editor...</td>
<td>1 / S</td>
<td>CALCulate:MEASure:CONVersion:FUNCtion</td>
<td>None</td>
</tr>
<tr>
<td>--------------------</td>
<td>------</td>
<td>----------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td></td>
<td>Conjugation</td>
<td>CALCulate:MEASure:CONVersion:FUNCtion</td>
<td>None</td>
</tr>
<tr>
<td>Equation</td>
<td>On/Off</td>
<td>CALCulate:MEASure:EQUation[:STATe]</td>
<td>State</td>
</tr>
<tr>
<td>Equation</td>
<td>Equation</td>
<td>CALCulate:MEASure:EQUation:TEXT</td>
<td>Text</td>
</tr>
<tr>
<td>Enable Equation</td>
<td>Enable Equation</td>
<td>CALCulate:MEASure:EQUation[:STATe]</td>
<td>State</td>
</tr>
<tr>
<td>Functions</td>
<td>Functions</td>
<td>CALCulate:EQUation:LIBRary:FUNCtions</td>
<td>GetLibraryFunctions</td>
</tr>
<tr>
<td>Trace Data</td>
<td>Trace Data</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Parameter</td>
<td>Parameter</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Store Equation</td>
<td>Store Equation</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Delete Equation</td>
<td>Delete Equation</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Use Short Names</td>
<td>Use Short Names</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Fast Processing</td>
<td>Fast Processing</td>
<td>CALCulate:MEASure:EQUation:FAST</td>
<td>FastProcessing</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Command Function</td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------------------------------</td>
<td>------------------------</td>
<td></td>
</tr>
<tr>
<td>Enable Matlab</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Import...</td>
<td>Import Library</td>
<td>ImportLibrary</td>
<td></td>
</tr>
<tr>
<td>Import Library</td>
<td>CALCulate:EQUation:LIBRary:IMPort</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Remove Library</td>
<td>CALCulate:EQUation:LIBRary:REMove</td>
<td>RemoveLibrary</td>
<td></td>
</tr>
<tr>
<td>Remove Library</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Statistics...</td>
<td>On/Off</td>
<td>ShowStatistics</td>
<td></td>
</tr>
<tr>
<td>Statistics - Mean,</td>
<td>CALCulate:MEASure:FUNCtion:STATistics[:STATe]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standard Deviation,</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peak to Peak</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User 1 to 16</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>CALCulate:MEASure:FUNCtion:DOMain:USER:STARt</td>
<td>UserRangeMin</td>
<td></td>
</tr>
<tr>
<td>AM Distortion</td>
<td>AM Distortion</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td>CALCulate:MEASure:DISTortion:MODE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>AM-PM</td>
<td>CALCulate:MEASure:DISTortion:MODE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>AM-AM</td>
<td>CALCulate:MEASure:DISTortion:MODE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Y-axis = Slope Calculated Over Aperture</td>
<td>CALCulate:MEASure:DISTortion:SLOPe[:STATe]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:MEASure:DISTortion:SLOPe:APERture</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>X-axis = Backoff For Compression Level</td>
<td>CALCulate:MEASure:DISTortion:BACKoff[:STATe]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:MEASure:DISTortion:BACKoff:COMPression</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Trace Deviation</td>
<td>Off</td>
<td>CALCulate:MEASure:COMPutation:DEViation</td>
<td>TraceDeviationType</td>
</tr>
<tr>
<td></td>
<td>Linear</td>
<td>CALCulate:MEASure:COMPutation:DEViation</td>
<td>TraceDeviationType</td>
</tr>
<tr>
<td></td>
<td>Parabolic</td>
<td>CALCulate:MEASure:COMPutation:DEViation</td>
<td>TraceDeviationType</td>
</tr>
<tr>
<td></td>
<td>Cubic</td>
<td>CALCulate:MEASure:COMPutation:DEViation</td>
<td>TraceDeviationType</td>
</tr>
<tr>
<td>Trace</td>
<td>CALCulate:MEASure:UNCertainty:DISPlay:CFACtor</td>
<td>CoverageFactor</td>
<td></td>
</tr>
<tr>
<td>Uncertainty Analysis...</td>
<td>Trace Type</td>
<td>CALCulate:MEASure:UNCertainty:DISPlay:TYPE</td>
<td>DisplayType</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------</td>
<td>-------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Settings</td>
<td>CALCulate:MEASure:UNCertainty:MODE:NOISe</td>
<td>MeasurementNoiseUncertainty</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CALCulate:MEASure:UNCertainty:MODE:CABLE:REPeat</td>
<td>CableRepeatabilityUncertainty</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CALCulate:MEASure:UNCertainty:MODE:ETERm</td>
<td>ErrorTermUncertainty</td>
</tr>
<tr>
<td></td>
<td>Add Trace</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Apply to all traces</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Limits...</td>
<td>Limit</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Limit Test</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Limit Test ON</td>
<td>CALCulate:MEASure:LIMit[:STATE]</td>
<td>State</td>
</tr>
<tr>
<td></td>
<td>Limit Line ON</td>
<td>CALCulate:MEASure:LIMit:DISPlay[:STATE]</td>
<td>LineDisplay</td>
</tr>
<tr>
<td></td>
<td>Sound ON Fail</td>
<td>CALCulate:MEASure:LIMit:SOUNd[:STATE]</td>
<td>SoundOnFail</td>
</tr>
<tr>
<td></td>
<td>Pass/Fail Position</td>
<td>(DISPlay:WINDow:ANNotation:LIMit:XPOSition</td>
<td>LimitTestXPosition</td>
</tr>
<tr>
<td></td>
<td>X Pass/Fail Position</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Command/Function</td>
<td>Value</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>Y Pass/Fail Position</td>
<td>DISPLAY:WINDOW:ANNOTATION:LIMIT:YPOSITION</td>
<td>LimitTestYPosition</td>
<td></td>
</tr>
<tr>
<td>Limit Table</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Show Table</td>
<td>DISPLAY:WINDOW:TABLE</td>
<td>ShowTable</td>
<td></td>
</tr>
<tr>
<td>Load Table</td>
<td>MMEMORY:LOAD:LIMIT</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Save Table</td>
<td>MMEMORY:STORE:LIMIT</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Global Pass/Fail</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Global Pass/Fail ON</td>
<td>DISPLAY:FSIGN</td>
<td>DisplayGlobalPassFail</td>
<td></td>
</tr>
<tr>
<td>Ripple</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ripple Test</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ripple Test ON</td>
<td>CALCULATE:MEASURE:RLIMIT:STATE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Ripple Line ON</td>
<td>CALCULATE:MEASURE:RLIMIT:DISPLAY:LINE:STATE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sound ON Fail</td>
<td>CALCulate:MEASure:LI$\text{M}i$t$:SOUNd$:S$TATe</td>
<td>SoundOnFail</td>
<td></td>
</tr>
<tr>
<td>Ripple Result</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type - Off, Absolute or Margin</td>
<td>CALCulate:MEASure:RLIMit:DISPlay:TYPE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Segment</td>
<td>CALCulate:MEASure:RLIMit:DATA</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Ripple Table</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Show Table</td>
<td>DISPlay:WINDow:TABLE</td>
<td>ShowTable</td>
<td></td>
</tr>
<tr>
<td>Load Table</td>
<td>MMEMory:LOAD:RLIMit</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Save Table</td>
<td>MMEMory:STORe:RLIMit</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Bandwidth</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth Test</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth Test ON</td>
<td>CALCulate:MEASure:BLIMit$:S$TATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Value</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------------------------</td>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>Bandwidth Marker ON</td>
<td>CALCulate:MEASure:BLIMit:DISPlay:MARKer:STATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sound ON Fail</td>
<td>CALCulate:MEASure:LIImit:SOUNd[:STATe]</td>
<td>SoundOnFail</td>
<td></td>
</tr>
<tr>
<td>N dB Points</td>
<td>CALCulate:MEASure:BLIMit:BWIDth:THreshold</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Min Bandwidth</td>
<td>CALCulate:MEASure:BLIMit:MINimum</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Max Bandwidth</td>
<td>CALCulate:MEASure:BLIMit:MAXimum</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Limit Table</td>
<td>DISPlay:WIN Dow:TABLEl</td>
<td>ShowTable</td>
<td></td>
</tr>
<tr>
<td>Limit</td>
<td>DISPlay:WIN Dow:TABLEl</td>
<td>ShowTable</td>
<td></td>
</tr>
<tr>
<td>Ripple</td>
<td>DISPlay:WIN Dow:TABLEl</td>
<td>ShowTable</td>
<td></td>
</tr>
<tr>
<td>Compression Analysis... (Gain Compression and Gain Compression Converters Measuremen)</td>
<td>CALCulate:MEASure:GCMeas:ANALysis:ENABle</td>
<td>AnalysisEnable</td>
<td></td>
</tr>
<tr>
<td>Analysis Frequency</td>
<td>CALCulate:MEASure:GCMeas:ANALysis:CWFrequency</td>
<td>AnalysisCWFreq</td>
<td></td>
</tr>
<tr>
<td>Classes</td>
<td>Use Measured Pin</td>
<td>Calculation: MEASURE: GCMeas: ANALysis: XAXis</td>
<td>AnalysisXAxis</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------------</td>
<td>---------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td></td>
<td>Use Source Pwr Settings</td>
<td>Calculation: MEASURE: GCMeas: ANALysis: XAXis</td>
<td>AnalysisXAxis</td>
</tr>
</tbody>
</table>

### Time Domain Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transform</td>
<td>ON/OFF</td>
<td>Calculation: MEASURE: TRANsform: TIME: STATE</td>
<td>State</td>
</tr>
<tr>
<td>Start Time</td>
<td></td>
<td>Calculation: MEASURE: TRANsform: TIME: START</td>
<td>Start</td>
</tr>
<tr>
<td>Stop Time</td>
<td></td>
<td>Calculation: MEASURE: TRANsform: TIME: STOP</td>
<td>Stop</td>
</tr>
<tr>
<td>Center Time</td>
<td></td>
<td>Calculation: MEASURE: TRANsform: TIME: CENTER</td>
<td>Center</td>
</tr>
<tr>
<td>Span Time</td>
<td></td>
<td>Calculation: MEASURE: TRANsform: TIME: SPAN</td>
<td>Span</td>
</tr>
<tr>
<td>TD Toolbar</td>
<td>ON/OFF</td>
<td>DISPlay:TOOLbar:TRANsform[:STATe]</td>
<td>app.ShowToolbar</td>
</tr>
<tr>
<td>------------</td>
<td>--------</td>
<td>-----------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><strong>Time Domain Setup...</strong></td>
<td><strong>Transform</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Transform On</strong></td>
<td>CALCulate:MEASure:TRANSform:TIME:STATe</td>
<td>State</td>
</tr>
<tr>
<td></td>
<td><strong>Start</strong></td>
<td>CALCulate:MEASure:TRANSform:TIME:STARt</td>
<td>Start</td>
</tr>
<tr>
<td></td>
<td><strong>Stop</strong></td>
<td>CALCulate:MEASure:TRANSform:TIME:STOP</td>
<td>Stop</td>
</tr>
<tr>
<td></td>
<td><strong>Center</strong></td>
<td>CALCulate:MEASure:TRANSform:TIME:CENTer</td>
<td>Center</td>
</tr>
<tr>
<td></td>
<td><strong>Span</strong></td>
<td>CALCulate:MEASure:TRANSform:TIME:SPAN</td>
<td>Span</td>
</tr>
<tr>
<td></td>
<td><strong>Set Low Pass Frequencies</strong></td>
<td>CALCulate:MEASure:TRANSform:TIME:LPFREQuency</td>
<td>SetFrequencyLowPass</td>
</tr>
<tr>
<td></td>
<td><strong>Gating</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------------------------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>Gating On</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STATe</td>
<td>State</td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STARt</td>
<td>Start</td>
<td></td>
</tr>
<tr>
<td>Stop</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STOP</td>
<td>Stop</td>
<td></td>
</tr>
<tr>
<td>Center</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:CENTer</td>
<td>Center</td>
<td></td>
</tr>
<tr>
<td>Span</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:SPAN</td>
<td>Span</td>
<td></td>
</tr>
<tr>
<td>Gate Type</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME[:TYPE]</td>
<td>Type</td>
<td></td>
</tr>
<tr>
<td>Gate Shape</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:SHApe</td>
<td>Shape</td>
<td></td>
</tr>
<tr>
<td>Coupling</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Coupling On</td>
<td>SENSE:COUPle:PARameter[:STATe]</td>
<td>CoupleChannelParams</td>
<td></td>
</tr>
<tr>
<td>Transform Parameters</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stimulus</td>
<td>CALCulate:MEASure:TRANsform:COUple:PARameters</td>
<td>CoupledParameters</td>
<td></td>
</tr>
<tr>
<td>State (On/Off)</td>
<td>CALCulate:MEASure:TRANsform:COUple:PARameters</td>
<td>CoupledParameters</td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>CALCulate:MEASure:TRANsform:COUple:PARameters</td>
<td>CoupledParameters</td>
<td></td>
</tr>
<tr>
<td>Mode</td>
<td>CALCulate:MEASure:TRANSform:COUPlE:PARameters</td>
<td>CoupledParameters</td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------</td>
<td>-------------------</td>
<td></td>
</tr>
<tr>
<td>Distance Marker Unit</td>
<td>CALCulate:MEASure:TRANSform:COUPlE:PARameters</td>
<td>CoupledParameters</td>
<td></td>
</tr>
<tr>
<td>Gating Parameters</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stimulus</td>
<td>CALCulate:MEASure:FILTER[:GATE]:COUPlE:PARameters</td>
<td>CoupledParameters (Gating)</td>
<td></td>
</tr>
<tr>
<td>State (On/Off)</td>
<td>CALCulate:MEASure:FILTER[:GATE]:COUPlE:PARameters</td>
<td>CoupledParameters (Gating)</td>
<td></td>
</tr>
<tr>
<td>Shape</td>
<td>CALCulate:MEASure:FILTER[:GATE]:COUPlE:PARameters</td>
<td>CoupledParameters (Gating)</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>CALCulate:MEASure:FILTER[:GATE]:COUPlE:PARameters</td>
<td>CoupledParameters (Gating)</td>
<td></td>
</tr>
<tr>
<td>Marker</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Auto</td>
<td>CALCulate:MEASure:TRANSform:TIME:MARKer:MODE</td>
<td>DistanceMarkerMode</td>
<td></td>
</tr>
<tr>
<td>Reflection (divide by 2)</td>
<td>CALCulate:MEASure:TRANSform:TIME:MARKer:MODE</td>
<td>DistanceMarkerMode</td>
<td></td>
</tr>
<tr>
<td>Transmission</td>
<td>CALCulate:MEASure:TRANSform:TIME:MARKer:MODE</td>
<td>DistanceMarkerMode</td>
<td></td>
</tr>
<tr>
<td>Meters (m)</td>
<td>CALCulate:MEASure:TRANSform:TIME:MARKer:UNIT</td>
<td>DistanceMarkerUnit</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command Sequence</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------------------------------------------------</td>
<td>-------------------</td>
<td></td>
</tr>
<tr>
<td>Feet (ft)</td>
<td>CALCulate:MEASure:TRANsform:TIME:MARKer:UNIT</td>
<td>DistanceMarkerUnit</td>
<td></td>
</tr>
<tr>
<td>Inches (in)</td>
<td>CALCulate:MEASure:TRANsform:TIME:MARKer:UNIT</td>
<td>DistanceMarkerUnit</td>
<td></td>
</tr>
<tr>
<td>Velocity Factor</td>
<td>SENSE:CORRection:EXTension:PORT:VELFactor</td>
<td>VelocityFactor</td>
<td></td>
</tr>
<tr>
<td>Advanced</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Impulse Width</td>
<td>CALCulate:MEASure:TRANsform:TIME:IMPulse:WIDTH</td>
<td>ImpulseWidth</td>
<td></td>
</tr>
<tr>
<td>Alignment Type</td>
<td>CALCulate:MEASure:TRANsform:TIME:ALIGNment</td>
<td>Alignment</td>
<td></td>
</tr>
<tr>
<td>Limit Start/Stop Times to</td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Avoid Aliasing</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Time Gating Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gating</td>
<td>ON/OFF</td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME:STATe</td>
<td>State</td>
</tr>
<tr>
<td>Gate Start</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STARt</td>
<td>Start</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>Gate Stop</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STOP</td>
<td>Stop</td>
<td></td>
</tr>
<tr>
<td>Gate Center</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:CENTer</td>
<td>Center</td>
<td></td>
</tr>
<tr>
<td>Gate Span</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:SPAN</td>
<td>Span</td>
<td></td>
</tr>
<tr>
<td>Gate Type</td>
<td>Band Pass</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME[:TYPE]</td>
<td>Type</td>
</tr>
<tr>
<td>Notch</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME[:TYPE]</td>
<td>Type</td>
<td></td>
</tr>
<tr>
<td>Gate Shape</td>
<td>Maximum</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:SHApe</td>
<td>Shape</td>
</tr>
<tr>
<td></td>
<td>Wide</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:SHApe</td>
<td>Shape</td>
</tr>
<tr>
<td></td>
<td>Normal</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:SHApe</td>
<td>Shape</td>
</tr>
<tr>
<td></td>
<td>Minimum</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:SHApe</td>
<td>Shape</td>
</tr>
<tr>
<td>Gating Setup...</td>
<td>Gating</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gating On</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STATe</td>
<td>State</td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:START</td>
<td>Start</td>
<td></td>
</tr>
<tr>
<td>Stop</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STOP</td>
<td>Stop</td>
<td></td>
</tr>
<tr>
<td>Center</td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME:CENTer</td>
<td>Center</td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>Span</td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME:SPAN</td>
<td>Span</td>
<td></td>
</tr>
<tr>
<td>Gate Type</td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME[:TYPE]</td>
<td>Type</td>
<td></td>
</tr>
<tr>
<td>Gate Shape</td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME:SHAPe</td>
<td>Shape</td>
<td></td>
</tr>
</tbody>
</table>
CF_Meas Commands

The Meas softkeys vary depending on which measurement class is currently active. Click on a measurement class link below to view the corresponding softkeys.

Standard

Active Hot Parameters

Gain Compression

Differential I/Q

IM Spectrum

Swept IMD and Swept IMD Converters

Modulation Distortion and Modulation Distortion Converters

Noise Figure Cold Source

Phase Noise

Gain Compression Converters

IM Spectrum Converters

Noise Figure Converters

Scalar Mixer/Converter + Phase

Vector Mixer/Converter

Spectrum Analyzer

TDR (Time Domain Reflectometry)
CF_Meas Commands - DIQ

Meas Commands - Differential I/Q Measurement Class

Click [here](#) to view links to Meas commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Meas Tab Commands</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
</tr>
<tr>
<td>IPwrF1</td>
<td>On/Off</td>
<td>C:CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>OPwrF1</td>
<td>On/Off</td>
<td>C:CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>GainF1</td>
<td>On/Off</td>
<td>C:CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Other...</td>
<td>IPwrF1 = a1_F1</td>
<td>C:CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td></td>
<td>OPwrF1 = b2_F1</td>
<td>C:CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td></td>
<td>GainF1 = b2_F</td>
<td>C:CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Meas Class...</td>
<td></td>
<td>C:CALCulate:MEASure:DEFine</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Meas Setup Tab Commands</th>
<th></th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conversions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Correction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel Correction On</td>
<td>SENSE:CORRection[:STATe]</td>
<td></td>
</tr>
<tr>
<td>Channel Correction Off</td>
<td>SENSE:CORRection[:STATe]</td>
<td></td>
</tr>
<tr>
<td>Cal Set...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smart Cal...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equation Editor...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time Domain...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## CF_Meas Commands - GCA

**Meas Commands - Gain Compression Measurement Class**

Click [here](#) to view links to Meas commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CompIn21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>CompOut21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>DeltaGain21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>CompGain21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>CompS11</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Ref21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td>S21, S11, S12, S22, AI1, AI2, CompIn21, CompOut21, DeltaGain21, CompGain21, CompS11, Ref21, CompAI1, CompAI2</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
</tbody>
</table>
### Meas Class...

- CALCulate:MEASure:DEFine
- CreateCustomMeasurementEx

### S-Param Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>S11</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
</tr>
<tr>
<td>S21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
</tr>
<tr>
<td>S12</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
</tr>
<tr>
<td>S22</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
</tr>
<tr>
<td>Other...</td>
<td>S21, S11, S12, S22, AI1, AI2, CompIn21, CompOut21, DeltaGain21, CompGain21, CompS11, Ref21, CompAI1, CompAI2</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
</tr>
</tbody>
</table>

### Auxiliary Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuxIn1 Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>AuxIn2 Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>------</td>
<td>-----</td>
</tr>
<tr>
<td>Meas Setup Tab Commands</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conversions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td>Channel Correction On</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td>Channel Correction Off</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
<td></td>
</tr>
<tr>
<td>Cal Set...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smart Cal...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equation Editor...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>Memory...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time Domain...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## CF_Meas Commands - GCX

**Meas Commands - Gain Compression Converters Measurement Class**

Click [here](#) to view links to Meas commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Compression Tab Commands</th>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>ComplIn21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>ComplOut21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>DeltaGain21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>CompGain21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>CompS11</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>Ref21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>Other...</td>
<td>SC21 Linear Gain</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S11 Linear Input Match</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SC12 Linear Reverse Isolation</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>-----------------------------------</td>
<td>--------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S22 Linear Output Match</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AI1 Linear AI1</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AI2 Linear AI2</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CompIn21 Incident Power at Compression</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CompOut21 Output Power at Compression</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DeltaGain21 Gain Deviation at Compression</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CompGain21 Gain at Compression</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CompS11 Input Match at Compression</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RefS21 Reference Gain for Compression</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CompAI1 AI1 at Compression</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CompAI2 AI2 at Compression</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>-----------------------------------------</td>
<td>------------------------------------</td>
<td></td>
</tr>
<tr>
<td>S11</td>
<td>On/Off</td>
<td>CALCulate:MEasure:PARameter</td>
<td>CreateS-Parameter</td>
<td></td>
</tr>
<tr>
<td>SC21</td>
<td>On/Off</td>
<td>CALCulate:MEasure:PARameter</td>
<td>CreateS-Parameter</td>
<td></td>
</tr>
<tr>
<td>SC12</td>
<td>On/Off</td>
<td>CALCulate:MEasure:PARameter</td>
<td>CreateS-Parameter</td>
<td></td>
</tr>
<tr>
<td>S22</td>
<td>On/Off</td>
<td>CALCulate:MEasure:PARameter</td>
<td>CreateS-Parameter</td>
<td></td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Meas Class...</td>
<td></td>
<td>CALCulate:MEasure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
</tbody>
</table>
### Power Tab Commands

| Command | Status | SCPI Code | Creation
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>OPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>RevIPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>RevOPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
</tbody>
</table>

### Auxiliary Tab Commands

| Command | Status | SCPI Code | Creation
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>AuxIn1 Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>AuxIn2 Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>CompAI1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>CompAI2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------</td>
<td>---------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Conversions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td>Channel Correction On</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td></td>
<td>Channel Correction Off</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td></td>
<td>Cal Set...</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Smart Cal...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equation Editor...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time Domain...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**CF_Meas Commands - HotS22**

**Meas Commands - Active Hot Parameters**

Click [here](#) to view links to Meas commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>HotS11</td>
<td>On/Off</td>
<td><code>CALCulate:MEASure:DEFine</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td>HotS31</td>
<td>On/Off</td>
<td><code>CALCulate:MEASure:DEFine</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td>HotS13</td>
<td>On/Off</td>
<td><code>CALCulate:MEASure:DEFine</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td>HotS33</td>
<td>On/Off</td>
<td><code>CALCulate:MEASure:DEFine</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td>Other...</td>
<td>IPwr</td>
<td><code>CALCulate:MEASure:DEFine</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td></td>
<td>OPwr</td>
<td><code>CALCulate:MEASure:DEFine</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td></td>
<td>Gamma</td>
<td><code>CALCulate:MEASure:DEFine</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td></td>
<td>Pmax</td>
<td><code>CALCulate:MEASure:DEFine</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td></td>
<td>Xs(3,3)</td>
<td><code>CALCulate:MEASure:DEFine</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td>X(t,3)</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>--------------------------</td>
<td>---------------------------</td>
<td></td>
</tr>
<tr>
<td>Xf(3,1)</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>DeltaOPwr</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
</tbody>
</table>

### Meas Setup Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conversions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td>Channel Correction On</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td></td>
<td>Channel Correction Off</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td>Cal Set...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smart Cal...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equation Editor...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Time Domain...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Click [here](#) to view links to Meas commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main Tone Power</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>TonePower</td>
</tr>
<tr>
<td>IM3</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>OIP3</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>IIP3</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td>PwrMain</td>
<td>CALCulate:MEASure:PARameter</td>
<td>TonePower</td>
</tr>
<tr>
<td></td>
<td>IM3</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td>OIP3</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td>IIP3</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td>Pwr2</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------</td>
<td>---------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Meas Setup Tab Commands</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Softkey</strong></td>
<td><strong>Sub-item</strong></td>
<td><strong>SCPI</strong></td>
<td><strong>COM</strong></td>
</tr>
<tr>
<td>Conversions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td>Channel Correction On</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td></td>
<td>Channel Correction Off</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td>Cal Set...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smart Cal...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equation Editor...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time Domain...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**CF_Meas Commands - IMS**

**Meas Commands - IM Spectrum Measurement Class**

Click [here](#) to view links to Meas commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Spectrum Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td>Output Spectrum</td>
</tr>
<tr>
<td>Input Spectrum</td>
</tr>
<tr>
<td>Reflected Spectrum</td>
</tr>
<tr>
<td>Other...</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Meas Class...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Meas Setup Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
</tbody>
</table>

7443
<table>
<thead>
<tr>
<th>Conversions</th>
<th>Correction</th>
<th>Channel Correction On</th>
<th>SENSe:CORRection[:STATe]</th>
<th>ErrorCorrection</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Channel Correction Off</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cal Set...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cal All...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equation Editor...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time Domain...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## CF_Meas Commands - IMSX

**Meas Commands - IM Spectrum Converters Measurement Class**

Click [here](#) to view links to Meas commands for all Measurement Classes.

### Spectrum Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output Spectrum</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td>Output Port 2, Incident</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Meas Class...</td>
<td></td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
</tbody>
</table>

### Meas Setup Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conversions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td>Channel Correction On</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td></td>
<td>Channel Correction Off</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td></td>
<td>Cal Set...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Menu Item</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cal All...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equation Editor...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time Domain...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# CF_Meas Commands - MOD

**Meas Commands - Modulation Distortion and Modulation Distortion Converters Measurement Class**

Click [here](#) to view links to Meas commands for all Measurement Classes.

## Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>POut2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Pln1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>POut1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>MSig2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Distortion Table...</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SENSe:DISTortion:TABLE:DISPlay:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSe:DISTortion:TABLE:DISPlay:DELeTe</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSe:DISTortion:TABLE:DISPlay:FEED</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSe:DISTortion:TABLE:DISPlay:FONT</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSe:DISTortion:TABLE:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSe:DISTortion:TABLE:DATA:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSe:DISTortion:TABLE:DATA:VALue?</td>
<td>None</td>
</tr>
</tbody>
</table>
| MOD Setup... |  | SENSE:DISTortion:TABLE:DISPLAY:SAVE  
SENSe:DISTortion:TABLE:DISPLAY:SORT |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MODX Setup...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| Meas Class... |  | CALCulate:MEASure:DEFine       
CreateCustomMeasurementEx |

**Auxiliary Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuxIn1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Source Port 1/2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AuxIn2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Source Port 1/2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
</tbody>
</table>

**Meas Setup Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conversions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td>Channel Correction</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td>------------</td>
<td>--------------------</td>
<td>---------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>On</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Select Cal Set... |
| Cal All...       |
| Src Mod Cal...   |

Trace Hold

Equation Editor...

Memory...

Time Domain...

Pulse Setup...
## CF_Meas Commands - NF

**Meas Commands - Noise Figure Cold Source Measurement Class**

Click [here](#) to view links to Meas commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Noise</th>
<th>Incident</th>
<th>Available</th>
<th>Noise Correlation</th>
<th>S-Param</th>
<th>Receivers</th>
<th>Meas Setup</th>
</tr>
</thead>
</table>

### Noise Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>NF</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>T-Eff</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>ENR</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>NFmin</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>GammaOpt</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Rn</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td>NF Noise Figure</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td>T-Eff Noise Temperature</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td>ENR Excess Noise Ratio</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Measurement</td>
<td>Description</td>
<td>Command</td>
<td>Function</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>---------</td>
<td>----------</td>
</tr>
<tr>
<td>DUTRNP</td>
<td>Available DUT Relative Noise Power</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>SYSRNP</td>
<td>Available System Relative Noise Power</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>DUTNPD</td>
<td>Available DUT Noise Power Density</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>SYSNPD</td>
<td>Available System Noise Power Density</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>DUTRNPI</td>
<td>Incident DUT Relative Noise Power</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>SYSRNPI</td>
<td>Incident System Relative Noise Power</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>DUTNPDI</td>
<td>Incident DUT Noise Power Density</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>SYSNPDI</td>
<td>Incident System Noise Power Density</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>S11, S21, S12, S22</td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>R1_1</td>
<td>Input pwr at DUT-IN @ IN freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Measurement</td>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------------</td>
<td>---------------------------------</td>
<td></td>
</tr>
<tr>
<td>R2_2 Input pwr at DUT-OUT @ OUT freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>A_1 Out pwr at DUT-IN @ IN freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>A_2 Out pwr at DUT-IN @ IN freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>B_1 Out Pwr at DUT-OUT @ OUT freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>B_2 Out Pwr at DUT-OUT @ OUT freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>NFmin Minimum Noise Figure</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>GammOpt Optimum Complex Reflection Coefficient</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>Rn Noise Resistance</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>NCorr_11 Correlation(1,1)</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>NCorr_12 Correlation(1,2)</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>NCorr_21 Correlation(2,1)</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>----------------</td>
<td>----------</td>
<td>-----------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Incident DUT</td>
<td>Relative NPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Incident DUT NPwr Density</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Incident Syst</td>
<td>Relative NPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Incident Syst NPwr Density</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Available Noise Power Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Available DUT</td>
<td>Relative NPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
</tbody>
</table>

**NCorr_22 Correlation(2,2)**

<table>
<thead>
<tr>
<th>CALCulate:MEASure:PARameter</th>
<th>CreateCustomMeasurementEx</th>
</tr>
</thead>
</table>

**Meas Class...**

<table>
<thead>
<tr>
<th>CALCulate:MEASure:DEFine</th>
<th>CreateCustomMeasurementEx</th>
</tr>
</thead>
</table>

**Incident Noise Power Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Available DUT</td>
<td>Relative NPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Available DUT NPwr Density</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------</td>
<td>------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Available Syst Relative NPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Available Syst NPwr Density</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Noise Correlation Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCorr_11</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>NCorr_12</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>NCorr_21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>NCorr_22</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### S-Param Tab Commands
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>S11</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>S21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>S12</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>S22</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Receivers Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>A Source Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>B Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>B Source Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>R1 Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>R2 Source Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------</td>
<td>-----------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Meas Setup Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conversions</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Correction

<table>
<thead>
<tr>
<th>Channel Correction On</th>
<th>SENSE:CORRection[:STATe]</th>
<th>ErrorCorrection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel Correction Off</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cal Set...</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Smart Cal...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Trace Hold

<table>
<thead>
<tr>
<th>Equation Editor...</th>
<th></th>
<th></th>
</tr>
</thead>
</table>

#### Memory...

<table>
<thead>
<tr>
<th>Time Domain...</th>
<th></th>
<th></th>
</tr>
</thead>
</table>
Pulse Setup...
# CF_Meas Commands - NFX

Meas Commands - Noise Figure Converters Measurement Class

Click [here](##) to view links to Meas commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>NF</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>T-Eff</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>ENR</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>NFmin</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>GammaOpt</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Rn</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td>NF Noise Figure</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td>T-Eff Noise Temperature</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td>ENR Excess Noise Ratio</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Comment</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------------</td>
<td>--------------------------</td>
<td></td>
</tr>
<tr>
<td>DUTRNPN Available DUT Relative Noise Power</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>SYSRNPN Available System Relative Noise Power</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>DUTNPD Available DUT Noise Power Density</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>SYSNPD Available System Noise Power Density</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>DUTRNPI Incident DUT Relative Noise Power</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>SYSRNPI Incident System Relative Noise Power</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>DUTNPDI Incident DUT Noise Power Density</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>SYSNPDI Incident System Noise Power Density</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>S11, SC21, SC12, S22</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>IPwr Input pwr at DUT-IN @ IN freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Command</td>
<td>Function</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>---------</td>
<td>----------</td>
</tr>
<tr>
<td>RevIPwr</td>
<td>Input pwr at DUT-OUT @ OUT freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>RevOPwr</td>
<td>Out pwr at DUT-IN @ IN freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>OPwr</td>
<td>Out Pwr at DUT-OUT @ OUT freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>ALO1</td>
<td>Pwr in Port-1 @ LO1 freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>BLO1</td>
<td>Pwr in Port-2 @ LO1 freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>R1LO1</td>
<td>pwr out Port-1 @ LO1 freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>R2LO1</td>
<td>pwr out Port-2 @ LO1 freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>R1_1</td>
<td>Input pwr at DUT-IN @ IN freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>R2_2</td>
<td>Input pwr at DUT-OUT @ OUT freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>A_1</td>
<td>Out pwr at DUT-IN @ IN freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------------------------</td>
<td>---------------------------</td>
<td></td>
</tr>
<tr>
<td>A_2 Out pwr at DUT-IN @ IN freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>B_1 Out Pwr at DUT-OUT @ OUT freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>B_2 Out Pwr at DUT-OUT @ OUT freq</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>NFmin Minimum Noise Figure</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>GammOpt Optimum Complex Reflection Coefficient</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>Rn Noise Resistance</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>NCorr_11 Correlation(1,1)</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>NCorr_12 Correlation(1,2)</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>NCorr_21 Correlation(2,1)</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>NCorr_22 Correlation(2,2)</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
</tbody>
</table>

**Incident Noise Power Tab Commands**
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incident DUT NPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Incident DUT NPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Incident Syst NPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Incident Syst NPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Available DUT NPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Available DUT NPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Available Syst NPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Available Syst NPrw Density</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------</td>
<td>-------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Noise Correlation Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCorr_11</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>NCorr_12</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>NCorr_21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>NCorr_22</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### S-Param Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>S11</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>SC21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td><strong>SC12</strong></td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td><strong>S22</strong></td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td><strong>Other...</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Power Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IPwr</strong></td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td><strong>OPwr</strong></td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td><strong>RevIPwr</strong></td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td><strong>RevOPwr</strong></td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td><strong>Other...</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Receivers Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A Source Port 1</strong></td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Source Port</td>
<td>On/Off</td>
<td>SCPI</td>
<td>Command</td>
</tr>
<tr>
<td>--------------</td>
<td>--------</td>
<td>---------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>A Source Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>B Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>B Source Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>R1 Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>R2 Source Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Meas Setup Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Conversions

<table>
<thead>
<tr>
<th>Correction</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel Correction On</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
<td></td>
</tr>
<tr>
<td>Channel Correction Off</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
<td></td>
</tr>
<tr>
<td>Cal Set...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smart Cal...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Equation Editor...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time Domain...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## CF_Meas Commands - PN

Meas Commands - Phase Noise Measurement Class

Click [here](#) to view links to Meas commands for all Measurement Classes.

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PN_b2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>AM_b2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VNA Input</th>
<th></th>
<th>SENSE:PN:RECeiver</th>
<th>None</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Noise Type</th>
<th>Phase Noise</th>
<th>SENSE:PN:NTYPE PNOise</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Residual Noise</td>
<td>SENSE:PN:NTYPE RESidual</td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Phase Noise Setup...</th>
<th></th>
</tr>
</thead>
</table>

| Meas Class... | CALCulate:MEASure:DEFine | CreateCustomMeasurementEx |

### Spurious & Noise Tables Tab Commands
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spur Analysis...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spurious Table...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spot Noise Table...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integrated Noise Table...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Meas Setup Tab Commands</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Conversions</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel Correction On</td>
<td></td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td>Channel Correction Off</td>
<td></td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td>Select Cal Set...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cal All...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Src Mod Cal...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------</td>
<td>------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Equation Editor...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time Domain...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CF_Meas Commands - SA
Meas Commands - Spectrum Analyzer Measurement Class

Click [here](#) to view links to Meas commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Receivers Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>A</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>D</td>
</tr>
<tr>
<td>R1</td>
</tr>
<tr>
<td>Other...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Waves Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
</tbody>
</table>

7470
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Conversions**

**Correction**

Channel Correction On

SENSe:CORRection[:STATe]

ErrorCorrection

Channel Correction Off

SENSe:CORRection[:STATe]

ErrorCorrection

Cal Set...

Cal All...
<table>
<thead>
<tr>
<th>Trace Hold</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Equation Editor</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## CF_Meas Commands - SMC

**Meas Commands - Scalar Mixer/Converter + Phase Measurement Class**

Click [here](#) to view links to Meas commands for all Measurement Classes.

### S-Param Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>S11</td>
<td>On/Off</td>
<td><code>CALCulate:MEASure:PARameter</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td>SC21</td>
<td>On/Off</td>
<td><code>CALCulate:MEASure:PARameter</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td>SC12</td>
<td>On/Off</td>
<td><code>CALCulate:MEASure:PARameter</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td>S22</td>
<td>On/Off</td>
<td><code>CALCulate:MEASure:PARameter</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td>Other...</td>
<td>SC21 Forward Conversion</td>
<td><code>CALCulate:MEASure:PARameter</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td></td>
<td>SC12 Reverse Conversion</td>
<td><code>CALCulate:MEASure:PARameter</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td></td>
<td>S11 Input Match</td>
<td><code>CALCulate:MEASure:PARameter</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td></td>
<td>S22 Output Match</td>
<td><code>CALCulate:MEASure:PARameter</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td></td>
<td>IPwr Input Power</td>
<td><code>CALCulate:MEASure:PARameter</code></td>
<td><code>CreateCustomMeasurementEx</code></td>
</tr>
<tr>
<td>Power Tab Commands</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------</td>
<td>------------------</td>
<td></td>
</tr>
<tr>
<td>OPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>RevIPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>RevOPwr</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>AI1,1 AI1</td>
<td></td>
<td>CALCulate:MEASure:PARameter CreateMeasurement</td>
<td></td>
</tr>
<tr>
<td>AI2,1 AI2</td>
<td></td>
<td>CALCulate:MEASure:PARameter CreateMeasurement</td>
<td></td>
</tr>
<tr>
<td>AI1,2 AI1</td>
<td></td>
<td>CALCulate:MEASure:PARameter CreateMeasurement</td>
<td></td>
</tr>
<tr>
<td>AI2,2 AI2</td>
<td></td>
<td>CALCulate:MEASure:PARameter CreateMeasurement</td>
<td></td>
</tr>
<tr>
<td>Meas Class...</td>
<td></td>
<td>CALCulate:MEASure:DEFine CreateCustomMeasurementEx</td>
<td></td>
</tr>
</tbody>
</table>
### Auxiliary Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuxIn1 Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateMeasurement</td>
</tr>
<tr>
<td>AuxIn1 Source Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateMeasurement</td>
</tr>
<tr>
<td>AuxIn2 Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateMeasurement</td>
</tr>
<tr>
<td>AuxIn2 Source Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateMeasurement</td>
</tr>
</tbody>
</table>

### Meas Setup Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conversions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td>Channel Correction On</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------</td>
<td>---------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td>Channel Correction Off</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td></td>
<td>Cal Set...</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Smart Cal...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equation Editor...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time Domain...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## CF_Meas Commands - Standard

### Meas Commands - Standard Measurement Class

Click [here](#) to view links to Meas commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>S-Param</th>
<th>Balanced</th>
<th>Receivers</th>
<th>Waves</th>
<th>Auxiliary</th>
<th>Meas Setup</th>
</tr>
</thead>
</table>

### S-Param Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>S11</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
</tr>
<tr>
<td>S21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
</tr>
<tr>
<td>S12</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
</tr>
<tr>
<td>S22</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
</tr>
<tr>
<td>Other...</td>
<td>S11, S12, S13, S14, S21, S22, S23, S24, S31, S32, S33, S34, S41, S42, S43, S44</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
</tr>
<tr>
<td>S-Parameter</td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
</tr>
</tbody>
</table>
### Balanced Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ss11</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BalSMeasurement</td>
</tr>
<tr>
<td>Sds21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BalSMeasurement</td>
</tr>
<tr>
<td>Ssd12</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BalSMeasurement</td>
</tr>
<tr>
<td>Sdd22</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BalSMeasurement</td>
</tr>
<tr>
<td>Other...</td>
<td>Ssd11, Scd11, Sdc11</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BalSMeasurement</td>
</tr>
<tr>
<td>Scc11, Ssd21, Ssc21, Sds12, Scs12, Sss22</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BBalMeasurement</td>
<td></td>
</tr>
<tr>
<td>Sdd11, Sdd12, Sdd22, Scd11, Scd21, Sdc11, Sdc12, Sdc21, Sdc22, Scs11, Scs12, Scs21, Scs22</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BSSMeasurement</td>
<td></td>
</tr>
<tr>
<td>Sdd11, Scd11, Sdc11, Scc11, Ssd21, Ssc21, Ssd31, Ssc31, Sds12, Sds13, Scs12, Scs13, Sss22, Sss32, Sss23, Sss33</td>
<td>CALCulate:MEASure:PARameter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sss11, Ssd12, Ssc12, Sds21, Scs21, Sdd22, Sdc22, Scd22, Scs21, Scs22</td>
<td>CALCulate:MEASure:PARameter</td>
<td>SBalMeasurement</td>
<td></td>
</tr>
<tr>
<td>Sss11, Sss12, Sss21, Sss22, Ssd13, Ssc13, Ssd23, Ssc23, Sds31, Sds32, Scs31, Scs32, Sdd33, Scd33, Scc33</td>
<td>CALCulate:MEASure:PARameter</td>
<td>SSBMeasurement</td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td>-------------------------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>S-Parameter</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
<td></td>
</tr>
<tr>
<td>Balanced</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BBalMeasurement</td>
<td></td>
</tr>
<tr>
<td>Receivers</td>
<td>CALCulate:MEASure:PARameter</td>
<td>App.CreateCustomMeasurementEx</td>
<td></td>
</tr>
</tbody>
</table>

**Topology...**  
**Topology Tab**

<table>
<thead>
<tr>
<th>BAL</th>
<th>CALCulate:FSIMulator:BALun:DEVice</th>
<th>SetBPort</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAL-BAL</td>
<td>CALCulate:FSIMulator:BALun:DEVice</td>
<td>SetBBPorts</td>
</tr>
<tr>
<td>BAL-SE</td>
<td>CALCulate:FSIMulator:BALun:DEVice</td>
<td>SetBSPorts</td>
</tr>
<tr>
<td>BAL-SE-SE</td>
<td>CALCulate:FSIMulator:BALun:DEVice</td>
<td>SetBSSPorts</td>
</tr>
<tr>
<td>Configuration</td>
<td>Command</td>
<td>Functions</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>SE-BAL</td>
<td>CALCulate:FSIMulator:BALun:DEVice</td>
<td>SetSBPorts</td>
</tr>
<tr>
<td>SE-SE-BAL</td>
<td>CALCulate:FSIMulator:BALun:DEVice</td>
<td>SetSSBPorts</td>
</tr>
<tr>
<td>Custom</td>
<td>CALCulate:FSIMulator:BALun:DEVice</td>
<td>SetCustomDUTTopology</td>
</tr>
<tr>
<td>Balanced Port</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(set specific port)</td>
<td>CALCulate:FSIMulator:BPORt:STIMulus:TRUe:STATe</td>
<td>BalancedPortTrueState</td>
</tr>
<tr>
<td></td>
<td>CALCulate:DTOPology</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SetCustomDUTTopology</td>
</tr>
<tr>
<td>Balanced Port</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(logical port)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>VNA (+)Port</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VNA (-)Port</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:TOPology:BALanced[:PPORts]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:TOPology:BALSended[:PPORts]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:TOPology:BBALanced[:PPORts]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:TOPology:BSSended[:PPORts]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:TOPology:SBALanced[:PPORts]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:TOPology:SSBalanced[:PPORts]</td>
<td></td>
</tr>
<tr>
<td>True Mode</td>
<td>CALCulate:FSIMulator:STIMulus:MODE</td>
<td>Mode</td>
</tr>
<tr>
<td>Port Z Tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable Differential Z Conversion</td>
<td>\texttt{CALCulate:FSIMulator:BALun:DZConversion:STATe}</td>
<td>DiffZConvState</td>
</tr>
<tr>
<td>Enable Common Z Conversion</td>
<td>\texttt{CALCulate:FSIMulator:BALun:CZConversion:STATe}</td>
<td>CmnModeZConvState</td>
</tr>
<tr>
<td>Enable SE Port Z Conversion</td>
<td>\texttt{CALCulate:FSIMulator:SENDed:ZCONversion:STATe}</td>
<td>PortArbzState</td>
</tr>
<tr>
<td>Converted Z Real</td>
<td>\texttt{CALCulate:FSIMulator:BALun:CZConversion:BPORt:REAL}</td>
<td>PortArbzReal</td>
</tr>
<tr>
<td>Converted Z Imaginary</td>
<td>\texttt{CALCulate:FSIMulator:BALun:CZConversion:BPORt:IMAG}</td>
<td>PortArbzImag</td>
</tr>
</tbody>
</table>

**Offset Tab**

<p>| Apply Power Offset As Fixture  | \texttt{CALCulate:FSIMulator:BALun:FIXTure:OFFSet:POWer} | PowerAsFixture |
| Apply Phase Offset As Fixture  | \texttt{CALCulate:FSIMulator:BALun:FIXTure:OFFSet:PHASE} | PhaseAsFixture |
| Power Offset                  | \texttt{CALCulate:FSIMulator:BALun:BPORt:OFFSet:POWer} | BalPort1PowerOffset, BalPort2PowerOffset |</p>
<table>
<thead>
<tr>
<th>Feature</th>
<th>Command Details</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase Offset</td>
<td>CALCulate:FSIMulator:BALun:BPORt:OFFSET:PHASE</td>
<td>BalPort1PhaseOffset, BalPort2PhaseOffset</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:BALun:LPORt:OFFSET:PHASE</td>
<td>None</td>
</tr>
<tr>
<td>Sweep Tab</td>
<td>CALCulate:FSIMulator:PHASE:Sweep:PORT</td>
<td>None</td>
</tr>
<tr>
<td>Select Active Source</td>
<td>CALCulate:FSIMulator:PHASE:Sweep:PORT</td>
<td>None</td>
</tr>
<tr>
<td>Enable Balanced Phase Sweep</td>
<td>CALCulate:FSIMulator:PHASE:Sweep:STATE</td>
<td>PhaseSwpState</td>
</tr>
<tr>
<td>Start Phase</td>
<td>CALCulate:FSIMulator:BPORt:Sweep:PHASE:SSTART</td>
<td>BalPort1StartPhase, BalPort2StartPhase</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:LPORt:Sweep:PHASE:SSTART</td>
<td>None</td>
</tr>
<tr>
<td>Stop Phase</td>
<td>CALCulate:FSIMulator:BPORt:Sweep:PHASE:STOP</td>
<td>BalPort1StopPhase, BalPort2StopPhase</td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:LPORt:Sweep:PHASE:STOP</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>Offset as fixture</td>
<td>CALCulate:FSIMulator:BALun:FIXTure:PHASe</td>
<td>PhaseSwpAsFixture</td>
</tr>
<tr>
<td>Enable Differential Source Only Mode</td>
<td>CALCulate:FSIMulator:BALun:STIMulus:MODE</td>
<td>Mode</td>
</tr>
</tbody>
</table>

**Receivers Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>A Source Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>B Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>B Source Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>R1 Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Source</td>
<td>Port</td>
<td>On/Off</td>
<td>SCPI Command</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
<td>--------</td>
<td>--------------</td>
</tr>
<tr>
<td>R2 Source</td>
<td>Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Other...</td>
<td></td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>S-Parameter</td>
<td></td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Balanced</td>
<td></td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Receivers</td>
<td></td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
</tbody>
</table>

**Waves Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1 Source Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>a2 Source Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Port Type</td>
<td>Port</td>
<td>Action</td>
<td>Command</td>
</tr>
<tr>
<td>-----------</td>
<td>------</td>
<td>--------</td>
<td>---------</td>
</tr>
<tr>
<td>b1 Source</td>
<td>Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>b1 Source</td>
<td>Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>b2 Source</td>
<td>Port 1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>b2 Source</td>
<td>Port 2</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Other...</td>
<td>(a1,1), (a2,2), (a3,3), (a4,4), (b1,1), (b1,2), (b1,3), (b1,4), (b2,1), (b2,2), (b2,3), (b2,4), (b3,1), (b3,2), (b3,3), (b3,4), (b4,1), (b4,2), (b4,3), (b4,4)</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>S-Parameter</td>
<td></td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Balanced</td>
<td></td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Receivers</td>
<td></td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
</tbody>
</table>

**Auxiliary Tab Commands**

7486
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuxIn1 Source</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Port 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AuxIn1 Source</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Port 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AuxIn2 Source</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Port 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AuxIn2 Source</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Port 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other...</td>
<td>(AuxIn1,1),</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td>(AuxIn1,2),</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(AuxIn1,3),</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(AuxIn1,4),</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(AuxIn2,1),</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(AuxIn2,2),</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(AuxIn2,3),</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(AuxIn2,4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S-Parameter</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateS-Parameter</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------</td>
<td>-------------------</td>
<td></td>
</tr>
<tr>
<td>Balanced</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BBalMeasurement</td>
<td></td>
</tr>
<tr>
<td>Receivers</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
</tbody>
</table>

**Meas Setup Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI/COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>CALCulate:MEASure:CONVersion:FUNCtion</td>
</tr>
<tr>
<td>Conversions</td>
<td>Off</td>
<td>CALCulate:MEASure:CONVersion:FUNCtion</td>
</tr>
<tr>
<td></td>
<td>Z-Reflect</td>
<td>CALCulate:MEASure:CONVersion:FUNCtion</td>
</tr>
<tr>
<td></td>
<td>Z-Transmit</td>
<td>CALCulate:MEASure:CONVersion:FUNCtion</td>
</tr>
<tr>
<td></td>
<td>Z-Trans-Shunt</td>
<td>CALCulate:MEASure:CONVersion:FUNCtion</td>
</tr>
<tr>
<td></td>
<td>Y-Reflect</td>
<td>CALCulate:MEASure:CONVersion:FUNCtion</td>
</tr>
<tr>
<td></td>
<td>Y-Transmit</td>
<td>CALCulate:MEASure:CONVersion:FUNCtion</td>
</tr>
<tr>
<td></td>
<td>Y-Trans-Shunt</td>
<td>CALCulate:MEASure:CONVersion:FUNCtion</td>
</tr>
<tr>
<td></td>
<td>1 / S</td>
<td>CALCulate:MEASure:CONVersion:FUNCtion</td>
</tr>
<tr>
<td>Correction</td>
<td>Channel Correction On</td>
<td>SENSe:CORRection[:STATe]</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td></td>
<td>Channel Correction Off</td>
<td>SENSe:CORRection[:STATe]</td>
</tr>
<tr>
<td></td>
<td>Cal Set...</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Basic Cal...</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Smart Cal...</td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td>Off</td>
<td>CALCulate:MEASure:HOLD:TYPE</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>CALCulate:MEASure:HOLD:TYPE</td>
</tr>
<tr>
<td></td>
<td>Min</td>
<td>CALCulate:MEASure:HOLD:TYPE</td>
</tr>
<tr>
<td></td>
<td>Restart</td>
<td>CALCulate:MEASure:HOLD:CLEar</td>
</tr>
<tr>
<td>Equation Editor...</td>
<td>On/Off</td>
<td>CALCulate:MEASure:EQUation[:STATe]</td>
</tr>
<tr>
<td></td>
<td>Equation</td>
<td>CALCulate:MEASure:EQUation:TEXT</td>
</tr>
<tr>
<td></td>
<td>Enable Equation</td>
<td>CALCulate:MEASure:EQUation[:STATe]</td>
</tr>
<tr>
<td>Functions</td>
<td>CALCulate:EQUation:LIBRary:FUNCtions</td>
<td>GetLibraryFunctions</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Trace Data</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Parameter</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Store Equation</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Delete Equation</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Use Short Names</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Enable Matlab</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Import...</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Import Library</td>
<td>CALCulate:EQUation:LIBRary:IMPort</td>
<td>ImportLibrary</td>
</tr>
<tr>
<td>Remove Library</td>
<td>CALCulate:EQUation:LIBRary:REMove</td>
<td>RemoveLibrary</td>
</tr>
<tr>
<td>Memory...</td>
<td>Data-&gt;Memory</td>
<td>DataToMemory</td>
</tr>
<tr>
<td>Data Math</td>
<td></td>
<td>TraceMath</td>
</tr>
<tr>
<td>Off</td>
<td>CALCulate:MEASure:MATH:FUNCtion</td>
<td>TraceMath</td>
</tr>
<tr>
<td><strong>Data / Memory</strong></td>
<td><strong>CALCulate:MEASure:MATH:FUNCTION</strong></td>
<td><strong>TraceMath</strong></td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><strong>Data * Memory</strong></td>
<td><strong>CALCulate:MEASure:MATH:FUNCTION</strong></td>
<td><strong>TraceMath</strong></td>
</tr>
<tr>
<td><strong>Data - Memory</strong></td>
<td><strong>CALCulate:MEASure:MATH:FUNCTION</strong></td>
<td><strong>TraceMath</strong></td>
</tr>
<tr>
<td><strong>Data + Memory</strong></td>
<td><strong>CALCulate:MEASure:MATH:FUNCTION</strong></td>
<td><strong>TraceMath</strong></td>
</tr>
<tr>
<td><strong>8510 Mode</strong></td>
<td><strong>None</strong></td>
<td><strong>None</strong></td>
</tr>
</tbody>
</table>

**Trace View Options**

<table>
<thead>
<tr>
<th><strong>Data Trace</strong></th>
<th><strong>DISPlay:WINDow:TRACe[:STATe]</strong></th>
<th><strong>View</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Memory Trace</strong></td>
<td><strong>DISPlay:WINDow:TRACe:MEMory</strong></td>
<td><strong>View</strong></td>
</tr>
<tr>
<td><strong>Data and Memory Trace</strong></td>
<td><strong>DISPlay:WINDow:TRACe[:STATe]</strong></td>
<td><strong>View</strong></td>
</tr>
<tr>
<td></td>
<td><strong>DISPlay:WINDow:TRACe:MEMory</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Time Domain...**

<table>
<thead>
<tr>
<th><strong>Transform</strong></th>
<th><strong>CALCulate:MEASure:TRANSform:TIME:STATe</strong></th>
<th><strong>State</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Transform On</strong></td>
<td><strong>CALCulate:MEASure:TRANSform:TIME:START</strong></td>
<td><strong>Start</strong></td>
</tr>
<tr>
<td><strong>Start</strong></td>
<td><strong>CALCulate:MEASure:TRANSform:TIME:STOP</strong></td>
<td><strong>Stop</strong></td>
</tr>
<tr>
<td><strong>Stop</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Center</td>
<td>CALCulate:MEASure:TRANsform:TIME:CENTer</td>
<td>Center</td>
</tr>
<tr>
<td>Span</td>
<td>CALCulate:MEASure:TRANsform:TIME:SPAN</td>
<td>Span</td>
</tr>
<tr>
<td>Set Low Pass Frequencies</td>
<td>CALCulate:MEASure:TRANsform:TIME:LPFREQuency</td>
<td>SetFrequencyLowPass</td>
</tr>
</tbody>
</table>

**Gating**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gating On</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STATE</td>
<td>State</td>
</tr>
<tr>
<td>Start</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STARt</td>
<td>Start</td>
</tr>
<tr>
<td>Stop</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:STOP</td>
<td>Stop</td>
</tr>
<tr>
<td>Center</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:CENTer</td>
<td>Center</td>
</tr>
<tr>
<td>Span</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME:SPAN</td>
<td>Span</td>
</tr>
<tr>
<td>Gate Type</td>
<td>CALCulate:MEASure:FILTer[:GATE]:TIME[:TYPE]</td>
<td>Type</td>
</tr>
<tr>
<td>Gate Shape</td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME:SHAPe</td>
<td>Shape</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Window</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Minimum/Maximum</td>
<td>DISPlay:WNDow:SIZE</td>
<td>app.WindowState</td>
</tr>
<tr>
<td>Impulse Width</td>
<td>CALCulate:MEASure:TRANSform:TIME:IMPulse:WIDTh</td>
<td>ImpulseWidth</td>
</tr>
<tr>
<td><strong>Coupling</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Coupling On</td>
<td>SENSE:COUPLE:PARameter[:STATe]</td>
<td>CoupleChannelParams</td>
</tr>
<tr>
<td><strong>Transform Parameters</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stimulus</td>
<td>CALCulate:MEASure:TRANSform:COUPLE:PARameters</td>
<td>CoupledParameters</td>
</tr>
<tr>
<td>State (On/Off)</td>
<td>CALCulate:MEASure:TRANSform:COUPLE:PARameters</td>
<td>CoupledParameters</td>
</tr>
<tr>
<td>Window</td>
<td>CALCulate:MEASure:TRANSform:COUPLE:PARameters</td>
<td>CoupledParameters</td>
</tr>
<tr>
<td>Mode</td>
<td>CALCulate:MEASure:TRANSform:COUPLE:PARameters</td>
<td>CoupledParameters</td>
</tr>
<tr>
<td>Distance Marker Unit</td>
<td>CALCulate:MEASure:TRANSform:COUPLE:PARameters</td>
<td>CoupledParameters</td>
</tr>
<tr>
<td>Gating Parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------</td>
<td>------------------</td>
</tr>
<tr>
<td><strong>Stimulus</strong></td>
<td>CALCulate:MEASure:FILTER[:GATE]:COUPle:PARameters</td>
<td>CoupledParameters (Gating)</td>
</tr>
<tr>
<td><strong>State (On/Off)</strong></td>
<td>CALCulate:MEASure:FILTER[:GATE]:COUPle:PARameters</td>
<td>CoupledParameters (Gating)</td>
</tr>
<tr>
<td><strong>Shape</strong></td>
<td>CALCulate:MEASure:FILTER[:GATE]:COUPle:PARameters</td>
<td>CoupledParameters (Gating)</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>CALCulate:MEASure:FILTER[:GATE]:COUPle:PARameters</td>
<td>CoupledParameters (Gating)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Marker</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Auto</strong></td>
<td>CALCulate:MEASure:TRANSform:TIME:MARKer:MODE</td>
<td>DistanceMarkerMode</td>
</tr>
<tr>
<td><strong>Reflection (divide by 2)</strong></td>
<td>CALCulate:MEASure:TRANSform:TIME:MARKer:MODE</td>
<td>DistanceMarkerMode</td>
</tr>
<tr>
<td><strong>Transmission</strong></td>
<td>CALCulate:MEASure:TRANSform:TIME:MARKer:MODE</td>
<td>DistanceMarkerMode</td>
</tr>
<tr>
<td><strong>Meters (m)</strong></td>
<td>CALCulate:MEASure:TRANSform:TIME:MARKer:UNIT</td>
<td>DistanceMarkerUnit</td>
</tr>
<tr>
<td><strong>Feet (ft)</strong></td>
<td>CALCulate:MEASure:TRANSform:TIME:MARKer:UNIT</td>
<td>DistanceMarkerUnit</td>
</tr>
<tr>
<td><strong>Inches (in)</strong></td>
<td>CALCulate:MEASure:TRANSform:TIME:MARKer:UNIT</td>
<td>DistanceMarkerUnit</td>
</tr>
<tr>
<td>Velocity Factor</td>
<td>SENSE:CORRection:EXTension:PORT:VELFactor</td>
<td>VelocityFactor</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CF_Meas Commands - VMC

Meas Commands - Vector Mixer/Converter Measurement Class

Click [here](#) to view links to Meas commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>S11</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>VC21</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>S22</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>R1</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>B</td>
<td>On/Off</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Other...</td>
<td>S11 Input Match</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td>S22 Output Match</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td>VC21 Forward Conversion</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td>R1 R1 ref mixer diagnostic</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------------</td>
<td>-------------------------------------</td>
<td>-----</td>
</tr>
<tr>
<td>Conversions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td>Channel Correction On</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td></td>
<td>Channel Correction Off</td>
<td>SENSE:CORRection[:STATe]</td>
<td>ErrorCorrection</td>
</tr>
<tr>
<td></td>
<td>Cal Set...</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Smart Cal...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equation Editor...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time Domain...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CF_Power Commands

The Power softkeys vary depending on which measurement class is currently active. Click on a measurement class link below to view the corresponding softkeys.

Standard
Active Hot Parameters
Gain Compression and Gain Compression Converters
Differential I/Q
IM Spectrum and IM Spectrum Converters
Swept IMD and Swept IMD Converters
Modulation Distortion and Modulation Distortion Converters
Noise Figure Cold Source and Noise Figure Converters
Phase Noise
Scalar Mixer/Converter + Phase and Vector Mixer/Converter
Spectrum Analyzer
CF_Power Commands - DIQ

Power Commands - Differential I/Q Measurement Class

Only the Main and Leveling & Offsets Power commands corresponding to the Differential I/Q measurement class are documented here. The Attenuators commands are identical to the Attenuator commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtab on the graphic below.

Click here to view links to Power commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>RF Power</td>
</tr>
<tr>
<td>Power and Attenuators...</td>
</tr>
<tr>
<td>DIQ Setup...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Leveling &amp; Offsets Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>Select</td>
</tr>
<tr>
<td>Port 2</td>
</tr>
<tr>
<td>Port 3</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>Port 4</td>
</tr>
<tr>
<td>Port 1 Src2</td>
</tr>
<tr>
<td>Source3</td>
</tr>
<tr>
<td>Offset</td>
</tr>
<tr>
<td>Limit</td>
</tr>
<tr>
<td>Power Limit</td>
</tr>
<tr>
<td>Offsets and Limits...</td>
</tr>
</tbody>
</table>
CF_Power Commands - GCA_GCX

Power Commands - Gain Compression and Gain Compression Converters Measurement Classes

Only the Main, Compression Levels, and Leveling & Offsets Power commands corresponding to the Gain Compressions and Gain Compression Converters measurement classes are documented here. The Attenuators commands are identical to the Attenuator commands for the Standard measurement class and can be accessed by clicking [here](#) or by clicking on the softtab on the graphic below.

Click [here](#) to view links to Power commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Input Pwr</td>
<td>ON/OFF</td>
<td>SENSE:GCSetup:POWER:LINEar:INPUT:LEVEL</td>
<td>InputLinearPowerLevel</td>
</tr>
<tr>
<td>RF Power</td>
<td></td>
<td>OUTPUT[:STATe]</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td>Start Power</td>
<td></td>
<td>SOURce:POWER:PORT:STARt</td>
<td>StartPower</td>
</tr>
<tr>
<td>Stop Power</td>
<td></td>
<td>SOURce:POWER:PORT:STOP</td>
<td>StopPower</td>
</tr>
<tr>
<td>Power and Attenuators...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GCA Setup...</td>
<td></td>
<td></td>
<td>(Gain Compression Measurement Class only)</td>
</tr>
</tbody>
</table>
### GCX Setup...
(Gain Compression Converters Measurement Class only)

#### Compress Levels Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comp Method</td>
<td>Linear Gain</td>
<td><code>SENSe:GCSetup:POWer:LINear:INPut:LEVel</code></td>
<td>InputLinearPowerLevel</td>
</tr>
<tr>
<td></td>
<td>Max Gain</td>
<td><code>SENSe:GCSetup:COMPression:LEVel</code></td>
<td>CompressionLevel</td>
</tr>
<tr>
<td></td>
<td>Backoff</td>
<td><code>SENSe:GCSetup:COMPression:BACKoff:LEVel</code></td>
<td>CompressionBackoff</td>
</tr>
<tr>
<td></td>
<td>XY</td>
<td><code>SENSe:GCSetup:COMPression:DELTa:X</code></td>
<td>CompressionDeltaX</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>SENSe:GCSetup:COMPression:DELTa:Y</code></td>
<td>CompressionDeltaY</td>
</tr>
<tr>
<td></td>
<td>Saturation</td>
<td><code>SENSe:GCSetup:COMPression:SATuration:LEVel</code></td>
<td>CompressionSaturation</td>
</tr>
<tr>
<td>Linear Input Pwr</td>
<td></td>
<td><code>SENSe:GCSetup:POWer:LINear:INPut:LEVel</code></td>
<td>InputLinearPowerLevel</td>
</tr>
<tr>
<td>Reverse Pwr</td>
<td></td>
<td><code>SENSe:GCSetup:POWer:REVerse:LEVel</code></td>
<td>ReverseLinearPowerLevel</td>
</tr>
<tr>
<td>Compression Level</td>
<td></td>
<td><code>SENSe:GCSetup:COMPression:LEVel</code></td>
<td>CompressionLevel</td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>-------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Select</td>
<td>Port 1</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 2</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 3</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 4</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 1 Src2</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Source3</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Offset</td>
<td></td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:OFFSet</td>
<td>PowerOffset</td>
</tr>
</tbody>
</table>

<p>| Back Off Level | SENSe:GCSetup:COMPression:BACKoff:LEVel | CompressionBackoff |
| Delta X       | SENSe:GCSetup:COMPression:DELTa:X       | CompressionDeltaX  |
| Delta Y       | SENSe:GCSetup:COMPression:DELTa:Y       | CompressionDeltaY  |
| Saturation    | SENSe:GCSetup:COMPression:SATuration:LEVel | CompressionSaturation |</p>
<table>
<thead>
<tr>
<th>Limit</th>
<th>ON/OFF</th>
<th>SYSTem:POWer:LIMit:STATe</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Limit</td>
<td></td>
<td>SYSTem:POWer:LI Mit</td>
<td>Limit</td>
</tr>
<tr>
<td>Offsets and Limits...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALC Hardware</td>
<td>Internal</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
<tr>
<td>Open Loop</td>
<td></td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
<tr>
<td>Receiver Leveling...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CF_Power Commands - HotS22

Power Commands - Active Hot Parameters

Only the Main and Leveling & Offsets Power commands corresponding to the Active Hot Parameters measurement classes are documented here. The Port Power and Attenuators commands are identical to the Attenuator commands for the Standard measurement class and can be accessed by clicking [here](#) or by clicking on the softtab on the graphic below.

Click [here](#) to view links to Power commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main</th>
<th>Port Power</th>
<th>Leveling &amp; Offsets</th>
<th>Attenuators</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Level</td>
<td></td>
<td>SOURce:POWer[:LEVel][:IMMediate][:AMPLitude]</td>
<td>TestPortPower PowerSpinResolution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DISPlay:GUI:POWer:SPIN:RESolution</td>
<td></td>
</tr>
<tr>
<td>RF Power</td>
<td>ON/OFF</td>
<td>OUTPut[:STATe]</td>
<td>SourcePortMode DisplayInputPower</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SENSE:ACTive:DISPlay:TRACe:IPWer</td>
<td></td>
</tr>
<tr>
<td>Start Power</td>
<td></td>
<td>SOURce:POWer:PORT:STARt</td>
<td>StartPower</td>
</tr>
<tr>
<td>Stop Power</td>
<td></td>
<td>SOURce:POWer:PORT:STOP</td>
<td>StopPower</td>
</tr>
<tr>
<td>Power and Attenuators...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AHP Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Leveling & Offsets Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>Port 1</td>
<td>None</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td></td>
<td>Port 2</td>
<td>None</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td></td>
<td>Port 3</td>
<td>None</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td></td>
<td>Port 4</td>
<td>None</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td>Slope</td>
<td>ON/OFF</td>
<td>SOURce:POWer[:LEVel]:SLOPe:STATe</td>
<td>PowerSlopeState</td>
</tr>
<tr>
<td></td>
<td>Power Slope</td>
<td>SOURce:POWer[:LEVel]:SLOPe</td>
<td>PowerSlope</td>
</tr>
<tr>
<td>Offset</td>
<td></td>
<td>SOURce:POWer:CORRection:OFFSet:MAGNitude</td>
<td>PowerOffset</td>
</tr>
<tr>
<td>Limit</td>
<td>ON/OFF</td>
<td>SYSTem:POWer:LIMit:STATe</td>
<td>State</td>
</tr>
<tr>
<td></td>
<td>Power Limit</td>
<td>SYSTem:POWer:LIMit</td>
<td>Limit</td>
</tr>
<tr>
<td>Offsets and Limits...</td>
<td>Power Limit - State</td>
<td>SYSTem:POWer:LIMit:STATe</td>
<td>State</td>
</tr>
<tr>
<td></td>
<td>Power Limit - Limit</td>
<td>SYSTem:POWer:LIMit</td>
<td>Limit</td>
</tr>
<tr>
<td>Source Power</td>
<td>SOURce:POWer[:LEVe][:IMMediate][:AMPLitude]</td>
<td>TestPortPower</td>
<td></td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>Power Offset</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver.OFFSet</td>
<td>PowerOffset</td>
<td></td>
</tr>
<tr>
<td>Port Power</td>
<td>SOURce:POWer:PORT:STARt</td>
<td>StartPower</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SOURce:POWer:PORT:STOP</td>
<td>StopPower</td>
<td></td>
</tr>
</tbody>
</table>
CF_Power Commands - IMD_IMDX

Power Commands - Swept IMD and Swept IMD Converters Measurement Class

Only the Main and Leveling & Offsets Power commands corresponding to the Swept IMD and Swept IMD Converters measurement classes are documented here. The Port Power and Attenuators commands are identical to the Port Power and Attenuator commands for the Standard measurement class and can be accessed by clicking on the softtab on the graphic below.

Click [here](#) to view links to Power commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main</th>
<th>Port Power</th>
<th>Leveling &amp; Offsets</th>
<th>Attenuators</th>
</tr>
</thead>
</table>

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tone Power</strong></td>
<td></td>
<td>SENSe:IMD:TPOWer:F1</td>
<td>TonePower</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SENSe:IMD:TPOWer:F2</td>
<td>TonePower</td>
</tr>
<tr>
<td><strong>RF Power</strong></td>
<td>ON/OFF</td>
<td>OUTPut[:STATe]</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td><strong>Start Power</strong></td>
<td></td>
<td>SOURce:POWer:PORT:STARt</td>
<td>StartPower</td>
</tr>
<tr>
<td><strong>Stop Power</strong></td>
<td></td>
<td>SOURce:POWer:PORT:STOP</td>
<td>StopPower</td>
</tr>
<tr>
<td><strong>Power and Attenuators...</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>IMD Setup...</strong></td>
<td>(Swept IMD Measurement Class only)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### IMDX Setup...
(Swept IMD Converters Measurement Class only)

<table>
<thead>
<tr>
<th>Leveling &amp; Offsets Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td>Select</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Offset</strong></td>
</tr>
<tr>
<td><strong>Limit</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Offsets and Limits...</td>
</tr>
<tr>
<td>----------------------</td>
</tr>
<tr>
<td>ALC Hardware</td>
</tr>
<tr>
<td>Open Loop</td>
</tr>
<tr>
<td>Receiver Leveling...</td>
</tr>
</tbody>
</table>
CF_Power Commands - IMS_IMSX

Power Commands - IM Spectrum and IM Spectrum Converters Measurement Classes

Only the Main and Leveling & Offsets Power commands corresponding to the IM Spectrum and IM Spectrum Converters measurement classes are documented here. The Port Power and Attenuators commands are identical to the Port Power and Attenuator commands for the Standard measurement class and can be accessed by clicking on the softtab on the graphic below.

Click [here](#) to view links to Power commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main</th>
<th>Port Power</th>
<th>Leveling &amp; Offsets</th>
<th>Attenuators</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Main Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Tone Power**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**RF Power**

<table>
<thead>
<tr>
<th>RF Power</th>
<th>ON/OFF</th>
<th>OUTPut[:STATe]</th>
<th>SourcePortMode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Power and Attenuators...**

<table>
<thead>
<tr>
<th>Power and Attenuators...</th>
<th></th>
</tr>
</thead>
</table>

**IMS Setup...**

(IMS Spectrum Measurement Class only)

<table>
<thead>
<tr>
<th>IMS Setup...</th>
<th></th>
</tr>
</thead>
</table>

**IMSX Setup...**

(IM Spectrum Converters Measurement Class only)

<table>
<thead>
<tr>
<th>IMSX Setup...</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
</tr>
<tr>
<td>---------------</td>
<td>----------</td>
</tr>
<tr>
<td>Select</td>
<td>Port 1</td>
</tr>
<tr>
<td></td>
<td>Port 2</td>
</tr>
<tr>
<td></td>
<td>Port 3</td>
</tr>
<tr>
<td></td>
<td>Port 4</td>
</tr>
<tr>
<td>Source3</td>
<td>None</td>
</tr>
<tr>
<td>Offset</td>
<td></td>
</tr>
<tr>
<td>Limit</td>
<td>ON/OFF</td>
</tr>
<tr>
<td></td>
<td>Power</td>
</tr>
</tbody>
</table>

Offsets and Limits...
<table>
<thead>
<tr>
<th>ALC Hardware</th>
<th>Internal</th>
<th>SOURc:POWer:ALC[:MODE]</th>
<th>ALCLevelingMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Loop</td>
<td></td>
<td>SOURc:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
<tr>
<td>Receiver Leveling...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

7514
## CF_Power Commands - MOD

### Power Commands - Modulation Distortion and Modulation Distortion Converters Measurement Class

Click [here](#) to view links to Power commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carrier Level</td>
<td></td>
<td><code>SENSe:DISTortion:SWEep:POWer:CARRier:LEVel</code></td>
<td>None</td>
</tr>
<tr>
<td>RF Power</td>
<td>ON/OFF</td>
<td><code>OUTPut[:STATe]</code></td>
<td><code>SourcePortMode</code></td>
</tr>
<tr>
<td>S-Param Input Pwr</td>
<td></td>
<td><code>SENSe:DISTortion:SWEep:POWer:SPARam:LEVel</code></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>SENSe:DISTortion:SWEep:POWer:CARRier:LEVel:PORT</code></td>
<td>None</td>
</tr>
<tr>
<td>Start Power</td>
<td></td>
<td><code>SENSe:DISTortion:SWEep:POWer:CARRier:LEVel</code></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>SENSe:DISTortion:SWEep:POWer:CARRier:LEVel:PORT</code></td>
<td>None</td>
</tr>
<tr>
<td>Stop Power</td>
<td></td>
<td><code>SENSe:DISTortion:SWEep:POWer:CARRier:LEVel</code></td>
<td>None</td>
</tr>
<tr>
<td>Power and Attenuators...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOD Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>----------</td>
<td>----------</td>
<td>--------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Offset</td>
<td></td>
<td>SOURce:POWer:CORRection:OFFSet:MAGNitude</td>
<td>PowerOffset</td>
</tr>
<tr>
<td>Limit</td>
<td>ON/OFF</td>
<td>SYSTem:POWer:LIMit:STATe</td>
<td>State</td>
</tr>
<tr>
<td></td>
<td>Power Limit</td>
<td>SYSTem:POWer:LIMit</td>
<td>Limit</td>
</tr>
</tbody>
</table>

**Offsets and Limits...**

**Receiver Leveling...** (VNAs with Pre-Sweep Mode, Point Mode, and Prior Sweep Mode)

<table>
<thead>
<tr>
<th>Controlled Source</th>
<th></th>
<th>SOURce:POWer:ALC[:MODE]:RECeiver:REFerence</th>
<th>ReferenceReceiver</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Leveling</td>
<td></td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver</td>
<td>State</td>
</tr>
<tr>
<td>Leveling Receiver</td>
<td></td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:REFerence</td>
<td>ReferenceReceiver</td>
</tr>
<tr>
<td>Leveling Type</td>
<td></td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:ACQuisition:MODE</td>
<td>None</td>
</tr>
<tr>
<td>Setting</td>
<td>Command</td>
<td>Value</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>Max Power</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:SAFE:MAX</td>
<td>PowerMax</td>
<td></td>
</tr>
<tr>
<td>Min Power</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:SAFE:MIN</td>
<td>PowerMin</td>
<td></td>
</tr>
<tr>
<td>Enable Safe Mode Leveling Using Max Step Size</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:SAFE</td>
<td>SafeMode</td>
<td></td>
</tr>
<tr>
<td>Max step size</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:SAFE:STEP</td>
<td>PowerStep</td>
<td></td>
</tr>
<tr>
<td>Update Source Power Calibration with Leveling Data</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:LSPC</td>
<td>LastLevelingAsSPC</td>
<td></td>
</tr>
<tr>
<td>Source ALC Hardware</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
<td></td>
</tr>
<tr>
<td>Leveling Tolerance</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:TOLERance</td>
<td>Tolerance</td>
<td></td>
</tr>
<tr>
<td>Leveling Max Iterations</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:ITERation:VALUE</td>
<td>IterationNumber</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:ITERation:ENABLE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Leveling IFBW</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:IFBW</td>
<td>LevelingIFBW</td>
<td></td>
</tr>
<tr>
<td>Leveling Receiver Frequency</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:FTYPE</td>
<td>FrequencyType</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------------------</td>
<td>---------------</td>
<td></td>
</tr>
</tbody>
</table>

7518
**CF_Power Commands - NF_NFX**

**Power Commands - Noise Figure Cold Source and Noise Figure Converters Measurement Classes**

Only the Main and Leveling & Offsets Power commands corresponding to the Noise Figure Source and Noise Figure Converters measurement classes are documented here. The Port Power and Attenuators commands are identical to the Port Power and Attenuator commands for the Standard measurement class and can be accessed by clicking on the softtab on the graphic below.

Click [here](#) to view links to Power commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main</th>
<th>Port Power</th>
<th>Leveling &amp; Offsets</th>
<th>Attenuators</th>
</tr>
</thead>
</table>

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Level</td>
<td></td>
<td>SOURce:POWer[:LEVEL][:IMMediate][:AMPLitude]</td>
<td>TestPortPower</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DISPlay:GUI:POWer:SPIN:RESolution</td>
<td>PowerSpinResolution</td>
</tr>
<tr>
<td>RF Power</td>
<td>ON/OFF</td>
<td>OUTPut[:STATE]</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td>Noise Source</td>
<td>ON/OFF</td>
<td>OUTPut:MANual:NOISe[:STATE]</td>
<td>NoiseSourceState</td>
</tr>
<tr>
<td>Power and Attenuators...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NF Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Noise Figure Measurement Class only)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
NFX Setup...
(Noise Figure Converters Measurement Class only)

### Leveling & Offsets Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>Port 1</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 2</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 3</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 4</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 1  Src2</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Source3</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

#### Slope

<table>
<thead>
<tr>
<th>ON/OFF</th>
<th>SOURce:POWer[:LEVEL]:SLOPe:STATE</th>
<th>PowerSlopeState</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power</td>
<td>SOURce:POWer[:LEVEL]:SLOPe</td>
<td>PowerSlope</td>
</tr>
</tbody>
</table>

#### Offset

<p>| SOURce:POWer:ALC[:MODE]:RECeiver:OFFSet | PowerOffset |</p>
<table>
<thead>
<tr>
<th>Limit</th>
<th>ON/OFF</th>
<th>SYSTem:POWer:LIMit:STATe</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Limit</td>
<td></td>
<td>SYSTem:POWer:LIMit</td>
<td>Limit</td>
</tr>
</tbody>
</table>

**Offsets and Limits...**

**ALC Hardware**

<table>
<thead>
<tr>
<th>Internal</th>
<th>SOURce:POWer:ALC[:MODE]</th>
<th>ALCLevelingMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Loop</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
</tbody>
</table>

**Receiver Leveling...**
CF_Power Commands - PN

Power Commands - Phase Noise Measurement Class

Only the Main Power commands corresponding to the Phase Noise measurement class are documented here. The Port Power and Attenuators commands are identical to the Port Power and Attenuator commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtab on the graphic below.

Click here to view links to Power commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main</th>
<th>Port Power</th>
<th>Attenuators</th>
</tr>
</thead>
</table>

**Main Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Level</td>
<td></td>
<td>SOURce:POWer[:LEVEL][:IMMediate][:AMPLitude]</td>
<td>TestPortPower</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DISPlay:GUI:POWer:SPIN:RESolution</td>
<td>PowerSpinResolution</td>
</tr>
<tr>
<td>RF Power</td>
<td>ON/OFF</td>
<td>OUTPUT[:STATe]</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td>Power and Attenuators</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phase Noise Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CF_Power Commands - SA

Power Commands - Spectrum Analyzer Measurement Class

Only the Main, Port Power, and Leveling & Offsets Power commands corresponding to the Spectrum Analyzer measurement class are documented here. The Attenuators commands are identical to the Attenuator commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtab on the graphic below.

Click here to view links to Power commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>RF Power</td>
</tr>
<tr>
<td>Power and Attenuators...</td>
</tr>
<tr>
<td>SA Setup...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Port Power Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>Select</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Port 3</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>Port 4</td>
</tr>
<tr>
<td>Port 1 Src2</td>
</tr>
<tr>
<td>Source3</td>
</tr>
</tbody>
</table>

### Power Level
- **SOURce:POWer[:LEVEL][:IMMediate][:AMPLitude]**
  - TestPortPower

### Source State
- **AUTO**
  - SOURce:POWer:MODE
  - SourcePortMode
- **ON**
  - SOURce:POWer:MODE
  - SourcePortMode
- **OFF**
  - SOURce:POWer:MODE
  - SourcePortMode

### Coupling
- **ON**
  - SOURce:POWer:COUPle
  - CouplePorts
- **OFF**
  - SOURce:POWer:COUPle
  - CouplePorts

### Leveling & Offsets Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>Port 1</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Port 2</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>Port 3</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Port 4</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Port 1 Src2</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Source3</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Slope</td>
<td>ON/OFF</td>
<td>SOURce:POWer[:LEVel]:SLOPe:STATe</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>PowerSlopeState</td>
<td></td>
</tr>
<tr>
<td>Power Slope</td>
<td>SOURce:POWer[:LEVel]:SLOPe</td>
<td>PowerSlope</td>
<td></td>
</tr>
<tr>
<td>Offset</td>
<td>SOURce:POWer:CORRection:OFFSet:MAGNitude</td>
<td>PowerOffset</td>
<td></td>
</tr>
<tr>
<td>Limit</td>
<td>ON/OFF</td>
<td>SYSTem:POWer:LIMit:STATe</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>State</td>
<td></td>
</tr>
<tr>
<td>Power Limit</td>
<td>SYSTem:POWer:LIMit</td>
<td>Limit</td>
<td></td>
</tr>
<tr>
<td>Offsets and Limits...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALC Hardware</td>
<td>Internal</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ALCLevelingMode</td>
<td></td>
</tr>
<tr>
<td>Open Loop</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
<td></td>
</tr>
</tbody>
</table>
CF_Power Commands - SMC_VMC

Power Commands - Scalar Mixer/Converter + Phase and Vector Mixer/Converter Measurement Classes

Only the Main, Port Power, and Leveling & Offsets Power commands corresponding to the Scalar Mixer/Converter + Phase and Vector Mixer/Converter measurement classes are documented here. The Attenuators commands are identical to the Attenuator commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtab on the graphic below.

Click here to view links to Power commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main</th>
<th>Port Power</th>
<th>Leveling &amp; Offsets</th>
<th>Attenuators</th>
</tr>
</thead>
</table>

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Level</td>
<td></td>
<td>SOURce:POWer[:LEVEL][:IMMediate][:AMPLitude]</td>
<td>TestPortPower</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DISPlay:GUI:POWer:SPIN:RESolution</td>
<td>PowerSpinResolution</td>
</tr>
<tr>
<td>RF Power</td>
<td>ON/OFF</td>
<td>OUTPut[:STATe]</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td>Power and Attenuators...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMC Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Scalar Mixer/Converter Measurement Class only)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VMC Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Vector Mixer/Converter Measurement Class only)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port Power Tab Commands</td>
<td>SCPI</td>
<td>COM</td>
<td></td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>-------------------------------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td><strong>Softkey</strong></td>
<td>Sub-item</td>
<td>SCPI</td>
<td></td>
</tr>
<tr>
<td><strong>Select</strong></td>
<td>Port 1</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port 2</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port 3</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port 4</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port 1</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Source3</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>Power Level</strong></td>
<td>SOURce:POWer[:LEVel][:IMMediate][:AMPLitude]</td>
<td>TestPortPower</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Source State</strong></td>
<td>AUTO</td>
<td>SOURce:POWer:MODE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SourcePortMode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ON</td>
<td>SOURce:POWer:MODE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SourcePortMode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>SOURce:POWer:MODE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SourcePortMode</td>
<td></td>
</tr>
<tr>
<td><strong>Coupling</strong></td>
<td>ON</td>
<td>SOURce:POWer:COUPlE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CouplePorts</td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>-------------</td>
<td>----------</td>
<td>------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Select</td>
<td>Port 1</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 2</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 3</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 4</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 1 Src2</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Source3</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Slope</td>
<td>ON/OFF</td>
<td>SOURce:POWer[:LEVEL]:SLOPe:STATe</td>
<td>PowerSlopeState</td>
</tr>
<tr>
<td></td>
<td>Power Slope</td>
<td>SOURce:POWer[:LEVEL]:SLOPe</td>
<td>PowerSlope</td>
</tr>
<tr>
<td>Offset</td>
<td></td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:OFFSet</td>
<td>PowerOffset</td>
</tr>
<tr>
<td>Limit</td>
<td>ON/OFF</td>
<td>SYSTem:POWer:LIMit:STATe</td>
<td>State</td>
</tr>
<tr>
<td>-------</td>
<td>--------</td>
<td>--------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Power Limit</td>
<td>SYSTem:POWer:LIMit</td>
<td>Limit</td>
<td></td>
</tr>
<tr>
<td>Offsets and Limits...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALC Hardware</td>
<td>Internal</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
<tr>
<td></td>
<td>Open Loop</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
<tr>
<td>Receiver Leveling...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## CF_Power Commands - Standard

### Power Commands - Standard Measurement Class

Click [here](#) to view links to Power commands for all Measurement Classes.

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>SOURce:POWer[:LEVel][:IMMediate][:AMPLitude]</td>
<td>TestPortPower</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DISPlay:GUI:POWer:SPIN:RESolution</td>
<td>PowerSpinResolution</td>
</tr>
<tr>
<td>RF Power</td>
<td>ON/OFF</td>
<td>OUTPut[:STATe]</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td>Start Power</td>
<td></td>
<td>SOURce:POWer:PORT:STARt</td>
<td>StartPower</td>
</tr>
<tr>
<td>Stop Power</td>
<td></td>
<td>SOURce:POWer:PORT:STOP</td>
<td>StopPower</td>
</tr>
<tr>
<td>Power and</td>
<td>Power ON (All</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Attenuators...</td>
<td>Channels)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port Powers</td>
<td>SOURce:POWer:COUPle</td>
<td>CouplePorts</td>
</tr>
<tr>
<td></td>
<td>Coupled</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Name</td>
<td>SOURce:POWer:MODE</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td></td>
<td>State</td>
<td>SOURce:POWer:MODE</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td>Port Power</td>
<td>SOURcE:POWer[:LEVel][:IMMediate][:AMPLitude]</td>
<td>TestPortPower</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>Start Power</td>
<td>SOURcE:POWer:PORT:STARt</td>
<td>StartPower</td>
<td></td>
</tr>
<tr>
<td>Stop Power</td>
<td>SOURcE:POWer:PORT:STOP</td>
<td>StopPower</td>
<td></td>
</tr>
<tr>
<td>Leveling Mode</td>
<td>SOURcE:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
<td></td>
</tr>
<tr>
<td>Channel Power Slope</td>
<td>SOURcE:POWer[:LEVel]:SLOPe:STATe SOURcE:POWer[:LEVel]:SLOPe</td>
<td>PowerSlopeState PowerSlope</td>
<td></td>
</tr>
<tr>
<td>Offsets and Limits...</td>
<td>Power Limit - State</td>
<td>SYSTem:POWer:LIMit:STATe</td>
<td>State</td>
</tr>
<tr>
<td></td>
<td>Power Limit - Limit</td>
<td>SYSTem:POWer:LIMit</td>
<td>Limit</td>
</tr>
<tr>
<td>Source Power</td>
<td>SOURcE:POWer[:LEVel][:IMMediate][:AMPLitude]</td>
<td>TestPortPower</td>
<td></td>
</tr>
<tr>
<td>Power Offset</td>
<td>SOURcE:POWer:ALC[:MODE]:RECeiver:OFFSet</td>
<td>PowerOffset</td>
<td></td>
</tr>
<tr>
<td>Port Power</td>
<td>SOURcE:POWer:PORT:STARt SOURcE:POWer:PORT:STOP</td>
<td>StartPower StopPower</td>
<td></td>
</tr>
</tbody>
</table>

Receiver Leveling... (VNAs with Pre-Sweep Mode, Point Mode, and Prior Sweep Mode)
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controlled Source</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:REFerence</td>
<td>ReferenceReceiver</td>
</tr>
<tr>
<td>Enable Leveling</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver</td>
<td>State</td>
</tr>
<tr>
<td>Leveling Receiver</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:REFerence</td>
<td>ReferenceReceiver</td>
</tr>
<tr>
<td>Leveling Type</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:ACQuisition:MODE</td>
<td>None</td>
</tr>
<tr>
<td>Max Power</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:SAFE:MAX</td>
<td>PowerMax</td>
</tr>
<tr>
<td>Min Power</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:SAFE:MIN</td>
<td>PowerMin</td>
</tr>
<tr>
<td>Enable Safe Mode Leveling Using Max Step Size</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:SAFE</td>
<td>SafeMode</td>
</tr>
<tr>
<td>Max step size</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:SAFE:STEP</td>
<td>PowerStep</td>
</tr>
<tr>
<td>Update Source Power Calibration with Leveling Data</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:LSPC</td>
<td>LastLevelingAsSPC</td>
</tr>
</tbody>
</table>

7533
<table>
<thead>
<tr>
<th>Source ALC Hardware</th>
<th>SOURc:POWer:ALC[:MODE]</th>
<th>ALCLevelingMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leveling Tolerance</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:TOLerance</td>
<td>Tolerance</td>
</tr>
<tr>
<td>Leveling Max Iterations</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:ITERation:VALue</td>
<td>IterationNumber</td>
</tr>
<tr>
<td></td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:ITERation:ENABle</td>
<td>None</td>
</tr>
<tr>
<td>Leveling Receiver Frequency</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:FTYPE</td>
<td>FrequencyType</td>
</tr>
<tr>
<td>Leveling IFBW</td>
<td>SOURc:POWer:ALC[:MODE]:RECeiver:IFBW</td>
<td>LevelingIFBW</td>
</tr>
<tr>
<td>Path Configuration...</td>
<td>SENSE:PATH:CONFig:ELEMen[[:STATe]]</td>
<td>Element</td>
</tr>
</tbody>
</table>

**Port Power Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>Port 1</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 2</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Port</td>
<td>Power Level</td>
<td>Start Power</td>
<td>Stop Power</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td>-------------</td>
<td>------------</td>
</tr>
<tr>
<td>Port 3</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Port 4</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Port 1 Src2</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Source3</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Power Level

- **SOURce:POWer[:L]EVel[:IMM]ediate[:AMPL]itude**
- TestPortPower

### Start Power

- **SOURce:POWer:PORT:STARt**
- StartPower

### Stop Power

- **SOURce:POWer:PORT:STOP**
- StopPower

### Source State

- **SOURce:POWer:MODE**
- SourcePortMode

### Coupling

- **SOURce:POWer:COUPle**
- CouplePorts

---

**Leveling & Offsets Tab Commands**
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>Port 1</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 2</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 3</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 4</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Port 1 Src2</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Source3</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Slope</td>
<td>ON/OFF</td>
<td>SOURc:POWer[:LEVel]:SLOPe:STATe</td>
<td>PowerSlopeState</td>
</tr>
<tr>
<td></td>
<td>Power Slope</td>
<td>SOURc:POWer[:LEVel]:SLOPe</td>
<td>PowerSlope</td>
</tr>
<tr>
<td>Offset</td>
<td></td>
<td>SOURc:POWer:CORRection:OFFSet:MA GNitude</td>
<td>PowerOffset</td>
</tr>
<tr>
<td>Limit</td>
<td>ON/OFF</td>
<td>SYSTem:POWer:LIMit:STATe</td>
<td>State</td>
</tr>
<tr>
<td></td>
<td>Power Limit</td>
<td>SYSTem:POWer:LIMit</td>
<td>Limit</td>
</tr>
<tr>
<td></td>
<td>Power Limit - State</td>
<td>SYSTem:POWer:LIMit:STATe</td>
<td>State</td>
</tr>
<tr>
<td>Offsets and Limits...</td>
<td>Power Limit - Limit</td>
<td>SYSTem:POWer:LIMit</td>
<td>Limit</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------</td>
<td>---------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Source Power</td>
<td>SOURce:POWer[:LEVel][:IMMediate][:AMPLitude]</td>
<td>TestPortPower</td>
<td></td>
</tr>
<tr>
<td>Power Offset</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:OFFSet</td>
<td>PowerOffset</td>
<td></td>
</tr>
<tr>
<td>Port Power</td>
<td>SOURce:POWer:PORT:STARt SOURce:POWer:PORT:STOP</td>
<td>StartPower StopPower</td>
<td></td>
</tr>
<tr>
<td>ALC Hardware</td>
<td>Internal</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
<tr>
<td></td>
<td>Open Loop</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
<tr>
<td>Receiver Leveling...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Attenuators Tab Commands</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>Source Port 1</td>
<td></td>
<td>SOURce:POWer1:ATTenuation</td>
<td>Attenuator</td>
</tr>
<tr>
<td>Source Port 2</td>
<td></td>
<td>SOURce:POWer2:ATTenuation</td>
<td>Attenuator</td>
</tr>
<tr>
<td>Source Port 3</td>
<td></td>
<td>SOURce:POWer3:ATTenuation</td>
<td>Attenuator</td>
</tr>
<tr>
<td>Source Port 4</td>
<td>SOURce:POWer4:ATTenuation</td>
<td>Attenuator</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------</td>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>Receiver A</td>
<td>SENSE:POWer:ATTenuation</td>
<td>ReceiverAttenuator</td>
<td></td>
</tr>
<tr>
<td>Receiver B</td>
<td>SENSE:POWer:ATTenuation</td>
<td>ReceiverAttenuator</td>
<td></td>
</tr>
<tr>
<td>Receiver C</td>
<td>SENSE:POWer:ATTenuation</td>
<td>ReceiverAttenuator</td>
<td></td>
</tr>
<tr>
<td>Receiver D</td>
<td>SENSE:POWer:ATTenuation</td>
<td>ReceiverAttenuator</td>
<td></td>
</tr>
</tbody>
</table>
# CF_Preset Commands

## Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preset</td>
<td></td>
<td>SYSTem:PRESet</td>
<td>Preset</td>
</tr>
<tr>
<td>User Preset...</td>
<td>ON/OFF</td>
<td>SYSTem:PREferences:ITEM:POWer[STATE]</td>
<td>PresetPowerState</td>
</tr>
<tr>
<td>Confirm Preset</td>
<td></td>
<td>SYSTem:PREferences:ITEM:PRESet:CONFirm</td>
<td>None</td>
</tr>
</tbody>
</table>
# CF_Save-Recall Commands

## Recall Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recall State</td>
<td></td>
<td>MME:LOAD</td>
<td>Recall</td>
</tr>
<tr>
<td>Recall State</td>
<td></td>
<td>MME:LOAD</td>
<td>Recall</td>
</tr>
<tr>
<td>Recall State...</td>
<td></td>
<td>MME:LOAD</td>
<td>Recall</td>
</tr>
<tr>
<td>Recall Register</td>
<td></td>
<td>MME:LOAD</td>
<td>Recall</td>
</tr>
<tr>
<td>Recall Calset...</td>
<td></td>
<td>MME:LOAD</td>
<td>Recall</td>
</tr>
<tr>
<td>Recall Data...</td>
<td></td>
<td>MME:LOAD</td>
<td>Recall</td>
</tr>
<tr>
<td>Recall Order</td>
<td>NAME</td>
<td>MME:LOAD</td>
<td>Recall</td>
</tr>
<tr>
<td></td>
<td>RECENT</td>
<td>MME:LOAD</td>
<td>Recall</td>
</tr>
</tbody>
</table>

## Save State Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save State</td>
<td></td>
<td>MME:STORe</td>
<td>Save</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Action</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>Auto Save</td>
<td>MMEMory:STORe</td>
<td>Save</td>
<td></td>
</tr>
<tr>
<td>Save State As...</td>
<td>MMEMory:STORe</td>
<td>Save</td>
<td></td>
</tr>
<tr>
<td>Save Register</td>
<td>MMEMory:STORe</td>
<td>Save</td>
<td></td>
</tr>
<tr>
<td>Save Type</td>
<td>MMEMory:STORe</td>
<td>Save</td>
<td></td>
</tr>
<tr>
<td>State</td>
<td>MMEMory:STORe</td>
<td>Save</td>
<td></td>
</tr>
<tr>
<td>State + Cal Data</td>
<td>MMEMory:STORe</td>
<td>Save</td>
<td></td>
</tr>
<tr>
<td>State + Cal Link</td>
<td>MMEMory:STORe</td>
<td>Save</td>
<td></td>
</tr>
<tr>
<td>Delete State</td>
<td>MMEMory:STORe</td>
<td>Save</td>
<td></td>
</tr>
<tr>
<td>Save Other Tab Commands</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>Save Calset...</td>
<td>SENSE:CORRection:CSET:SAVE</td>
<td>Save</td>
<td></td>
</tr>
<tr>
<td>Save Data...</td>
<td>MMEMory:STORe</td>
<td>Save</td>
<td></td>
</tr>
<tr>
<td>Save Screen...</td>
<td>MMEMory:STORe</td>
<td>Save</td>
<td></td>
</tr>
<tr>
<td>Save User Preset...</td>
<td>MMEMory:STORe</td>
<td>Save</td>
<td></td>
</tr>
<tr>
<td>Manage Files...</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
## CF_Scale Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autoscale</td>
<td></td>
<td>DISPlay:MEASure:Y[:SCALe]:AUTO</td>
<td>Autoscale</td>
</tr>
<tr>
<td>Autoscale All</td>
<td></td>
<td>DISPlay:WINDow:Y:AUTO</td>
<td>Autoscale</td>
</tr>
<tr>
<td>Scale</td>
<td></td>
<td>DISPlay:MEASure:Y[:SCALe]:PDIVision</td>
<td>YScale</td>
</tr>
<tr>
<td>Reference Level</td>
<td></td>
<td>DISPlay:MEASure:Y[:SCALe]:RLEVel</td>
<td>ReferenceValue</td>
</tr>
<tr>
<td>Reference Position</td>
<td></td>
<td>DISPlay:MEASure:Y[:SCALe]:RPOSITION</td>
<td>ReferencePosition</td>
</tr>
<tr>
<td>Y-Axis Spacing</td>
<td></td>
<td>DISPlay:WINDow:TRACe:Y:SPACING</td>
<td>None</td>
</tr>
<tr>
<td>Top</td>
<td></td>
<td>DISPlay:WINDow:TRACe:Y:SCALE:TOP</td>
<td>None</td>
</tr>
<tr>
<td>Bottom</td>
<td></td>
<td>DISPlay:WINDow:TRACe:Y:SCALE:BOTTom</td>
<td>None</td>
</tr>
<tr>
<td>Ref Y Level</td>
<td></td>
<td>DISPlay:WINDow:TRACe:Y:SCALE:RLEVel</td>
<td>None</td>
</tr>
<tr>
<td>Ref Y Position</td>
<td></td>
<td>DISPlay:WINDow:TRACe:Y:SCALE:RPOSITION</td>
<td>None</td>
</tr>
<tr>
<td>Ref X Level</td>
<td>Display: Window: Trace: X: Scale: Level</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------------------------</td>
<td>----------</td>
<td></td>
</tr>
<tr>
<td>Ref X Position</td>
<td>Display: Window: Trace: X: Scale: Position</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Scale Coupling...</td>
<td>Off</td>
<td>ScaleCouplingMethod</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Window</td>
<td>ScaleCouplingMethod</td>
<td></td>
</tr>
<tr>
<td></td>
<td>All: couple between all selected windows</td>
<td>ScaleCouplingMethod</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Selected Windows</td>
<td>ScaleCouplingState</td>
<td></td>
</tr>
</tbody>
</table>

**Electrical Delay Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay Time</td>
<td></td>
<td>CALCulate: MEASure: CORRection: EDELay[:TIME]</td>
<td>ElectricalDelay</td>
</tr>
<tr>
<td>Delay Distance</td>
<td></td>
<td>CALCulate: MEASure: CORRection: EDELay: DISTance</td>
<td>ElecDistanceDelay</td>
</tr>
<tr>
<td>Distance Units</td>
<td>Meters</td>
<td>CALCulate: MEASure: CORRection: EDELay: UNIT</td>
<td>ElecDistanceDelayUnit</td>
</tr>
<tr>
<td></td>
<td>Feet</td>
<td>CALCulate: MEASure: CORRection: EDELay: UNIT</td>
<td>ElecDistanceDelayUnit</td>
</tr>
<tr>
<td></td>
<td>Inches</td>
<td>CALCulate: MEASure: CORRection: EDELay: UNIT</td>
<td>ElecDistanceDelayUnit</td>
</tr>
<tr>
<td></td>
<td>SCPI</td>
<td>COM</td>
<td></td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------------------------------</td>
<td>--------------------------</td>
<td></td>
</tr>
<tr>
<td>Velocity Factor</td>
<td>SENSE:CORRection:RVELocity:COAX</td>
<td>PortVelocityFactor</td>
<td></td>
</tr>
<tr>
<td>Media</td>
<td>CALCulate:MEASure:CORRection:EDELay:MEDium</td>
<td>PortMedium</td>
<td></td>
</tr>
<tr>
<td>Wavegd Cutoff</td>
<td>CALCulate:MEASure:CORRection:EDELay:WGCutoff</td>
<td>PortWGCutoffFreq</td>
<td></td>
</tr>
<tr>
<td><strong>Constants Tab Commands</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sub-item</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCPI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System Z0</td>
<td>SENSE:CORRection:IMPedance:INPut:MAGNitude</td>
<td>SystemImpedanceZ0</td>
<td></td>
</tr>
<tr>
<td>Phase Offset</td>
<td>CALCulate:MEASure:OFFSet:PHASE</td>
<td>PhaseOffset</td>
<td></td>
</tr>
<tr>
<td>Mag Offset</td>
<td>CALCulate:MEASure:OFFSet:MAGNitude</td>
<td>MagnitudeOffset</td>
<td></td>
</tr>
<tr>
<td>Mag Slope</td>
<td>CALCulate:MEASure:OFFSet:MAGNitude:SOPe</td>
<td>MagnitudeSlopeOffset</td>
<td></td>
</tr>
</tbody>
</table>
## CF_Search Commands

<table>
<thead>
<tr>
<th>Main</th>
<th>Peak</th>
<th>Target</th>
<th>Multi Peak &amp; Target</th>
<th>Bandwidth &amp; Notch</th>
<th>Comp &amp; Sat</th>
<th>Normal Op Pt</th>
<th>Distortion</th>
<th>Spurious</th>
</tr>
</thead>
</table>

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Search</td>
<td></td>
<td><code>CALCulate:MEASure:MARKer:FUNCtion:EXECute</code></td>
<td><code>SearchMax</code></td>
</tr>
<tr>
<td>Min Search</td>
<td></td>
<td><code>CALCulate:MEASure:MARKer:FUNCtion:EXECute</code></td>
<td><code>SearchMin</code></td>
</tr>
<tr>
<td>Domain Start</td>
<td></td>
<td><code>CALCulate:MEASure:MARKer:FUNCtion:DOMain:USER:STARt</code></td>
<td><code>UserRangeMin</code></td>
</tr>
<tr>
<td>Tracking</td>
<td>Off</td>
<td><code>CALCulate:MEASure:MARKer:FUNCtion:TRACKing</code></td>
<td><code>Tracking SearchFunction</code></td>
</tr>
<tr>
<td>Max</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TRACKing</td>
<td>Tracking</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>--------------------------------------------</td>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SearchFunction</td>
<td></td>
</tr>
<tr>
<td>Min</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TRACKing</td>
<td>Tracking</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SearchFunction</td>
<td></td>
</tr>
<tr>
<td>Peak</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TRACKing</td>
<td>Tracking</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SearchFunction</td>
<td></td>
</tr>
<tr>
<td>Target</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TRACKing</td>
<td>Tracking</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SearchFunction</td>
<td></td>
</tr>
<tr>
<td>Multi Peak</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TRACKing</td>
<td>Tracking</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SearchFunction</td>
<td></td>
</tr>
<tr>
<td>Multi Target</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TRACKing</td>
<td>Tracking</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SearchFunction</td>
<td></td>
</tr>
<tr>
<td>Compression</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TRACKing</td>
<td>Tracking</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SearchFunction</td>
<td></td>
</tr>
</tbody>
</table>

**Peak Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peak</td>
<td>Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:EXECute</td>
<td>SearchFunction</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Target Tab Commands</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>---------------------</td>
<td></td>
</tr>
<tr>
<td>Peak Right &gt;&gt; Search</td>
<td><code>CALCulate:MEASure:MARKer:FUNCtion:EXECute</code></td>
<td><strong>SearchFunction</strong></td>
<td></td>
</tr>
<tr>
<td>&lt;&lt; Peak Left Search</td>
<td><code>CALCulate:MEASure:MARKer:FUNCtion:EXECute</code></td>
<td><strong>SearchFunction</strong></td>
<td></td>
</tr>
<tr>
<td>Next Peak Search</td>
<td><code>CALCulate:MEASure:MARKer:FUNCtion:EXECute</code></td>
<td><strong>SearchFunction</strong></td>
<td></td>
</tr>
<tr>
<td>Threshold</td>
<td><code>CALCulate:MEASure:MARKer:FUNCtion:PEAK:THReshold</code></td>
<td><strong>PeakThreshold</strong></td>
<td></td>
</tr>
<tr>
<td>Excursion</td>
<td><code>CALCulate:MEASure:MARKer:FUNCtion:PEAK:EXCursion</code></td>
<td><strong>PeakExcursion</strong></td>
<td></td>
</tr>
<tr>
<td>Peak Polarity</td>
<td><strong>Positive</strong></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Negative</strong></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Both</strong></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Tracking</td>
<td><strong>None</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Target Search</strong></td>
<td></td>
<td><code>CALCulate:MEASure:MARKer:FUNCtion:EXECute</code></td>
<td><strong>SearchFunction</strong></td>
</tr>
<tr>
<td>Target Right &gt;&gt; Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:EXECute</td>
<td>SearchFunction</td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------------------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>&lt;&lt; Target Left Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:EXECute</td>
<td>SearchFunction</td>
<td></td>
</tr>
<tr>
<td>Target Value</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TARGet:VALUE</td>
<td>TargetValue</td>
<td></td>
</tr>
<tr>
<td>Transition</td>
<td>Positive</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TARGet:TRANSition</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Negative</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TARGet:TRANSition</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Both</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:TARGet:TRANSition</td>
<td>None</td>
</tr>
</tbody>
</table>

| Tracking               | Multi Peak & Target Tab Commands |

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Multi Peak Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:MULTi:EXECute</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Peak Threshold</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:MULTi:PEAK:THreshhold</td>
<td>None</td>
</tr>
<tr>
<td>Feature</td>
<td>Setting</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------</td>
<td>--------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Peak Excursion</td>
<td></td>
<td>CALCulate:MEASure:MARKer:FUNCTION:MULTi:PEAK:EXCu rsion</td>
<td>None</td>
</tr>
<tr>
<td>Peak Polarity</td>
<td>Positive</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:MULTi:PEAK:POLar ity</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Negative</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:MULTi:PEAK:POLar ity</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Both</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:MULTi:PEAK:POLar ity</td>
<td>None</td>
</tr>
<tr>
<td>Multi Target Search</td>
<td></td>
<td>CALCulate:MEASure:MARKer:FUNCTION:MULTi:EXECute</td>
<td>None</td>
</tr>
<tr>
<td>Target Value</td>
<td></td>
<td>CALCulate:MEASure:MARKer:FUNCTION:MULTi:TARGET[:VAlue]</td>
<td>None</td>
</tr>
<tr>
<td>Transition</td>
<td>Positive</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:MULTi:TARGET:TRANSition</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Negative</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:MULTi:TARGET:TRANSition</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Both</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:MULTi:TARGET:TRANSition</td>
<td>None</td>
</tr>
<tr>
<td>Tracking</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwidth &amp; Notch Tab Commands</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>Bandwidth Search</td>
<td>On/Off</td>
<td>CALCulate:MEASure:MARKer:BWIDth[:STATe]</td>
<td>SearchFilterBandwidth</td>
</tr>
<tr>
<td>BW Ref To Marker</td>
<td>CALCulate:MEASure:MARKer:BWIDth:REF</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Peak</td>
<td>CALCulate:MEASure:MARKer:BWIDth:REF</td>
<td>None</td>
</tr>
<tr>
<td>BW Level</td>
<td>CALCulate:MEASure:MARKer:BWIDth:THReshold</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Notch Search On/Off</td>
<td>CALCulate:MEASure:MARKer:NOTCh[:STATe]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Notch Ref To Marker</td>
<td>CALCulate:MEASure:MARKer:NOTCh:REF</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Peak</td>
<td>CALCulate:MEASure:MARKer:NOTCh:REF</td>
<td>None</td>
</tr>
<tr>
<td>Notch Level</td>
<td>CALCulate:MEASure:MARKer:NOTCh:THReshold</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Tracking</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Comp &amp; Sat Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
</tbody>
</table>

7551
<table>
<thead>
<tr>
<th>Feature</th>
<th>On/Off</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compressi on Search SearchFunction</td>
<td></td>
<td>CALCulate:MEASure:MARKer:FUNCtion:COMPression[:STATe]</td>
<td>SearchFunction</td>
</tr>
<tr>
<td>Comp Level</td>
<td></td>
<td>CALCulate:MEASure:MARKer:FUNCtion:COMPression:LEVEL</td>
<td>CompressionLevel</td>
</tr>
<tr>
<td>Saturation Search SearchPowerSaturation</td>
<td></td>
<td>CALCulate:MEASure:MARKer:PSATuration[:STATe]</td>
<td>SearchPowerSaturation</td>
</tr>
<tr>
<td>Pmax Backoff</td>
<td></td>
<td>CALCulate:MEASure:MARKer:PSATuration:BACKoff</td>
<td>SearchPowerSaturation PMaxIn</td>
</tr>
</tbody>
</table>

### Normal Op Pt Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backoff</td>
<td></td>
<td>CALCulate:MEASure:MARKer:PNOP:BACKoff</td>
<td>BackOff</td>
</tr>
<tr>
<td>Pin Offset</td>
<td></td>
<td>CALCulate:MEASure:MARKer:PNOP:POFFset</td>
<td>PinOffset</td>
</tr>
</tbody>
</table>

### Distortion Tab Commands (Modulation Distortion Measurement Class Only)
<table>
<thead>
<tr>
<th>Tab</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACPR Search</td>
<td>CALCulate:MEASure:SA:MARKer:SEARch:ACPR</td>
<td>None</td>
</tr>
<tr>
<td>NPR Search</td>
<td>CALCulate:MEASure:SA:MARKer:SEARch:NPR</td>
<td>None</td>
</tr>
<tr>
<td>Occupied BW</td>
<td>CALCulate:MEASure:SA:MARKer:OCCBand:STATe</td>
<td>None</td>
</tr>
<tr>
<td>OBW Percent</td>
<td>CALCulate:MEASure:SA:MARKer:OCCBand:PERCent</td>
<td>None</td>
</tr>
<tr>
<td>Spurious Tab Commands (Phase Noise Measurement Class Only)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spurious Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:EXECute SPURious</td>
<td>None</td>
</tr>
<tr>
<td>Spurious Right &gt;&gt; Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:EXECute RSPurious</td>
<td>None</td>
</tr>
<tr>
<td>&lt;&lt; Spurious Left Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:EXECute LSPurious</td>
<td>None</td>
</tr>
<tr>
<td>Multi Spurious Search</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:MULTi:EXECute SPURious</td>
<td>None</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Spur Sensibility</td>
<td>CALCulate:MEASure:PN:SPURious:SENSibility</td>
<td>None</td>
</tr>
<tr>
<td>Tracking</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>
Only the Main Setup commands corresponding to the Differential I/Q measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIQ Setup...</td>
<td>Frequency Range</td>
<td>SENSE:DIQ:FREQuency:RANGe:ADD</td>
<td>AddRange</td>
</tr>
<tr>
<td></td>
<td>Settings</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Frequency</td>
<td>SENSE:DIQ:FREQuency:RANGe:Start</td>
<td>RangeStartFrequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SENSE:DIQ:FREQuency:RANGe:STOP</td>
<td>RangeStopFrequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SENSE:DIQ:FREQuency:RANGe:IFBW</td>
<td>RangIFBW</td>
</tr>
<tr>
<td></td>
<td>Coupling</td>
<td>SENSE:DIQ:FREQuency:RANGe:COUPle:STATe</td>
<td>RangeCoupleState</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SENSE:DIQ:FREQuency:RANGe:COUPle:ID</td>
<td>RangeCoupled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SENSE:DIQ:FREQuency:RANGe:COUPle:OFFSet</td>
<td>RangeOffset</td>
</tr>
<tr>
<td>Source Configuration</td>
<td>Source State</td>
<td>Frequency Range</td>
<td>Ext Source Port</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------</td>
<td>-----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RangeMultiplier</td>
<td>SOURce:PHAsE:EXTernal:PORT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RangeDivisor</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DeleteRange</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSE:DIQ:FREQuency:RANGe:COUPle:UCONvert</td>
<td>PortAttenuator</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:DIQ:FREQuency:RANGe:COUPle:MULTiplier</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:DIQ:FREQuency:RANGe:COUPle:DIVisor</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source State</th>
<th>Frequency Range</th>
<th>Ext Source Port</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>SourceState</td>
<td>SourceRange</td>
<td>None</td>
<td>PowerSweepState</td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td>PortStartPower</td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td>PortStopPower</td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td>PortLevelingMode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td>PortAttenuator</td>
</tr>
<tr>
<td>Category</td>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------</td>
<td>-----------------------</td>
<td></td>
</tr>
<tr>
<td>Auto Range</td>
<td>SENSe:DIQ:PORT:POWer:ATTenuation:AUTO</td>
<td>AutoRangeState</td>
<td></td>
</tr>
<tr>
<td>Phase</td>
<td>SENSe:DIQ:PORT:PHASe:STATe</td>
<td>PortPhaseState</td>
<td></td>
</tr>
<tr>
<td>Phase</td>
<td>SENSe:DIQ:PORT:PHASe:SWEep</td>
<td>PhaseSweepState</td>
<td></td>
</tr>
<tr>
<td>Phase</td>
<td>SENSe:DIQ:PORT:PHASe:STARt</td>
<td>PortPhaseStart</td>
<td></td>
</tr>
<tr>
<td>Phase</td>
<td>SENSe:DIQ:PORT:PHASe:STOP</td>
<td>PortPhaseStop</td>
<td></td>
</tr>
<tr>
<td>Phase</td>
<td>SENSe:DIQ:PORT:PHASe:REFerence</td>
<td>PortReference</td>
<td></td>
</tr>
<tr>
<td>Phase</td>
<td>SENSe:DIQ:PORT:PHASe:PARameter</td>
<td>PortPhaseParameter</td>
<td></td>
</tr>
<tr>
<td>Match</td>
<td>SOURce:PHASe:CONTrol:TOLerance</td>
<td>PhaseTolerance</td>
<td></td>
</tr>
<tr>
<td>Correction</td>
<td>SOURce:PHASe:CONTrol:ITERation</td>
<td>PhasIterationNumber</td>
<td></td>
</tr>
<tr>
<td>Match</td>
<td>SENSe:DIQ:PORT:MATCh:STATe</td>
<td>MatchState</td>
<td></td>
</tr>
<tr>
<td>Match</td>
<td>SENSe:DIQ:PORT:MATCh:TRECeiver</td>
<td>MatchTestReceiver</td>
<td></td>
</tr>
<tr>
<td>Match</td>
<td>SENSe:DIQ:PORT:MATCh:RRECeiver</td>
<td>MatchRefReceiver</td>
<td></td>
</tr>
<tr>
<td>Match</td>
<td>SENSe:DIQ:PORT:MATCh:RANGe</td>
<td>MatchFrequencyRange</td>
<td></td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------</td>
<td>--------------------------</td>
<td></td>
</tr>
<tr>
<td>Quick Start...</td>
<td>S-Param</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
</tr>
<tr>
<td>Balanced</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BalSMeasurement</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>BBalMeasurement</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SBalMeasurement</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SSBMMeasurement</td>
<td></td>
</tr>
<tr>
<td>Other</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
<td></td>
</tr>
</tbody>
</table>
The Setup softkeys vary depending on which measurement class is currently active. Click on a measurement class link below to view the corresponding softkeys.

**Standard**
- Active Hot Parameters
- Gain Compression
- Differential I/Q
- IM Spectrum
- Swept IMD

**Modulation Distortion and Modulation Distortion Converters**
- Noise Figure Cold Source
- Phase Noise
- Spectrum Analyzer
- Gain Compression Converters
- IM Spectrum Converters
- Swept IMD Converters
- Noise Figure Converters
- Scalar Mixer/Converter + Phase
- Vector Mixer/Converter
- TDR (Time Domain Reflectometry)
CF_Setup Commands - GCA

Setup Commands - Gain Compression Measurement Class

Only the Main and Layout Setup commands corresponding to the Gain Compression measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>GCA Setup...</td>
</tr>
<tr>
<td>Sweep Type</td>
</tr>
<tr>
<td>Linear Sweep</td>
</tr>
<tr>
<td>Log Sweep</td>
</tr>
<tr>
<td>Segment Sweep</td>
</tr>
<tr>
<td>Data Acquisition Mode</td>
</tr>
<tr>
<td>SMART Sweep</td>
</tr>
<tr>
<td><strong>Sweep</strong></td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td><strong>Sweep Power Per Frequency (2D)</strong></td>
</tr>
<tr>
<td><strong>Sweep Frequency Per Power (2D)</strong></td>
</tr>
<tr>
<td><strong>Sweep Settings</strong></td>
</tr>
<tr>
<td><strong>Number of Points</strong></td>
</tr>
<tr>
<td><strong>IF Bandwidth</strong></td>
</tr>
<tr>
<td><strong>Start</strong></td>
</tr>
<tr>
<td><strong>Stop</strong></td>
</tr>
<tr>
<td><strong>Center</strong></td>
</tr>
<tr>
<td><strong>Span</strong></td>
</tr>
<tr>
<td><strong>Power</strong></td>
</tr>
<tr>
<td><strong>Power On (All Channels)</strong></td>
</tr>
<tr>
<td>DUT Input Port</td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td><strong>Input Port</strong></td>
</tr>
<tr>
<td><strong>Linear Input Power</strong></td>
</tr>
<tr>
<td><strong>Source Leveling Mode</strong></td>
</tr>
<tr>
<td><strong>DUT Output Port</strong></td>
</tr>
<tr>
<td><strong>Output Port</strong></td>
</tr>
<tr>
<td><strong>Reverse Power</strong></td>
</tr>
<tr>
<td><strong>Source Leveling Mode</strong></td>
</tr>
<tr>
<td><strong>Power Sweep</strong></td>
</tr>
<tr>
<td><strong>Start (Min) Power</strong></td>
</tr>
<tr>
<td><strong>Stop (Max) Power</strong></td>
</tr>
<tr>
<td>Power Points</td>
</tr>
<tr>
<td>-------------</td>
</tr>
<tr>
<td>Power Step</td>
</tr>
<tr>
<td>Path</td>
</tr>
</tbody>
</table>

**Compression**

<table>
<thead>
<tr>
<th>Compression Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:GCSetup:COMPression:ALGorithm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Compression from Linear Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:GCSetup:COMPression:ALGorithm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Compression from Max Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:GCSetup:COMPression:ALGorithm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Compression from Back Off</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:GCSetup:COMPression:ALGorithm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>X/Y Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:GCSetup:COMPression:ALGorithm</td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>----------------------------------</td>
</tr>
<tr>
<td>Compression from Saturation</td>
</tr>
<tr>
<td>Magnitude Level</td>
</tr>
<tr>
<td>Back Off</td>
</tr>
<tr>
<td>Delta X</td>
</tr>
<tr>
<td>Delta Y</td>
</tr>
<tr>
<td>From Max Pout</td>
</tr>
<tr>
<td>Magnitude Only</td>
</tr>
<tr>
<td>Phase Only</td>
</tr>
<tr>
<td>Magnitude or Phase</td>
</tr>
<tr>
<td>Phase Level</td>
</tr>
<tr>
<td>Phase Details...</td>
</tr>
<tr>
<td>Setting</td>
</tr>
<tr>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>Compute Linear Power from Percent of Span</td>
</tr>
<tr>
<td>Force Source Power Out Port 1</td>
</tr>
<tr>
<td><strong>SMART Sweep</strong></td>
</tr>
<tr>
<td>Tolerance</td>
</tr>
<tr>
<td>Maximum Iterations</td>
</tr>
<tr>
<td>Show Iterations</td>
</tr>
<tr>
<td>Read DC at Compression Point</td>
</tr>
<tr>
<td>Safe Mode...</td>
</tr>
<tr>
<td>Safe Mode</td>
</tr>
<tr>
<td>Coarse Increment</td>
</tr>
<tr>
<td>Feature</td>
</tr>
<tr>
<td>--------------------------------</td>
</tr>
<tr>
<td>Fine Increment</td>
</tr>
<tr>
<td>Fine Threshold</td>
</tr>
<tr>
<td>Max Output Power</td>
</tr>
<tr>
<td>2D Sweep</td>
</tr>
<tr>
<td>Compression Point</td>
</tr>
<tr>
<td>Interpolation</td>
</tr>
<tr>
<td>End of Sweep Condition</td>
</tr>
<tr>
<td>Settling Time</td>
</tr>
<tr>
<td>Measure Class...</td>
</tr>
<tr>
<td>Quick Start...</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Balanced</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
## Layout Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>New Trace</strong></td>
<td></td>
<td>DISPLAY:WINDow:TRACe[:STATE]</td>
<td>View</td>
</tr>
<tr>
<td><strong>New Channel</strong></td>
<td>None</td>
<td>chans.Add</td>
<td></td>
</tr>
<tr>
<td><strong>New Window</strong></td>
<td></td>
<td>DISPLAY:WINDow[:STATE]</td>
<td>Add</td>
</tr>
<tr>
<td><strong>New Sheet</strong></td>
<td></td>
<td>DISPLAY:SHEet:STATE</td>
<td>None</td>
</tr>
<tr>
<td><strong>Delete</strong></td>
<td>TrN</td>
<td>DISPLAY:WINDow:TRACe:DELETE</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>ChN</td>
<td>SYSTEM:CHANnels:DELETE</td>
<td>RemoveChannelNumber</td>
</tr>
<tr>
<td></td>
<td>WinN</td>
<td>DISPLAY:WINDow[:STATE]</td>
<td>Add</td>
</tr>
<tr>
<td><strong>Select</strong></td>
<td>TrN</td>
<td>DISPLAY:WINDow:TRACe:SELECT</td>
<td>None</td>
</tr>
<tr>
<td>ChN</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>WinN</td>
<td>DISPlay:WINDow:TRACe:SELect</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Measure.. .</td>
<td>S21, S11, S12, S22, AI1, AI2, CompIn21, CompOut21, DeltaGain21, CompGain21, CompS11, Ref21, CompAI1, CompAI2</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
</tbody>
</table>
CF_Setup Commands - GCX

Setup Commands - Gain Compression Converters Measurement Class

Only the Main and Layout Setup commands corresponding to the Gain Compression Converters measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>GCX Setup...</td>
</tr>
<tr>
<td>Sweep Type</td>
</tr>
<tr>
<td>Linear Sweep</td>
</tr>
<tr>
<td>CW Frequency</td>
</tr>
<tr>
<td>Segment Sweep</td>
</tr>
<tr>
<td>Data Acquisition Mode</td>
</tr>
<tr>
<td>Sweep</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>Sweep</td>
</tr>
<tr>
<td>Power Per Frequency (2D)</td>
</tr>
<tr>
<td>Sweep Frequency Per Power (2D)</td>
</tr>
<tr>
<td>Sweep Settings</td>
</tr>
<tr>
<td>Number of Points</td>
</tr>
<tr>
<td>IF Bandwidth</td>
</tr>
<tr>
<td>Start</td>
</tr>
<tr>
<td>Stop</td>
</tr>
<tr>
<td>Center</td>
</tr>
<tr>
<td>Span</td>
</tr>
<tr>
<td>Power</td>
</tr>
</tbody>
</table>

7570
<table>
<thead>
<tr>
<th></th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Power On (All Channels)</strong></td>
<td><strong>OUTPut[:STATe]</strong></td>
<td><strong>SourcePowerState</strong></td>
</tr>
<tr>
<td><strong>DUT Input Port</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Input Port</strong></td>
<td><strong>SENSe:GCSetup:PMAP</strong></td>
<td><strong>SetPortMap</strong></td>
</tr>
<tr>
<td><strong>Linear Input Power</strong></td>
<td><strong>SENSe:GCSetup:POWer:LIinear:INPut:LEVel</strong></td>
<td><strong>InputLinearPowerLevel</strong></td>
</tr>
<tr>
<td><strong>Source Attenuator</strong></td>
<td><strong>SOURce:POWer:ATTenuation</strong></td>
<td><strong>Attenuator</strong></td>
</tr>
<tr>
<td><strong>Receiver Attenuator(A)</strong></td>
<td><strong>SENSe:POWer:ATTenuation</strong></td>
<td><strong>ReceiverAttenuator</strong></td>
</tr>
<tr>
<td><strong>Source Leveling Mode</strong></td>
<td><strong>SOURce:POWer:ALC[:MODE]</strong></td>
<td><strong>ALCLevelingMode</strong></td>
</tr>
<tr>
<td><strong>DUT Output Port</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Output Port</strong></td>
<td><strong>SENSe:GCSetup:PMAP</strong></td>
<td><strong>SetPortMap</strong></td>
</tr>
<tr>
<td><strong>Reverse Power</strong></td>
<td><strong>SENSe:GCSetup:POWer:REVerse:LEVel</strong></td>
<td><strong>ReverseLinearPowerLevel</strong></td>
</tr>
<tr>
<td><strong>Source Attenuator Auto</strong></td>
<td><strong>SOURce:POWer:ATTenuation</strong> <strong>SOURce:POWer:ATTenuation:AUTO</strong></td>
<td><strong>Attenuator</strong> <strong>AttenuatorMode</strong></td>
</tr>
<tr>
<td>Component</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Receiver Attenuator(B)</td>
<td>SENSE:POWer:ATTenuation</td>
<td>ReceiverAttenuator</td>
</tr>
<tr>
<td>Source Leveling Mode</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
<tr>
<td>Power Sweep</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start (Min) Power</td>
<td>SENSE:GCSetup:POWer:STARt:LEVel</td>
<td>StartPower</td>
</tr>
<tr>
<td>Stop (Max) Power</td>
<td>SENSE:GCSetup:POWer:STOP:LEVel</td>
<td>StopPower</td>
</tr>
<tr>
<td>Power Points</td>
<td>SENSE:GCSetup:SWEep:FREQuency:POINts</td>
<td>NumberOfPowerPoints</td>
</tr>
<tr>
<td>Power Step</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Path Configuration..</td>
<td>SENSE:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
</tr>
</tbody>
</table>

**Compression**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Compression Method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Command Name</td>
<td>Value Type</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Compression from Linear Gain</td>
<td>SENSe:GCSetup:COMPression:ALGorithm</td>
<td>CompressionAlgorithm</td>
</tr>
<tr>
<td>Compression from Max Gain</td>
<td>SENSe:GCSetup:COMPression:ALGorithm</td>
<td>CompressionAlgorithm</td>
</tr>
<tr>
<td>Compression from Back Off</td>
<td>SENSe:GCSetup:COMPression:ALGorithm</td>
<td>CompressionAlgorithm</td>
</tr>
<tr>
<td>X/Y Compression</td>
<td>SENSe:GCSetup:COMPression:ALGorithm</td>
<td>CompressionAlgorithm</td>
</tr>
<tr>
<td>Compression from Saturation</td>
<td>SENSe:GCSetup:COMPression:ALGorithm</td>
<td>CompressionAlgorithm</td>
</tr>
<tr>
<td>Magnitude Level</td>
<td>SENSe:GCSetup:COMPression:LEVel</td>
<td>CompressionLevel</td>
</tr>
<tr>
<td>Back Off</td>
<td>SENSe:GCSetup:COMPression:BACKoff:LEVel</td>
<td>CompressionBackoff</td>
</tr>
<tr>
<td>Delta X</td>
<td>SENSe:GCSetup:COMPression:DELTa:X</td>
<td>CompressionDeltaX</td>
</tr>
<tr>
<td>Delta Y</td>
<td>SENSe:GCSetup:COMPression:DELTa:Y</td>
<td>CompressionDeltaY</td>
</tr>
<tr>
<td>From Max Pout</td>
<td>SENSe:GCSetup:COMPression:SATuration:LEVel</td>
<td>SaturationLevel</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Magnitude Only</td>
<td>SENSe:GCSetup:COMPression:PHASE:MODE</td>
<td>None</td>
</tr>
<tr>
<td>Phase Only</td>
<td>SENSe:GCSetup:COMPression:PHASE:MODE</td>
<td>None</td>
</tr>
<tr>
<td>Magnitude and Phase</td>
<td>SENSe:GCSetup:COMPression:PHASE:MODE</td>
<td>None</td>
</tr>
<tr>
<td>Phase Level</td>
<td>SENSe:GCSetup:COMPression:PHASE:LEVel</td>
<td>None</td>
</tr>
<tr>
<td>Phase Details...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Compute Linear Power from Percent of Span</td>
<td>SENSe:GCSetup:POWer:LINEar:INPut:COMPute:APERt</td>
<td>None</td>
</tr>
<tr>
<td>Smooth Power Sweep Using</td>
<td>SENSe:GCSetup:SWEep:POWer:SMOoth</td>
<td>None</td>
</tr>
<tr>
<td>Aperture</td>
<td>SENSe:GCSetup:SWEep:POWer:SMOoth:APERtue</td>
<td>None</td>
</tr>
<tr>
<td>Use Reference Mixer</td>
<td>SENSe:GCSetup:MIXer:REFerence</td>
<td>None</td>
</tr>
<tr>
<td>Force Source</td>
<td>SENSe:GCSetup:PMAP:SOURce:OVERride</td>
<td>None</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Power Out Port 1</td>
<td>SENSe:GCSetup:SMARt:TOLerance</td>
<td>SmartSweepTolerance</td>
</tr>
<tr>
<td>SMART Sweep</td>
<td>SENSe:GCSetup:SMARt:MITerations</td>
<td>SmartSweepMaximumIterations</td>
</tr>
<tr>
<td>Tolerance</td>
<td>SENSe:GCSetup:SMARt:SITerations</td>
<td>SmartSweepShowIterations</td>
</tr>
<tr>
<td>Maximum Iterations</td>
<td>SENSe:GCSetup:SMARt:CDC</td>
<td>ReadDCAtCompression</td>
</tr>
<tr>
<td>Show Iterations</td>
<td>SENSe:GCSetup:SAFE:ENABle</td>
<td>SafeSweepEnable</td>
</tr>
<tr>
<td>Safe Mode...</td>
<td>SENSe:GCSetup:SAFE:CPADjustment</td>
<td>SafeSweepCoarsePowerAdjustment</td>
</tr>
<tr>
<td>Coarse Increment</td>
<td>SENSe:GCSetup:SAFE:FPADjustment</td>
<td>SafeSweepFinePowerAdjustment</td>
</tr>
<tr>
<td>Fine Increment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command Sequence</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>Fine Threshold</td>
<td><code>SENSe:GCSetup:SAFE:FTHReshold</code></td>
<td>SafeSweepFineThreshold</td>
</tr>
<tr>
<td>Max Output Power</td>
<td><code>SENSe:GCSetup:SAFE:MLimit</code></td>
<td>SafeSweepMaximumLimit</td>
</tr>
<tr>
<td>2D Sweep</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Compression Point Interpolation</td>
<td><code>SENSe:GCSetup:COMPression:INTerpolation</code></td>
<td>CompressionInterpolation</td>
</tr>
<tr>
<td>End of Sweep Condition</td>
<td><code>SENSe:GCSetup:EOSoperation</code></td>
<td>EndOfSweepOperation</td>
</tr>
<tr>
<td>Settling Time</td>
<td><code>SENSe:GCSetup:SMARt:STIME</code></td>
<td>SmartSweepSettlingTime</td>
</tr>
<tr>
<td>Mixer Frequency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input</td>
<td><code>SENSe:MIXer:CALCulate</code></td>
<td>Calculate</td>
</tr>
<tr>
<td>LO1</td>
<td><code>SENSe:MIXer:LO:FREQuency:FIXed</code></td>
<td>LOFixedFrequency</td>
</tr>
<tr>
<td>Input &gt; LO</td>
<td><code>SENSe:MIXer:LO:FREQuency:ILTI</code></td>
<td>IsInputGreaterThanLO</td>
</tr>
<tr>
<td>Output</td>
<td><code>SENSe:MIXer:CALCulate</code></td>
<td>Calculate</td>
</tr>
<tr>
<td>Mixer Power</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Variable</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>--------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Power On (All Channels)</td>
<td><code>OUTPut[:STATe]</code></td>
<td><code>SourcePowerState</code></td>
</tr>
<tr>
<td>LO1 Power</td>
<td><code>SENSe:MIXer:LO:POWer</code></td>
<td><code>LOPower</code></td>
</tr>
<tr>
<td>LO2 Power</td>
<td><code>SENSe:MIXer:LO:POWer</code></td>
<td><code>LOPower</code></td>
</tr>
<tr>
<td>Source Leveling Mode</td>
<td><code>SOURce:POWer:ALC[:MODE]</code></td>
<td><code>ALCLevelingMode</code></td>
</tr>
<tr>
<td>Source Attenuator</td>
<td><code>SOURce:POWer:ATTenuation</code></td>
<td><code>Attenuator</code></td>
</tr>
<tr>
<td>Receiver Attenuator(A)</td>
<td><code>SENSe:POWer:ATTenuation</code></td>
<td><code>ReceiverAttenuator</code></td>
</tr>
<tr>
<td>Path Configuration..</td>
<td><code>SENSe:PATH:CONFig:ELEMent[:STATe]</code></td>
<td><code>Element</code></td>
</tr>
</tbody>
</table>

### Mixer Setup

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Converter Stages</td>
<td><code>SENSe:MIXer:STAGe</code></td>
<td><code>LOStage</code></td>
</tr>
<tr>
<td>Recall a previously-configured</td>
<td><code>SENSe:MIXer:LO:NAME</code></td>
<td><code>LOName</code></td>
</tr>
<tr>
<td>Source Type</td>
<td>Command</td>
<td>Function</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>external source</td>
<td>(assign a source to mixer input or LO)</td>
<td>AssignSourceToRole</td>
</tr>
<tr>
<td>Read all assigned roles</td>
<td>SENSe:MIXer:ROLE:CATalog?</td>
<td>GetSourceByRole</td>
</tr>
<tr>
<td>Read the source assigned to a role</td>
<td>SENSe:MIXer:ROLE:DEVice</td>
<td>GetSourceRoles</td>
</tr>
<tr>
<td>Input Numerator Frac.Mult</td>
<td>SENSe:MIXer:INPut:FREQuency:NUMerator</td>
<td>InputNumerator</td>
</tr>
<tr>
<td>Input Denominator Frac.Mult</td>
<td>SENSe:MIXer:INPut:FREQuency:DENominator</td>
<td>InputDenominator</td>
</tr>
<tr>
<td>LO Numerator Frac. Mult.</td>
<td>SENSe:MIXer:LO:FREQuency:NUMerator</td>
<td>LONumerator</td>
</tr>
<tr>
<td>LO Denominator Frac. Mult.</td>
<td>SENSe:MIXer:LO:FREQuency:DENominator</td>
<td>LODenominator</td>
</tr>
<tr>
<td>Path Configuration</td>
<td>SENSe:PATH:CONFig:ELEMENT[:STATE]</td>
<td>Element</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Save...</td>
<td>SENSe:MIXer:SAVE</td>
<td>SaveFile</td>
</tr>
<tr>
<td>Load...</td>
<td>SENSe:MIXer:LOAD</td>
<td>LoadFile</td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Quick Start...</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Balanced</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BalSMeasurement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BBalMeasurement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SBalMeasurement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SSBMeasurement</td>
</tr>
<tr>
<td>Other</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
</tr>
<tr>
<td>Layout Tab Commands</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
</tr>
<tr>
<td>New Trace</td>
<td>DISPlay:WINDow:TRACe[:STATE]</td>
<td>View</td>
</tr>
<tr>
<td>New Channel</td>
<td>None</td>
<td><code>chans.Add</code></td>
</tr>
<tr>
<td>------------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>New Window</td>
<td><code>DISPlay:WINDow[:STATe]</code></td>
<td>Add</td>
</tr>
<tr>
<td>New Sheet</td>
<td><code>DISPlay:SHEeet:STATe</code></td>
<td>None</td>
</tr>
<tr>
<td>Delete</td>
<td>TrN</td>
<td><code>DISPlay:WINDow:TRACe:DELeTe</code></td>
</tr>
<tr>
<td></td>
<td>ChN</td>
<td><code>SYSTem:CHANnels:DELeTe</code></td>
</tr>
<tr>
<td></td>
<td>WinN</td>
<td><code>DISPlay:WINDow[:STATe]</code></td>
</tr>
<tr>
<td>Select</td>
<td>TrN</td>
<td><code>DISPlay:WINDow:TRACe:SELeCT</code></td>
</tr>
<tr>
<td></td>
<td>ChN</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>WinN</td>
<td><code>DISPlay:WINDow:TRACe:SELeCT</code></td>
</tr>
<tr>
<td>Measure..</td>
<td>S21, S11, S12, S22, AI1, AI2, CompIn21, CompOut21, DeltaGain21, CompGain21, CompS11,</td>
<td><code>CALCulate:MEASure:PARameter</code></td>
</tr>
<tr>
<td>Ref21, CompAI1, CompAI2</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Meas Class...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**CF_Setup Commands - HotS22**

**Setup Commands - Active Hot Parameters**

Only the Main Setup commands corresponding to the Active Hot Parameters measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking [here](#) or by clicking on the softtabs on the graphic below.

Click [here](#) to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td><strong>Sweep Setup...</strong></td>
</tr>
<tr>
<td><strong>Sweep Type</strong></td>
</tr>
<tr>
<td><strong>Source</strong></td>
</tr>
<tr>
<td><strong>Start Frequency</strong></td>
</tr>
<tr>
<td><strong>Center Frequency</strong></td>
</tr>
<tr>
<td><strong>Stop Frequency</strong></td>
</tr>
<tr>
<td><strong>Frequency Span</strong></td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>-------------------------------</td>
</tr>
<tr>
<td>Frequency Sweep</td>
</tr>
<tr>
<td>Number of Frequencies</td>
</tr>
<tr>
<td>Power Level</td>
</tr>
<tr>
<td>Start Power</td>
</tr>
<tr>
<td>Stop Power</td>
</tr>
<tr>
<td>Number Of Powers</td>
</tr>
<tr>
<td>CW Frequency</td>
</tr>
<tr>
<td>Extraction</td>
</tr>
<tr>
<td>IF Bandwidth</td>
</tr>
<tr>
<td>Absolute Power</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Relative to Input Power</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Number of Phases</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>DC</td>
</tr>
<tr>
<td>DC Sources...</td>
</tr>
<tr>
<td>RF Path</td>
</tr>
<tr>
<td>Power On - all channels</td>
</tr>
<tr>
<td>Source</td>
</tr>
<tr>
<td>Source Port</td>
</tr>
<tr>
<td>Source Attenuator</td>
</tr>
<tr>
<td>Receiver A Attenuator</td>
</tr>
<tr>
<td>ALC Hardware</td>
</tr>
<tr>
<td>Receiver Leveling</td>
</tr>
<tr>
<td>Extraction</td>
</tr>
<tr>
<td>Extraction Port</td>
</tr>
</tbody>
</table>

7584
<table>
<thead>
<tr>
<th>Source Attenuator</th>
<th>SOURce:POWer&lt;port&gt;:ATTenuation</th>
<th>Attenuator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receiver C Attenuator</td>
<td>SENSE:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
</tr>
<tr>
<td>ALC Hardware</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
<tr>
<td>Receiver Leveling</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver</td>
<td>State</td>
</tr>
<tr>
<td>RF Path Config...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>X-axis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
</tr>
<tr>
<td>Select all traces</td>
</tr>
<tr>
<td>X-axis display</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fixed Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Power</td>
</tr>
<tr>
<td>Frequency</td>
</tr>
<tr>
<td>Enable Interpolation</td>
</tr>
<tr>
<td>Name (DC source)</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>Meas Class...</td>
</tr>
<tr>
<td>Quick Start...</td>
</tr>
<tr>
<td>S-Param</td>
</tr>
<tr>
<td>Balanced</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Other</td>
</tr>
</tbody>
</table>


**CF_Setup Commands - IMD**

**Setup Commands - Swept IMD Measurement Class**

Only the Main and Layout Setup commands corresponding to the Swept IMD measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMD Setup</td>
<td>Frequency</td>
<td>Sweep Type</td>
<td></td>
</tr>
<tr>
<td>Sweep fc</td>
<td>SENSe:IMD:SWEep:TYPE</td>
<td>SweepType</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------------------</td>
<td>---------------------------</td>
<td></td>
</tr>
<tr>
<td>Start fc</td>
<td>SENSE:IMD:FREQuency:FCENter:STARt</td>
<td>FrequencyCenterStart</td>
<td></td>
</tr>
<tr>
<td>Center fc</td>
<td>SENSE:IMD:FREQuency:FCENter:CENTer</td>
<td>FrequencyCenter</td>
<td></td>
</tr>
<tr>
<td>Fixed DeltaF</td>
<td>SENSE:IMD:FREQuency:DFRequency[:CW]</td>
<td>DeltaFrequency</td>
<td></td>
</tr>
<tr>
<td>Number Of Points</td>
<td>SENSE:SWEep:POINt</td>
<td>NumberOfPoints</td>
<td></td>
</tr>
<tr>
<td>Stop fc</td>
<td>SENSE:IMD:FREQuency:FCENter:STOP</td>
<td>FrequencyCenterStop</td>
<td></td>
</tr>
<tr>
<td>Span fc</td>
<td>SENSE:IMD:FREQuency:FCENter:SPAN</td>
<td>FrequencyCenterSpan</td>
<td></td>
</tr>
<tr>
<td>Main Tone IFBW</td>
<td>SENSE:IMD:IFBWidth:MAIN</td>
<td>MainToneIFBandwidth</td>
<td></td>
</tr>
<tr>
<td>IM Tone IFBW</td>
<td>SENSE:IMD:IFBWidth:IMTone</td>
<td>IMToneIFBandwidth</td>
<td></td>
</tr>
<tr>
<td>Reduce IF BW at Low Frequencies</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>Power</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power On (All Channels)</td>
<td>OUTPut[:STATe]</td>
<td>SourcePowerState</td>
<td></td>
</tr>
<tr>
<td><strong>DUT Input</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input Port</td>
<td>SENSE:IMD:PMAP</td>
<td>SetPortMap</td>
<td></td>
</tr>
<tr>
<td>Source Attenuator</td>
<td>SOURce:POWer:ATTenuation</td>
<td>Attenuator</td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>Receiver Attenuator</td>
<td>SENSe:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
<td></td>
</tr>
<tr>
<td><strong>DUT Output</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output Port</td>
<td>SENSe:IMD:PMap</td>
<td>SetPortMap</td>
<td></td>
</tr>
<tr>
<td>Source Attenuator</td>
<td>SOURce:POWer:ATTenuation</td>
<td>Attenuator</td>
<td></td>
</tr>
<tr>
<td>Receiver Attenuator</td>
<td>SENSe:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
<td></td>
</tr>
<tr>
<td><strong>Tone Powers</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Coupled Tone Powers</td>
<td>SENSe:IMD:TPOWer:COUPle:[STATe]</td>
<td>CoupleTonePower</td>
<td></td>
</tr>
<tr>
<td>ALC On</td>
<td>SENSe:IMD:TPower:LEV</td>
<td>LevelingMethod</td>
<td></td>
</tr>
<tr>
<td><strong>Power Leveling</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set Input Power</td>
<td>SENSe:IMD:TPower:LEV</td>
<td>LevelingMethod</td>
<td></td>
</tr>
<tr>
<td>Set Input Power, receiver leveling</td>
<td>SENSe:IMD:TPower:LEV</td>
<td>LevelingMethod</td>
<td></td>
</tr>
<tr>
<td>Set Input Power, equal tones at output</td>
<td>SENSe:IMD:TPower:LEV</td>
<td>LevelingMethod</td>
<td></td>
</tr>
<tr>
<td>Set Output Power, receiver leveling</td>
<td>SENSE:IMD:TPOWER:LEV</td>
<td>LevelingMethod</td>
<td></td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>----------------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>Fixed</td>
<td>SENSE:IMD:TPOWER:F1</td>
<td>TonePower</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:IMD:TPOWER:F2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>SENSE:IMD:TPOWER:F1:START</td>
<td>TonePowerStart</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:IMD:TPOWER:F2:START</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stop</td>
<td>SENSE:IMD:TPOWER:F1:STOP</td>
<td>TonePowerStop</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:IMD:TPOWER:F2:STOP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Step</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Path Configuration...</td>
<td>SENSE:PATH:CONFig:ELEMent[:STATE]</td>
<td>Element</td>
<td></td>
</tr>
<tr>
<td>Configure</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add Source...</td>
<td>SENSE:ROLE:DEVice</td>
<td>AssignSourceToRole</td>
<td></td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
<tr>
<td>Quick Start...</td>
<td>S-Param</td>
<td>CreateSParameterEx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Balanced</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BalSMeasurement</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>BBalMeasurement</td>
<td></td>
</tr>
</tbody>
</table>

7590
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Trace</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
<td></td>
</tr>
<tr>
<td>New Channel</td>
<td>None</td>
<td></td>
<td>chans.Add</td>
</tr>
<tr>
<td>New Window</td>
<td>DISPlay:WINDow[:STATe]</td>
<td>Add</td>
<td></td>
</tr>
<tr>
<td>New Sheet</td>
<td>DISPlay:SHEet:STATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Delete</td>
<td>TrN</td>
<td>DISPlay:WINDow:TRACe:DELeTe</td>
<td>None</td>
</tr>
<tr>
<td>ChN</td>
<td>SYSTem:CHANnels:DELeTe</td>
<td></td>
<td>RemoveChannelNumber</td>
</tr>
<tr>
<td>WinN</td>
<td>DISPlay:WINDow[:STATe]</td>
<td>Add</td>
<td></td>
</tr>
<tr>
<td>Select</td>
<td>TrN</td>
<td>DISPlay:WINDow:TRACe:SELeCt</td>
<td>None</td>
</tr>
<tr>
<td>ChN</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>WinN</td>
<td>DISPLAY:WINDOW:TRACE:SELECT</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Measure...</td>
<td>CALCULATE:MEASURE:PARAMETER</td>
<td>CreateSPParameterEx</td>
<td></td>
</tr>
<tr>
<td>Measure Class...</td>
<td>CALCULATE:MEASURE:DEFINE</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
</tbody>
</table>
CF_Setup Commands - IMDX

Setup Commands - Swept IMD Converters Measurement Class

Only the Main and Layout Setup commands corresponding to the Swept IMD Converters measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>IMDX Setup...</td>
</tr>
<tr>
<td>Sweep Type</td>
</tr>
<tr>
<td>Sweep fc</td>
</tr>
<tr>
<td>Sweep DeltaF</td>
</tr>
<tr>
<td>Power Sweep</td>
</tr>
<tr>
<td>CW</td>
</tr>
<tr>
<td>LO Power Sweep</td>
</tr>
<tr>
<td>X-Axis Display</td>
</tr>
</tbody>
</table>

7593
<table>
<thead>
<tr>
<th>Annotation</th>
<th>CALCulate:MEASure:MIXer:XAXis</th>
<th>ActiveXAxisRange</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sweep Settings</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start fc</td>
<td>SENSE:IMD:FREQuency:FCENter:STARt</td>
<td>FrequencyCenterStart</td>
</tr>
<tr>
<td>Stop fc</td>
<td>SENSE:IMD:FREQuency:FCENter:STOP</td>
<td>FrequencyCenterStop</td>
</tr>
<tr>
<td>Center fc</td>
<td>SENSE:IMD:FREQuency:FCENter:CENTER</td>
<td>FrequencyCenter</td>
</tr>
<tr>
<td>Span fc</td>
<td>SENSE:IMD:FREQuency:FCENter:SPAN</td>
<td>FrequencyCenterSpan</td>
</tr>
<tr>
<td>Fixed DeltaF</td>
<td>SENSE:IMD:FREQuency:DFRequency[:CW]</td>
<td>DeltaFrequency</td>
</tr>
<tr>
<td>Number Of Points</td>
<td>SENSE:SWEep:POINt</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>Main Tone IFBW</td>
<td>SENSE:IMD:IFBWidth:MAIN</td>
<td>MainToneIFBandwidth</td>
</tr>
<tr>
<td>IM Tone IFBW</td>
<td>SENSE:IMD:IFBWidth:IMTone</td>
<td>IMToneIFBandwidth</td>
</tr>
<tr>
<td>Reduce IF BW at Low</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Frequencies</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Tone Power</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power On (All Channels)</td>
<td>OUTPut[:STATe]</td>
<td>SourcePowerState</td>
</tr>
<tr>
<td></td>
<td>Command</td>
<td>Method</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><strong>DUT Input Port</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Input Port</strong></td>
<td>SENSE:IMD:PMAP</td>
<td>SetPortMap</td>
</tr>
<tr>
<td><strong>Source Attenuator</strong></td>
<td>SOURce:POWer:ATTenuation</td>
<td>Attenuator</td>
</tr>
<tr>
<td><strong>Receiver Attenuator</strong></td>
<td>SENSE:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
</tr>
<tr>
<td><strong>DUT Output Port</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Output Port</strong></td>
<td>SENSE:IMD:PMAP</td>
<td>SetPortMap</td>
</tr>
<tr>
<td><strong>Source Attenuator</strong></td>
<td>SOURce:POWer:ATTenuation</td>
<td>Attenuator</td>
</tr>
<tr>
<td><strong>Receiver Attenuator</strong></td>
<td>SOURce:POWer:ATTenuation</td>
<td>ReceiverAttenuator</td>
</tr>
<tr>
<td><strong>Tone Powers</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Coupled Tone Powers</strong></td>
<td>SENSE:IMD:TPower:COUPle[:STATE]</td>
<td>CoupleTonePower</td>
</tr>
<tr>
<td><strong>ALC On</strong></td>
<td>SENSE:IMD:TPower:LEV</td>
<td>LevelingMethod</td>
</tr>
<tr>
<td><strong>Power Leveling</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Set Input Power</strong></td>
<td>SENSE:IMD:TPower:LEV</td>
<td>LevelingMethod</td>
</tr>
<tr>
<td>Function</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>Set Input Power, receiver leveling</td>
<td>SENSe:IMD:TPOWer:LEV</td>
<td>LevelingMethod</td>
</tr>
<tr>
<td>Set Input Power, equal tones at output</td>
<td>SENSe:IMD:TPOWer:LEV</td>
<td>LevelingMethod</td>
</tr>
<tr>
<td>Set Output Power, receiver leveling</td>
<td>SENSe:IMD:TPOWer:LEV</td>
<td>LevelingMethod</td>
</tr>
<tr>
<td>Fixed</td>
<td>SENSe:IMD:TPOWer:F1</td>
<td>TonePower</td>
</tr>
<tr>
<td></td>
<td>SENSe:IMD:TPOWer:F2</td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>SENSe:IMD:TPOWer:F1:STARt</td>
<td>TonePowerStart</td>
</tr>
<tr>
<td></td>
<td>SENSe:IMD:TPOWer:F2:STARt</td>
<td></td>
</tr>
<tr>
<td>Stop</td>
<td>SENSe:IMD:TPOWer:F1:STOP</td>
<td>TonePowerStop</td>
</tr>
<tr>
<td></td>
<td>SENSe:IMD:TPOWer:F2:STOP</td>
<td></td>
</tr>
<tr>
<td>Step</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Path Configuration...</td>
<td>SENSe:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
</tr>
</tbody>
</table>

### Mixer Frequency

<table>
<thead>
<tr>
<th>Component</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>SENSe:MIXer:CALCulate</td>
<td>Calculate</td>
</tr>
<tr>
<td>LO1</td>
<td>SENSe:MIXer:LO:FREQuency:FIXed</td>
<td>LOFixedFrequency</td>
</tr>
<tr>
<td>Input &gt; LO</td>
<td>SENSe:MIXer:LO:FREQuency:ILT</td>
<td>IsInputGreaterThanLO</td>
</tr>
<tr>
<td>-----------</td>
<td>--------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Output</td>
<td>SENSe:MIXer:CALCulate</td>
<td>Calculate</td>
</tr>
<tr>
<td><strong>Mixer Power</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power On (All Channels)</td>
<td>OUTPUT[:STATe]</td>
<td>SourcePowerState</td>
</tr>
<tr>
<td>LO1 Power</td>
<td>SENSe:MIXer:LO:POWer</td>
<td>LOPower</td>
</tr>
<tr>
<td>LO2 Power</td>
<td>SENSe:MIXer:LO:POWer</td>
<td>LOPower</td>
</tr>
<tr>
<td>LO Start Power</td>
<td>SENSe:MIXer:LO:FREQuency:STARt</td>
<td>LOStartFrequency</td>
</tr>
<tr>
<td>LO Stop Power</td>
<td>SENSe:MIXer:LO:FREQuency:STOP</td>
<td>LOStopFrequency</td>
</tr>
<tr>
<td>Source Attenuator</td>
<td>SOURce:POWer:ATTenuation</td>
<td>Attenuator</td>
</tr>
<tr>
<td>Receiver Attenuator</td>
<td>SENSe:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
</tr>
<tr>
<td>Path Configuration...</td>
<td>SENSe:PATH:CONFig:ELEMen[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td><strong>Mixer Setup</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Converter Stages</td>
<td>SENSe:MIXer:STAGe</td>
<td>LOStage</td>
</tr>
<tr>
<td>Operation</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>----------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Recall a previously-configured external source</td>
<td>SENSE:MIXer:LO:NAME</td>
<td>LOName</td>
</tr>
<tr>
<td>Assign a source to mixer input or LO</td>
<td>SENSE:MIXer:ROLE:DEVice</td>
<td>AssignSourceToRole</td>
</tr>
<tr>
<td>Read all assigned roles</td>
<td>SENSE:MIXer:ROLE:CATalog?</td>
<td>GetSourceByRole</td>
</tr>
<tr>
<td>Read the source assigned to a role</td>
<td>SENSE:MIXer:ROLE:DEVice</td>
<td>GetSourceRoles</td>
</tr>
<tr>
<td>Input Numerator Frac.Mult</td>
<td>SENSE:MIXer:INPUT:FREQuency:NUMerator</td>
<td>InputNumerator</td>
</tr>
<tr>
<td>Input Denominator Frac.Mult</td>
<td>SENSE:MIXer:INPUT:FREQuency:DENominator</td>
<td>InputDenominator</td>
</tr>
<tr>
<td>LO Numerator Frac. Mult.</td>
<td>SENSE:MIXer:LO:FREQuency:NUMerator</td>
<td>LONumerator</td>
</tr>
<tr>
<td>LO Denominator Frac.Mult</td>
<td>SENSE:MIXer:LO:FREQuency:DENominator</td>
<td>LODenominator</td>
</tr>
<tr>
<td>Path Configuration...</td>
<td>SENSE:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td>Save...</td>
<td>SENSE:MIXer:SAVE</td>
<td>SaveFile</td>
</tr>
<tr>
<td>Load...</td>
<td>SENSE:MIXer:LOAD</td>
<td>LoadFile</td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Quick Start...</td>
<td>S-Param</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>----------------</td>
<td>---------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>Other</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
</tr>
</tbody>
</table>

**Layout Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Trace</td>
<td>DISPlay:WINDow:TRACe[:STATE]</td>
<td>View</td>
<td></td>
</tr>
<tr>
<td>New Channel</td>
<td>None</td>
<td>chans.Add</td>
<td></td>
</tr>
<tr>
<td>New Window</td>
<td>DISPlay:WINDow[:STATE]</td>
<td>Add</td>
<td></td>
</tr>
<tr>
<td>New Sheet</td>
<td>DISPlay:SHEet:STATE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Delete</td>
<td>TrN</td>
<td>DISPlay:WINDow:TRACe:DELeTe</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>ChN</td>
<td>SYSTem:CHANnels:DELeTe</td>
<td>RemoveChannelNumber</td>
</tr>
<tr>
<td></td>
<td>WinN</td>
<td>DISPlay:WINDow[:STATe]</td>
<td>Add</td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------------</td>
<td>----------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Select</td>
<td>TrN</td>
<td>DISPlay:WINDow:TRACe:SELeCt</td>
<td>None</td>
</tr>
<tr>
<td>ChN</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>WinN</td>
<td>DISPlay:WINDow:TRACe:SELeCt</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Measure...</td>
<td>CALCulate:MEASure:PARameTer</td>
<td></td>
<td>CreateSParameterEx</td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td></td>
<td>CreateCustomMeasurementEx</td>
</tr>
</tbody>
</table>
CF_Setup Commands - IMS

Setup Commands - IM Spectrum Measurement Class

Only the Main and Layout Setup commands corresponding to the IM Spectrum measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>IMS Setup...</td>
</tr>
<tr>
<td>Sweep Type</td>
</tr>
<tr>
<td>Linear</td>
</tr>
<tr>
<td>2nd Order Spectrum</td>
</tr>
<tr>
<td>3rd Order Spectrum</td>
</tr>
<tr>
<td>Nth Order Spectrum</td>
</tr>
<tr>
<td>Order N</td>
</tr>
<tr>
<td>Resolution BW</td>
</tr>
<tr>
<td>Stimulus Settings</td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>fc (Tone Center)</td>
</tr>
<tr>
<td>Delta F</td>
</tr>
<tr>
<td>f1</td>
</tr>
<tr>
<td>f2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response Settings</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Spectrum</td>
<td>SENSe:IMS:RESPonse:STARt</td>
<td>SpectrumStartFrequency</td>
</tr>
<tr>
<td>Stop Spectrum</td>
<td>SENSe:IMS:RESPonse:STOP</td>
<td>SpectrumStopFrequency</td>
</tr>
<tr>
<td>Center Spectrum</td>
<td>SENSe:IMS:RESPonse:CENTer</td>
<td>SpectrumCenterFrequency</td>
</tr>
<tr>
<td>Span Spectrum</td>
<td>SENSe:IMS:RESPonse:SPAN</td>
<td>SpectrumSpanFrequency</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Power</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Power On (All Channels)</td>
<td>OUTPut[STATE]</td>
<td>SourcePowerState</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DUT Input</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Port</td>
<td>SENSe:IMS:PMAP</td>
<td>SetPortMap</td>
</tr>
<tr>
<td>Source Attenuator</td>
<td>SOURce:POWer:ATTenuation</td>
<td>Attenuator</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Receiver Attenuator</td>
<td>SENSE:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
</tr>
</tbody>
</table>

**DUT Output**

<table>
<thead>
<tr>
<th>Output Port</th>
<th>SENSE:IMS:PMAP</th>
<th>SetPortMap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Attenuator</td>
<td>SOURce:POWer:ATTenuation</td>
<td>Attenuator</td>
</tr>
<tr>
<td>Receiver Attenuator</td>
<td>SENSE:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
</tr>
</tbody>
</table>

**Tone Powers**

<table>
<thead>
<tr>
<th>Coupled Tone Powers</th>
<th>SENSE:IMS:TPOWer:COUPle[:STATe]</th>
<th>CoupleTonePower</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALC On</td>
<td>SENSE:IMS:TPOWer:LEVel</td>
<td>LevelingMethod</td>
</tr>
</tbody>
</table>

**Power Leveling**

<table>
<thead>
<tr>
<th>Set Input Power</th>
<th>SENSE:IMS:TPOWer:LEVel</th>
<th>LevelingMethod</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Input Power, receiver leveling</td>
<td>SENSE:IMS:TPOWer:LEVel</td>
<td>LevelingMethod</td>
</tr>
<tr>
<td>Set Input Power, equal tones at output</td>
<td>SENSE:IMS:TPOWer:LEVel</td>
<td>LevelingMethod</td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Method</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Set Output Power, receiver</td>
<td>SENSE:IMS:TPOWER:LEVel</td>
<td>LevelingMethod</td>
</tr>
<tr>
<td>leveling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fixed</td>
<td>SENSE:IMS:STIMulus:TPOWER:F1</td>
<td>TonePower</td>
</tr>
<tr>
<td></td>
<td>SENSE:IMS:STIMulus:TPOWER:F2</td>
<td></td>
</tr>
<tr>
<td>Path Configuration...</td>
<td>SENSE:PATH:CONFig:ELEMENT[:STATE]</td>
<td>Element</td>
</tr>
<tr>
<td>Tracking</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tracking Enable</td>
<td>SENSE:IMS:TRACKing:STATE</td>
<td>TrackingEnable</td>
</tr>
<tr>
<td>Tracking Channel</td>
<td>SENSE:IMS:TRACKing:CHANnel</td>
<td>TrackingChannel</td>
</tr>
<tr>
<td>Step Mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Automatic Step</td>
<td>SENSE:IMS:TRACKing:MSENable</td>
<td>TrackingManualStepEnable</td>
</tr>
<tr>
<td>Manual Step</td>
<td>SENSE:IMS:TRACKing:MSENable</td>
<td>TrackingManualStepEnable</td>
</tr>
<tr>
<td>Stimulus Point</td>
<td>SENSE:IMS:TRACKing:SINDex</td>
<td>TrackingStepIndex</td>
</tr>
<tr>
<td>Configure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add Source...</td>
<td>SENSE:ROLE:DEVICE</td>
<td>AssignSourceToRole</td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Quick Start...</td>
<td>S-Param</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Other</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSPParameterEx</td>
</tr>
</tbody>
</table>

**Layout Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Trace</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
<td></td>
</tr>
<tr>
<td>New Channel</td>
<td>None</td>
<td>chans.Add</td>
<td></td>
</tr>
<tr>
<td>New Window</td>
<td>DISPlay:WINDow[:STATe]</td>
<td>Add</td>
<td></td>
</tr>
<tr>
<td>New Sheet</td>
<td>DISPlay:SHEet:STATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Delete</td>
<td>TrN</td>
<td>DISPlay:WINDow:TRACe:DELeTe</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>System: Channels: Delete</td>
<td>Remove Channel Number</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>--------------------------</td>
<td>-----------------------</td>
<td></td>
</tr>
<tr>
<td><strong>ChN</strong></td>
<td>SYSTem:CHANnels:DELeTe</td>
<td>RemoveChannelNumber</td>
<td></td>
</tr>
<tr>
<td><strong>WinN</strong></td>
<td>DISPlay:WINDow[:STATe]</td>
<td>Add</td>
<td></td>
</tr>
<tr>
<td><strong>Select</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>TrN</strong></td>
<td>DISPlay:WINDow:TRACe:SElEcT</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>ChN</strong></td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>WinN</strong></td>
<td>DISPlay:WINDow:TRACe:SElEcT</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>Measure...</strong></td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
<td></td>
</tr>
<tr>
<td><strong>Meas Class...</strong></td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
</tbody>
</table>
CF_Setup Commands - IMSX

Setup Commands - IM Spectrum Converters Measurement Class

Only the Main and Layout Setup commands corresponding to the IM Spectrum Converters measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>IMSX Setup...</td>
</tr>
<tr>
<td>Sweep Type</td>
</tr>
<tr>
<td>Linear</td>
</tr>
<tr>
<td>2nd Order Spectrum</td>
</tr>
<tr>
<td>3rd Order Spectrum</td>
</tr>
<tr>
<td>Nth Order Spectrum</td>
</tr>
<tr>
<td>Order N</td>
</tr>
<tr>
<td>Resolution BW</td>
</tr>
</tbody>
</table>

7607
<table>
<thead>
<tr>
<th><strong>Stimulus Settings</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>fc (Tone Center)</td>
</tr>
<tr>
<td>Delta F</td>
</tr>
<tr>
<td>f1</td>
</tr>
<tr>
<td>f2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Response Settings</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Spectrum</td>
</tr>
<tr>
<td>Stop Spectrum</td>
</tr>
<tr>
<td>Center Spectrum</td>
</tr>
<tr>
<td>Span Spectrum</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Tone Power</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Power On (All Channels)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>DUT Input</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Port</td>
</tr>
<tr>
<td>Source Attenuator</td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>Receiver Attenuator</td>
</tr>
<tr>
<td><strong>DUT Output</strong></td>
</tr>
<tr>
<td>Output Port</td>
</tr>
<tr>
<td>Source Attenuator</td>
</tr>
<tr>
<td>Receiver Attenuator</td>
</tr>
<tr>
<td><strong>Tone Powers</strong></td>
</tr>
<tr>
<td>Coupled Tone Powers</td>
</tr>
<tr>
<td>ALC On</td>
</tr>
<tr>
<td><strong>Power Leveling</strong></td>
</tr>
<tr>
<td>Set Input Power</td>
</tr>
<tr>
<td>Set Input Power, receiver leveling</td>
</tr>
<tr>
<td>Set Input Power, equal tones at output</td>
</tr>
<tr>
<td>Set Output Power, receiver leveling</td>
</tr>
<tr>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Fixed</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Path Configuration...</td>
</tr>
<tr>
<td><strong>Mixer Frequency</strong></td>
</tr>
<tr>
<td>Input</td>
</tr>
<tr>
<td>LO1</td>
</tr>
<tr>
<td>Input &gt; LO</td>
</tr>
<tr>
<td>Output</td>
</tr>
<tr>
<td><strong>Mixer Power</strong></td>
</tr>
<tr>
<td>Power On (All Channels)</td>
</tr>
<tr>
<td>LO1 Power</td>
</tr>
<tr>
<td>LO2 Power</td>
</tr>
<tr>
<td>Path Configuration...</td>
</tr>
<tr>
<td>Mixer Setup</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Recall a previously-configured external source</td>
</tr>
<tr>
<td>Assign a source to mixer input or LO</td>
</tr>
<tr>
<td>Read all assigned roles</td>
</tr>
<tr>
<td>Read the source assigned to a role</td>
</tr>
<tr>
<td>Input Numerator Frac.Mult</td>
</tr>
<tr>
<td>LO Numerator Frac. Mult</td>
</tr>
<tr>
<td>LO Denominator Frac.Mult</td>
</tr>
<tr>
<td>Path Configuration...</td>
</tr>
<tr>
<td>Save...</td>
</tr>
<tr>
<td>Load...</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>Meas Class...</td>
</tr>
<tr>
<td>Quick Start...</td>
</tr>
<tr>
<td>Balanced</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Other</td>
</tr>
</tbody>
</table>

### Layout Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Trace</td>
<td></td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>New Channel</td>
<td></td>
<td>None</td>
<td>chans.Add</td>
</tr>
<tr>
<td>New Window</td>
<td></td>
<td>DISPlay:WINDow[:STATe]</td>
<td>Add</td>
</tr>
<tr>
<td>New Sheet</td>
<td></td>
<td>DISPlay:SHEet:STATe</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Column 1</td>
<td>Column 2</td>
<td>Column 3</td>
</tr>
<tr>
<td>-------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>Delete</td>
<td>TrN</td>
<td>DISPlay:WINDow:TRACe:DELeTe</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>ChN</td>
<td>SYSTem:CHANnels:DELeTe</td>
<td>RemoveChannelNumber</td>
</tr>
<tr>
<td></td>
<td>WinN</td>
<td>DISPlay:WINDow[:STATe]</td>
<td>Add</td>
</tr>
<tr>
<td>Select</td>
<td>TrN</td>
<td>DISPlay:WINDow:TRACe:SELect</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>ChN</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>WinN</td>
<td>DISPlay:WINDow:TRACe:SELect</td>
<td>None</td>
</tr>
<tr>
<td>Measure...</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
<td></td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
</tbody>
</table>
CF_Setup Commands - MOD

Setup Commands - Modulation Distortion and Modulation Distortion Converters Measurement Class

Only the Main Setup commands corresponding to the Modulation Distortion and Modulation Distortion Converters measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOD/MOD DX Setup...</td>
<td>Sweep Tab</td>
<td>SENSE:DISTortion:SWEep:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Sweep Type</td>
<td></td>
<td>SENSE:DISTortion:SWEep:CARRier:FREQuency</td>
<td>None</td>
</tr>
<tr>
<td>Carrier Frequency</td>
<td></td>
<td>SENSE:FREQuency:CENTer</td>
<td>chan.CenterFrequency</td>
</tr>
<tr>
<td>SA Center</td>
<td></td>
<td>SENSE:FREQuency:STARt</td>
<td>chan.StartFrequency</td>
</tr>
<tr>
<td>SA Start</td>
<td></td>
<td>SENSE:FREQuency:SPAN</td>
<td>chan.FrequencySpan</td>
</tr>
<tr>
<td>SA Span</td>
<td></td>
<td>SENSE:FREQuency:SPAN:FULL</td>
<td>FrequencySpanFull</td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>SA Stop</td>
<td>SENSe:FREQuency:STOP</td>
<td>chan.StopFrequency</td>
<td></td>
</tr>
<tr>
<td>Noise BW</td>
<td>SENSe:SA:BANDwidth:NOISe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSe:SA:BANDwidth:NOISe:AUTO</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Set Power At</td>
<td>SENSe:DISTortion:SWEep:POWER:CARRier:LEVel</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSe:DISTortion:SWEep:POWER:CARRier:LEVel:PORT</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>S-Param Power At</td>
<td>SENSe:DISTortion:SWEep:POWER:SPARam:LEVel</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>DUT In/Out</td>
<td>SENSe:DISTortion:SWEep:POWER:CARRier:LEVel:PORT</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Power Sweep...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power Sweep Type</td>
<td>ENSe:DISTortion:SWEep:POWER:CARRier:LEVel:TYPE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Ramp Power Sweep Type</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start Power At</td>
<td>SENSe:DISTortion:SWEep:POWER:CARRier:RAMP:LEVel:START</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Stop Power At</td>
<td>SENSe:DISTortion:SWEep:POWER:CARRier:RAMP:LEVel:STOP</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Number of Powers</td>
<td>SENSe:DISTortion:SWEep:POWER:CARRier:RAMP:POINts</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Noise BW</td>
<td>SENSe:DIStortion:SWEep:POWer:CARRier:LIST:NBW</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>Auto-Increase NBW</td>
<td>SENSe:DIStortion:SWEep:POWer:CARRier:RAMP:NBW:AUTO</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>List Power</td>
<td>SENSe:DIStortion:SWEep:POWer:CARRier:LIST:LEVel</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sweep Type</td>
<td>SENSe:DIStortion:SWEep:POWer:CARRier:LIST:SOURce:ATTenuation</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSe:DIStortion:SWEep:POWer:CARRier:LIST:SOURce:ATTenuation:MODE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSe:DIStortion:SWEep:POWer:CARRier:LIST:RECeiver:ATTenuation</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Custom</td>
<td>SENSe:DIStortion:SWEep:POWer:CARRier:LIST:NBW</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Noise BW Fixed</td>
<td>SENSe:DIStortion:SWEep:POWer:CARRier:LIST:NBW:MODE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Custom Auto-Increase</td>
<td>SENSe:DIStortion:SWEep:POWer:CARRier:LIST:NBW:MODE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Action</td>
<td>Command</td>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>Add Row</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LIST:ADD</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Delete Row</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LIST:DELETE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Load...</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LIST:LOAD</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Save...</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LIST:SAVE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>DC Sources...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable DC Outputs</td>
<td>SOURce:DC:ENABLE</td>
<td>EnableAllOutput</td>
<td></td>
</tr>
<tr>
<td>Min</td>
<td><strong>SOURce:DC:LI*M:Min</strong>mum</td>
<td>LimitMin</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td><strong>SOURce:DC:LI*M:Max</strong>imum</td>
<td>LimitMax</td>
<td></td>
</tr>
</tbody>
</table>

**Sweep Details...**

<table>
<thead>
<tr>
<th>Force RF Power OFF at the End of Sweep</th>
<th><strong>SENSe:DI*Song:SWEP:RETRace:POWer</strong></th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay Before Start of Sweep</td>
<td><strong>SENSe:SWEP:DELay</strong></td>
<td>SweepDelay</td>
</tr>
<tr>
<td>Delay Before Distortion Measurement</td>
<td><strong>SENSe:DI*Song:SWEP:DELI</strong></td>
<td>None</td>
</tr>
<tr>
<td>Enable S-Param Sweep</td>
<td><strong>SENSe:DI*Song:SWEP:SPARAM[:STATe]</strong></td>
<td>None</td>
</tr>
<tr>
<td>S-Param Source</td>
<td><strong>SENSe:DI*Song:SWEP:SPARAM:TYPE</strong></td>
<td>None</td>
</tr>
<tr>
<td>S-Param Freq Step</td>
<td><strong>SENSe:DI*Song:SWEP:SPARAM:BF</strong></td>
<td>None</td>
</tr>
<tr>
<td>S-Param IFBW</td>
<td><strong>SENSe:DI*Song:SWEP:SPARAM:IF</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>RF Path Tab</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td><strong>VNA Source Attenuator Include</strong></td>
<td><strong>SENSe:DIStortion:PATH:SOURce:ATTenuation:INCLUDe</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Nominal Source Amp</strong></td>
<td><strong>SENSe:DIStortion:PATH:SOURce:NOMinal:AMPLifier</strong></td>
<td></td>
</tr>
<tr>
<td><strong>DUT Input</strong></td>
<td><strong>SENSe:DIStortion:PATH:DUT:INPut</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Nominal DUT Gain</strong></td>
<td><strong>SENSe:DIStortion:PATH:DUT:NOMinal:GAIN</strong></td>
<td></td>
</tr>
<tr>
<td><strong>DUT Output</strong></td>
<td><strong>SENSe:DIStortion:PATH:DUT:OUTPut</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Receiver Attenuator</strong></td>
<td><strong>SOURce:POWer:ATTenuation:RECeiver:TEST</strong></td>
<td></td>
</tr>
<tr>
<td><strong>RF Path Config...</strong></td>
<td><strong>SENSe:PATH:CONFig:ELEMent[:STATe]</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Offsets and Limits...</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Mixer Tab</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Converter Stages</strong></td>
<td>SENSe:MIXer:STAGE</td>
<td>LOStage</td>
</tr>
<tr>
<td><strong>Enable Embedded LO</strong></td>
<td>SENSe:MIXer:ELO:STATe</td>
<td>IsOn</td>
</tr>
</tbody>
</table>

**Setup...**

<p>| <strong>Enable Embedded LO</strong> | SENSe:MIXer:ELO:STATe | IsOn |
| <strong>Tuning Method</strong> | SENSe:MIXer:ELO:TUNing:MODE | TuningMode |
| <strong>Tune every</strong> | SENSe:MIXer:ELO:TUNing:INTerval | TuningSweepInterval |
| <strong>Broadband Search</strong> | SENSe:MIXer:ELO:TUNing:SPAN | BroadbandTuningSpan |
| <strong>Noise BW</strong> | SENSe:MIXer:ELO:TUNing:NBW | None |
| <strong>Max Iterations</strong> | SENSe:MIXer:ELO:TUNing:ITERations | MaxPreciseTuningIterations |
| <strong>Tolerance</strong> | SENSe:MIXer:ELO:TUNing:TOLerance | PreciseTuningTolerance |</p>
<table>
<thead>
<tr>
<th>LO Frequency Delta</th>
<th>SENSE:MIXer:ELO:LO:DELTA</th>
<th>LO Frequency Delta</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>SENSE:MIXer:ELO:RESet</td>
<td>Reset LO Frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reset Tuning Parameters</td>
</tr>
<tr>
<td>Input</td>
<td>SENSE:DISTortion:SWEep:CARRier:FREQuency</td>
<td>None</td>
</tr>
<tr>
<td>Output</td>
<td>SENSE:MIXer:OUTPut:FREQuency:SIDE</td>
<td>Output Sideband</td>
</tr>
<tr>
<td>Fractional Multiplier - Numerator</td>
<td>SENSE:MIXer:LO:FREQuency:NUMerator</td>
<td>LO Numerator</td>
</tr>
<tr>
<td>Fractional Multiplier - Denominator</td>
<td>SENSE:MIXer:LO:FREQuency:DENominator</td>
<td>LODenominator</td>
</tr>
<tr>
<td>LO/LO1/LO2</td>
<td>SENSE:MIXer:LO:FREQuency:FIXed</td>
<td>LO Fixed Frequency</td>
</tr>
<tr>
<td>IF</td>
<td>SENSE:MIXer:IF:FREQuency:SIDE</td>
<td>IF Sideband</td>
</tr>
<tr>
<td>Source Name</td>
<td>Source Command</td>
<td>Parameter</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Name</td>
<td>SENSE:MIXer:LO:NAME</td>
<td>LOName</td>
</tr>
<tr>
<td></td>
<td>SENSE:ROLE:DEVICE</td>
<td>RoleDevice</td>
</tr>
<tr>
<td></td>
<td>SENSE:ROLE:CATALOG?</td>
<td>DefinedRoles</td>
</tr>
<tr>
<td>Power</td>
<td>SENSE:MIXer:LO:POWER</td>
<td>LOPower</td>
</tr>
<tr>
<td>Leveling</td>
<td>SOUR:POWER:ALC[:MODE]</td>
<td>ALCEncodingMode</td>
</tr>
<tr>
<td>Attenuator</td>
<td>SOUR:POWER:ATTenuation</td>
<td>chan.Attenuator</td>
</tr>
<tr>
<td>Save...</td>
<td>SENSE:MIXer:SAVE</td>
<td>SaveFile</td>
</tr>
<tr>
<td>Load...</td>
<td>SENSE:MIXer:LOAD</td>
<td>LoadFile</td>
</tr>
</tbody>
</table>

### Modulate Tab

<table>
<thead>
<tr>
<th>Source</th>
<th>Source Command</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SENSE:DISTortion:MODulate:SOURce</td>
<td>None</td>
</tr>
<tr>
<td>Add Source...</td>
<td>SENSE:ROLE:DEVICE</td>
<td>AssignSourceToRole</td>
</tr>
<tr>
<td>Load File...</td>
<td>SOUR:MODulation:FILE:LOAD</td>
<td>None</td>
</tr>
<tr>
<td>Create...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Edit...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Properties...</td>
<td>SOURce:MODulation:FILE:CORRection:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Enable Modulation</td>
<td>SOURce:MODulation[:STATe]</td>
<td>IsEnhancedModulationControl</td>
</tr>
<tr>
<td>Enable Source Correction</td>
<td>SOURce:MODulation:CORRection[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Enable LO Feedthru Monitor</td>
<td>SENSe:SA:COHerence:LO:FTHRu:MONitor[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Enable Pulse</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Source Cal...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LO Monitor...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable LO Feedthru Monitor</td>
<td>SENSe:SA:COHerence:LO:FTHRu:MONitor[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td><strong>Threshold Type</strong></td>
<td><code>SENSe:SA:COHerence:LO:FTHRu:MONitor:TYPE</code></td>
<td>None</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td><strong>Threshold Level</strong></td>
<td><code>SENSe:SA:COHerence:LO:FTHRu:MONitor:TOLerance</code></td>
<td>None</td>
</tr>
<tr>
<td><strong>Measure LO at</strong></td>
<td><code>SENSe:SA:COHerance:LO:FTHRu:MONitor:RECeiver</code></td>
<td>None</td>
</tr>
<tr>
<td><strong>Create/Edit Mod File...</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Modulation Type</strong></td>
<td><code>SOURce:MODulation:FILE:TYPE</code></td>
<td>None</td>
</tr>
<tr>
<td><strong>Source Name</strong></td>
<td><code>SENSe:DISTortion:MODulate:SOURce</code></td>
<td>None</td>
</tr>
<tr>
<td><strong>Sample Rate</strong></td>
<td><code>SOURce:MODulation:FILE:SIGNal:SRATe</code></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td><code>SOURce:MODulation:FILE:SIGNal:SRATe:AUTO</code></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td><code>SOURce:MODulation:FILE:SIGNal:SRATe:CALCulated?</code></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td><code>SOURce:MODulation:FILE:SIGNal:COMPact:OFILe:SRATe</code></td>
<td>None</td>
</tr>
<tr>
<td><strong>Filename</strong></td>
<td><code>SOURce:MODulation:FILE:SIGNal:COMPact:OFILe</code></td>
<td>None</td>
</tr>
<tr>
<td><strong>Signal Span</strong></td>
<td><code>SOURce:MODulation:FILE:SIGNal:SPAN</code></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td><code>SOURce:MODulation:FILE:SIGNal:SPAN:PRIority</code></td>
<td>None</td>
</tr>
<tr>
<td><strong>Tone Spacing</strong></td>
<td><code>SOURce:MODulation:FILE:SIGNal:TONE:SPACing</code></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td><code>SOURce:MODulation:FILE:SIGNal:TONE:SPACing:PRIority</code></td>
<td>None</td>
</tr>
<tr>
<td>Parameter</td>
<td>Value</td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Number of Tones</td>
<td>SOURc:MODulation:FILE:SIGNal:TONE:NUMBer</td>
<td></td>
</tr>
<tr>
<td>Peak-to-Avg</td>
<td>SOURc:MODulation:FILE:SIGNal:COMPact:PAVG?</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SOURc:MODulation:FILE:SIGNal:PAVG:CALCulated?</td>
<td></td>
</tr>
<tr>
<td>Carrier Offset</td>
<td>SOURc:MODulation:FILE:SIGNal:CARRier:OFFSET</td>
<td></td>
</tr>
<tr>
<td>Phase Type</td>
<td>SOURc:MODulation:FILE:SIGNal:PHASE:TYPE</td>
<td></td>
</tr>
<tr>
<td>Nmbr of Notches</td>
<td>SOURc:MODulation:FILE:SIGNal:NPR:NOTCh:NUMBer</td>
<td></td>
</tr>
<tr>
<td>Notch Location</td>
<td>SOURc:MODulation:FILE:SIGNal:NPR:NOTCh:LOCation</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Notch1 Span</td>
<td>SOURce:MODulation:FILE:SIGNal:NPR:NOTCh:SPAN</td>
<td>None</td>
</tr>
<tr>
<td>Notch1 Offset</td>
<td>SOURce:MODulation:FILE:SIGNal:NPR:NOTCh:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>Signal Start Time</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:TIME:STARt</td>
<td>None</td>
</tr>
<tr>
<td>Number of Files</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:FILE:NUMBer</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:FILE:SELect</td>
<td>None</td>
</tr>
<tr>
<td>DAC Scaling</td>
<td>SOURce:MODulation:FILE:SIGNal:DAC:SCALing</td>
<td>None</td>
</tr>
<tr>
<td>Enable Optimizer</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:ENABLE</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Setup... (Optimizer Setup)</td>
<td></td>
</tr>
<tr>
<td>Reject up to Harmonic Number</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:HREJect</td>
<td>None</td>
</tr>
<tr>
<td>Enable Brick-wall Filter For</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:FILTer:ENABLE</td>
<td>None</td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Spectral Leakage</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Tolerance Type</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Min Number of Tones</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:MIN:TONE:NUMBer</td>
<td>None</td>
</tr>
<tr>
<td>Max Tone Spacing</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:MAX:TONE:SPacing</td>
<td>None</td>
</tr>
<tr>
<td>Display</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Number of Samples</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Calculated Sample Rate</td>
<td>SOURce:MODulation:FILE:SIGNal:SRATe:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td>Measurement Time</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Filename</td>
<td>SOURce:MODulation:FILE:LOAD</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SOURce:MODulation:FILE:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Action</td>
<td>Command</td>
<td>None</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Calculate None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Save...</td>
<td>SOURce:MODulation:FILE:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Recall...</td>
<td>SOURce:MODulation:FILE:LOAD</td>
<td>None</td>
</tr>
<tr>
<td>Edit... (Edit Multitone)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tone</td>
<td>SOURce:MODulation:FILE:TONE:COUNt?</td>
<td>None</td>
</tr>
<tr>
<td>Frequency</td>
<td>SOURce:MODulation:FILE:TONE:FREQuency?</td>
<td>None</td>
</tr>
<tr>
<td>Power (dBm)</td>
<td>SOURce:MODulation:FILE:TONE:POWer</td>
<td>None</td>
</tr>
<tr>
<td>Phase (deg)</td>
<td>SOURce:MODulation:FILE:TONE:PHASe</td>
<td>None</td>
</tr>
<tr>
<td>State</td>
<td>SOURce:MODulation:FILE:TONE[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Set All Tones</td>
<td>SOURce:MODulation:FILE:TONE:ALL[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Display</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Go To Row</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Save...</td>
<td>SOURce:MODulation:FILE:TONE:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Load...</td>
<td>SOURce:MODulation:FILE:TONE:LOAD</td>
<td>None</td>
</tr>
<tr>
<td>Defaults</td>
<td>SOURce:MODulation:FILE:INITialize</td>
<td>None</td>
</tr>
<tr>
<td>Properties...</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Measure Tab**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measurement Type</td>
<td>SENSE:DISTortion:MEASure:BAND:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Autofill</td>
<td>SENSE:DISTortion:MEASure:BAND:AUTofill</td>
<td>None</td>
</tr>
<tr>
<td>Carrier</td>
<td>SENSE:DISTortion:SWEep:CARRier:FREQuency</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSE:DISTortion:MEASure:BAND:CARRier:IBW</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSE:DISTortion:MEASure:BAND:CARRier:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>ACPLo</td>
<td>SENSE:DISTortion:MEASure:BAND:ACP:LOWer:IBW</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSE:DISTortion:MEASure:BAND:ACP:LOWer:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>ACPUp</td>
<td>SENSE:DISTortion:MEASure:BAND:ACP:UPPer:IBW</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSE:DISTortion:MEASure:BAND:ACP:UPPer:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>Notch</td>
<td>SENSE:DISTortion:MEASure:BAND:NOTCh:IBW</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSE:DISTortion:MEASure:BAND:NOTCh:OFFSet</td>
<td>None</td>
</tr>
</tbody>
</table>
## Measurement Details

| Parameter                          | Command                                  | Default
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Distortion Measurement</td>
<td>SENSE:DISTortion:MEASure:CORRelation:APERture</td>
<td>None</td>
</tr>
<tr>
<td>Correlation Aperture</td>
<td>SENSE:DISTortion:MEASure:CORRelation:APERture:AUTO[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>ADC Anti-alias Filter</td>
<td>SENSE:DISTortion:ADC:FILTer:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Modulation Filter</td>
<td>SENSE:DISTortion:MEASure:FILTer</td>
<td>None</td>
</tr>
<tr>
<td>Alpha</td>
<td>SENSE:DISTortion:MEASure:FILTer:ALPHA</td>
<td>None</td>
</tr>
<tr>
<td>Symbol Rate</td>
<td>SENSE:DISTortion:MEASure:FILTer:SRATE</td>
<td>None</td>
</tr>
<tr>
<td>EVM Normalization</td>
<td>SENSE:DISTortion:EVM:NORMALize</td>
<td>None</td>
</tr>
<tr>
<td>Nominal DUT NF</td>
<td>SENSE:DISTortion:PATH:DUT:NOMinal:NF</td>
<td>None</td>
</tr>
</tbody>
</table>

## ADVANCED - Measurement Band Table

| Action       | Command                                  | Default
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Add Band</td>
<td>SENSE:DISTortion:MEASure:BAND:ADD</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSE:DISTortion:MEASure:BAND:COUN?</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SENSE:DISTortion:MEASure:BAND:NAMe</td>
<td>None</td>
</tr>
<tr>
<td>Delete Band</td>
<td>SENSE:DISTortion:MEASure:BAND:DELETE</td>
<td>None</td>
</tr>
<tr>
<td>Model</td>
<td>Module</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>Model 1</td>
<td>Module 1</td>
<td>Description 1</td>
</tr>
<tr>
<td>Model 2</td>
<td>Module 2</td>
<td>Description 2</td>
</tr>
<tr>
<td>Model 3</td>
<td>Module 3</td>
<td>Description 3</td>
</tr>
</tbody>
</table>

**Autofill from Mod File**
- None

**Reset Bands to defaults**
- SENSe:DISTortion:MEASure:BAND:INITialize

### Meas Class...
- CALCulate:MEASure:DEFine
- CreateCustomMeasurementEx

### Quick Start...
- S-Param
  - CALCulate:MEASure:PARameter
  - CreateSParameterEx
- Balanced
  - CALCulate:MEASure:PARameter
  - BalSMeasurement
  - BBalMeasurement
  - SBalMeasurement
  - SSBMeasurement
- Other
  - CALCulate:MEASure:PARameter
  - CreateSParameterEx
**CF_Setup Commands - NF**

**Setup Commands - Noise Figure Measurement Class**

Only the Main and Layout Setup commands corresponding to the Noise Figure measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking [here](#) or by clicking on the softtabs on the graphic below.

Click [here](#) to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td><strong>NF Setup...</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>--------------------</td>
</tr>
<tr>
<td>IF Bandwidth</td>
</tr>
<tr>
<td>Start</td>
</tr>
<tr>
<td>Stop</td>
</tr>
<tr>
<td>Center</td>
</tr>
<tr>
<td>Span</td>
</tr>
<tr>
<td><strong>Power</strong></td>
</tr>
<tr>
<td>Power On (All Channels)</td>
</tr>
<tr>
<td>DUT Input Port</td>
</tr>
<tr>
<td>Input Port</td>
</tr>
<tr>
<td>Power Level</td>
</tr>
<tr>
<td>Source Leveling Mode</td>
</tr>
<tr>
<td>DUT Output Port</td>
</tr>
<tr>
<td>Output Port</td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>----------------------------------------</td>
</tr>
<tr>
<td>Power Level</td>
</tr>
<tr>
<td>Source Leveling Mode</td>
</tr>
<tr>
<td>Path Configuration...</td>
</tr>
<tr>
<td><strong>Noise Figure</strong></td>
</tr>
<tr>
<td>Bandwidth/Average</td>
</tr>
<tr>
<td>Noise Bandwidth</td>
</tr>
<tr>
<td>Average Number</td>
</tr>
<tr>
<td>Average ON</td>
</tr>
<tr>
<td>Use Narrowband Compensation</td>
</tr>
<tr>
<td><strong>Noise Receiver</strong></td>
</tr>
<tr>
<td>NA Receiver (Port 2)</td>
</tr>
<tr>
<td>Noise Receiver</td>
</tr>
<tr>
<td>Receiver Gain</td>
</tr>
<tr>
<td>Ambient Temperature</td>
</tr>
<tr>
<td>---------------------</td>
</tr>
<tr>
<td>Impedance States</td>
</tr>
<tr>
<td>Max Acquired</td>
</tr>
<tr>
<td>Impedance States</td>
</tr>
<tr>
<td>Meas Class...</td>
</tr>
<tr>
<td>Quick Start...</td>
</tr>
<tr>
<td>Balanced</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Other</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Layout Tab Commands</td>
</tr>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>New Trace</td>
</tr>
<tr>
<td>New Channel</td>
</tr>
<tr>
<td>New Window</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td>New Sheet</td>
</tr>
<tr>
<td>Delete</td>
</tr>
<tr>
<td>TrN</td>
</tr>
<tr>
<td>ChN</td>
</tr>
<tr>
<td>WinN</td>
</tr>
<tr>
<td>Select</td>
</tr>
<tr>
<td>TrN</td>
</tr>
<tr>
<td>ChN</td>
</tr>
<tr>
<td>WinN</td>
</tr>
<tr>
<td>Measure...</td>
</tr>
<tr>
<td>Meas Class...</td>
</tr>
</tbody>
</table>
CF_Setup Commands - NFX

Setup Commands - Noise Figure Converters Measurement Class

Only the Main and Layout Setup commands corresponding to the Noise Figure Converters measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td><strong>NFX Setup...</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

7637
<table>
<thead>
<tr>
<th>IF Bandwidth</th>
<th>SENSe:BWIDth[:RESolution]</th>
<th>IFBandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td>SENSe:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td>SENSe:FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>Center</td>
<td>SENSe:FREQuency:CENTer</td>
<td>CenterFrequency</td>
</tr>
<tr>
<td>Span</td>
<td>SENSe:FREQuency:SPAN</td>
<td>FrequencySpan</td>
</tr>
</tbody>
</table>

### Power

<table>
<thead>
<tr>
<th>Power On (All Channels)</th>
<th>OUTPut[:STATe]</th>
<th>SourcePowerState</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port Powers Coupled</td>
<td>SOURce:POWer:COUPle</td>
<td>CouplePorts</td>
</tr>
</tbody>
</table>

### DUT Input Port

<table>
<thead>
<tr>
<th>Input Port</th>
<th>SENSe:NOISe:PMAP</th>
<th>SetPortMap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Level</td>
<td>SOURce:POWER[:LEVel][:IMMediate][:AMPLitude]</td>
<td>InputLinearPowerLevel</td>
</tr>
<tr>
<td>Source Leveling Mode</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
</tbody>
</table>

### DUT Output Port
<table>
<thead>
<tr>
<th>Output Port</th>
<th>SENSE:NOISe:PMAP</th>
<th>SetPortMap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Level</td>
<td>SOURce:POWer[:LEVEL][:IMMediate][:AMPLitude]</td>
<td>InputLinearPowerLevel</td>
</tr>
<tr>
<td>Source Leveling Mode</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
<tr>
<td>Path Configuration...</td>
<td>SENSepATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
</tr>
</tbody>
</table>

**Noise Figure**

**Bandwidth/Average**

<table>
<thead>
<tr>
<th>Noise Bandwidth</th>
<th>SENSepNOISe:BWIDth[:RESolution]</th>
<th>NoiseBandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Number</td>
<td>SENSepNOISe:AVERage[:COUNT]</td>
<td>NoiseAverageFactor</td>
</tr>
<tr>
<td>Average ON</td>
<td>SENSepNOISe:AVERage:STATe</td>
<td>NoiseAverageState</td>
</tr>
<tr>
<td>Use Narrowband Compensation</td>
<td>SENSepNOISe:NARrowband[:STATe]</td>
<td>NarrowBand</td>
</tr>
</tbody>
</table>

**Noise Receiver**

<table>
<thead>
<tr>
<th>NA Receiver (Port 2)</th>
<th>SENSepNOISe:RECeiver</th>
<th>NoiseReceiver</th>
</tr>
</thead>
<tbody>
<tr>
<td>Noise Receiver</td>
<td>SENSepNOISe:RECeiver</td>
<td>NoiseReceiver</td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>--------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Receiver Gain</td>
<td>SENSE:NOISE:GAIN</td>
<td>NoiseGain</td>
</tr>
<tr>
<td>Ambient Temperature</td>
<td>SENSE:NOISE:TEMPerature:AMBient</td>
<td>AmbientTemperature</td>
</tr>
<tr>
<td>Impedance States</td>
<td></td>
<td>ImpedanceStates</td>
</tr>
<tr>
<td>Max Acquired Impedance States</td>
<td>SENSE:NOISE:IMPedance:COUNt</td>
<td>ImpedanceStates</td>
</tr>
<tr>
<td>Enable Source Pulling for SParameters</td>
<td>SENSE:NOISE:PULL[:STATe]</td>
<td>SourcePullForSParameters</td>
</tr>
<tr>
<td>Mixer Frequency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input</td>
<td>SENSE:MIXer:CALCulate</td>
<td>Calculate</td>
</tr>
<tr>
<td>LO1</td>
<td>SENSE:MIXer:LO:FREQuency:FIXed</td>
<td>LOFixedFrequency</td>
</tr>
<tr>
<td>Input &gt; LO</td>
<td>SENSE:MIXer:LO:FREQuency:ILTI</td>
<td>IsInputGreaterThanLO</td>
</tr>
<tr>
<td>Output</td>
<td>SENSE:MIXer:CALCulate</td>
<td>Calculate</td>
</tr>
<tr>
<td>Mixer Power</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power On (All Channels)</td>
<td>OUTPut[:STATe]</td>
<td>SourcePowerState</td>
</tr>
<tr>
<td>LO1 Power</td>
<td>SENSE:MIXer:LO:POWer</td>
<td>LOPower</td>
</tr>
<tr>
<td>LO2 Power</td>
<td>SENSE:MIXer:LO:POWer</td>
<td>LOPower</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>LO1 Swept Power</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>SENSE:MIXer:LO:FREQuency:STARt</td>
<td>LOStartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td>SENSE:MIXer:LO:FREQuency:STOP</td>
<td>LOSTopFrequency</td>
</tr>
<tr>
<td>Step</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>LO2 Swept Power</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>SENSE:MIXer:LO:FREQuency:STARt</td>
<td>LOStartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td>SENSE:MIXer:LO:FREQuency:STOP</td>
<td>LOSTopFrequency</td>
</tr>
<tr>
<td>Step</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Path Configuration...</td>
<td>SENSE:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td>Mixer Setup</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Converter Stages</td>
<td>SENSE:MIXer:STAGe</td>
<td>LOStage</td>
</tr>
<tr>
<td>Add Source...</td>
<td>SENSE:ROLE:DEVice</td>
<td>AssignSourceToRole</td>
</tr>
<tr>
<td>Path Configuration...</td>
<td>SENSE:PATH:CONFig:ELEMent[:STATE]</td>
<td>Element</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Save...</td>
<td>SENSE:MIXer:SAVE</td>
<td>SaveFile</td>
</tr>
<tr>
<td>Load...</td>
<td>SENSE:MIXer:LOAD</td>
<td>LoadFile</td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFINE</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Quick Start...</td>
<td>S-Param</td>
<td>CreateSParameterEx</td>
</tr>
<tr>
<td>Balanced</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BalSMeasurement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BBalMeasurement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SBalMeasurement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SSBMeasurement</td>
</tr>
<tr>
<td>Other</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
</tr>
</tbody>
</table>

**Layout Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Trace</td>
<td></td>
<td>DISPLAY:WINDow:TRACe[:STATE]</td>
<td>View</td>
</tr>
<tr>
<td>New Channel</td>
<td>None</td>
<td>chans.Add</td>
<td></td>
</tr>
<tr>
<td>New Window</td>
<td>DISPlay:WINDow[:STATe]</td>
<td>Add</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>------------------------</td>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>New Sheet</td>
<td>DISPlay:SHEet:STATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Delete</td>
<td>DISPlay:WINDow:TRACe:DELeTe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SYSTem:CHANnels:DELeTe</td>
<td>RemoveChannelNumber</td>
<td></td>
</tr>
<tr>
<td>Select</td>
<td>DISPlay:WINDow[;STATe]</td>
<td>Add</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DISPlay:WINDow:TRACe:SELeCt</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DISPlay:WINDow:TRACe:SELeCt</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Measure...</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
<td></td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
</tbody>
</table>
CF_Setup Commands - PN

Setup Commands - Phase Noise Measurement Class

Only the Main Setup commands corresponding to the Phase Noise measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td><strong>Phase Noise Setup...</strong></td>
</tr>
<tr>
<td>Noise Type</td>
</tr>
<tr>
<td>Phase Noise</td>
</tr>
<tr>
<td>Residual Noise</td>
</tr>
<tr>
<td>Carrier Frequency</td>
</tr>
<tr>
<td>Carrier Threshold</td>
</tr>
<tr>
<td>Start Offset</td>
</tr>
<tr>
<td>Setting</td>
</tr>
<tr>
<td>-------------------------------</td>
</tr>
<tr>
<td>Stop Offset</td>
</tr>
<tr>
<td>RBW Ratio</td>
</tr>
<tr>
<td>FFT Avg Factor</td>
</tr>
<tr>
<td>Noise Mode</td>
</tr>
</tbody>
</table>

### RF Path Tab

<table>
<thead>
<tr>
<th>Setting</th>
<th>Command</th>
<th>Setting Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>VNA Input</td>
<td>SENSE: PN:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>DUT Input (for Residual Phase)</td>
<td>SENSE: PN:RESidual:INPut</td>
<td>None</td>
</tr>
<tr>
<td>DUT Output (for Residual Phase)</td>
<td>SENSE: PN:RESidual:OUTput</td>
<td>None</td>
</tr>
<tr>
<td>Rcvr Atten</td>
<td>SOURce:POWER:ATTenuation:RECeiver:TEST</td>
<td>None</td>
</tr>
<tr>
<td>RF Path Configuration...</td>
<td>SENSE: PATH:CONFig:ELEMent[:STATE]</td>
<td>Element</td>
</tr>
</tbody>
</table>

### Source Tab

<table>
<thead>
<tr>
<th>Setting</th>
<th>Command</th>
<th>Setting Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power On (All Channels)/State</td>
<td>SOURce:POWER:MODE</td>
<td>SourcePortMode</td>
</tr>
<tr>
<td>Port Powers Coupled</td>
<td><strong>SOURce:POWer:COUPle</strong> CouplePorts</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>--------------------------------------</td>
<td></td>
</tr>
<tr>
<td>State</td>
<td><strong>SOURce:POWer:MODE</strong> SourcePortMode</td>
<td></td>
</tr>
<tr>
<td>Frequency</td>
<td><strong>SOURce:FREQuency[:FIXed]</strong> None</td>
<td></td>
</tr>
<tr>
<td>Power</td>
<td><strong>SOURce:POWer[:LEVel][:IMMediate][:AMPLitude]</strong> TestPortPower</td>
<td></td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RF Path Configuration..</td>
<td><strong>SENSe:PATH:CONFig:ELEMent[:STATe]</strong> Element</td>
<td></td>
</tr>
<tr>
<td>Power and Attenuator...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>External Devices...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spurious Tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Show Spurious Table</td>
<td><strong>DISPlay:WINDow:TABLE:SPURious:ENABle</strong> None</td>
<td></td>
</tr>
<tr>
<td>Table Sort Order</td>
<td>CALCulate:MEASure:PN:SPURious:SORT</td>
<td>None</td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Select Trace</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementE</td>
</tr>
<tr>
<td>Enable Spur Analysis</td>
<td>CALCulate:MEASure:PN:SPURious:ANALysis[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Spur Sensibility</td>
<td>CALCulate:MEASure:PN:SPURious:SENSibility</td>
<td>None</td>
</tr>
<tr>
<td>Min. Spur Level</td>
<td>CALCulate:MEASure:PN:SPURious:THReshold:LEVel:MINimum</td>
<td>None</td>
</tr>
<tr>
<td>Omit Displayed Spur</td>
<td>CALCulate:MEASure:PN:SPURious:OMISSION[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Integrated Noise Tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Show Integrated Noise Table</td>
<td>DISPLAY:WINDOW:TABLE:INOise:ENABLE</td>
<td>None</td>
</tr>
<tr>
<td>Weighting Filter Setup...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Frequency</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>-----------</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>Weighting Value</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Add</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Delete</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Delete All</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Save Table</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Load Table</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Select Trace</td>
<td>CALC:MEAS:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Range</td>
<td>Specified in each CALC:MEAS:PN:INT:RANG[1-4]... command</td>
<td>None</td>
</tr>
<tr>
<td>Type</td>
<td>CALC:MEAS:PN:INT:RANG[1-4]:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Start</td>
<td>CALC:MEAS:PN:INT:RANG[1-4]:START</td>
<td>None</td>
</tr>
<tr>
<td>Stop</td>
<td>CALC:MEAS:PN:INT:RANG[1-4]:STOP</td>
<td>None</td>
</tr>
<tr>
<td>Weighting Filter</td>
<td>CALC:MEAS:PN:INT:RANG[1-4]:WEIGHTing</td>
<td>None</td>
</tr>
<tr>
<td>Spot Noise Tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Show Spot Noise Table</strong></td>
<td><strong>DISPlay:WINDow:TABLE:SNOise:ENABLe</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Select Traces</strong></td>
<td><strong>CALCulate:MEASure:DEFine</strong></td>
<td></td>
</tr>
</tbody>
</table>
| **Define Spot Frequencies** | **CALCulate:MEASure:PN:SNOise:USER[1-6]:X**  
**CALCulate:MEASure:PN:SNOise:USER[1-6][:STATe]** |
| **Decade Edges** | **CALCulate:MEASure:PN:SNOise:DECades[:STATe]** |
CF_Setup Commands - SA

Setup Commands - Spectrum Analyzer Measurement Class

Only the Main and Layout Setup commands corresponding to the Spectrum Analyzer measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

Click here to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>SA Setup..</td>
</tr>
<tr>
<td>Sweep Type</td>
</tr>
<tr>
<td>Start</td>
</tr>
<tr>
<td>Stop</td>
</tr>
<tr>
<td>Center</td>
</tr>
<tr>
<td>Span</td>
</tr>
<tr>
<td>Number of Points</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Video Bandwidth</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Detector Type</td>
</tr>
<tr>
<td>Detector Bypass</td>
</tr>
<tr>
<td>Averaging Type</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Receiver Attenuator s</td>
</tr>
<tr>
<td>----------------------</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Tab</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power On (All Channels)</td>
</tr>
<tr>
<td>Port Powers Coupled</td>
</tr>
<tr>
<td>State</td>
</tr>
<tr>
<td>None</td>
</tr>
<tr>
<td>Type</td>
</tr>
<tr>
<td>SourceSweepType2</td>
</tr>
<tr>
<td>Frequency</td>
</tr>
<tr>
<td>SENSE:SA:SOURce:FREQuency:STOP</td>
</tr>
<tr>
<td>Power</td>
</tr>
<tr>
<td>SENSE:SA:SOURce:POWer:STOP</td>
</tr>
<tr>
<td>SENSE:SA:SOURce:POWer[:VALue]</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>Phase</td>
</tr>
<tr>
<td>Pulse Setup...</td>
</tr>
<tr>
<td>IQMod - Modulation settings dialog</td>
</tr>
<tr>
<td>Enable Modulation</td>
</tr>
<tr>
<td>Modulation Filename</td>
</tr>
<tr>
<td>Show File Properties</td>
</tr>
<tr>
<td>Modulation File Name</td>
</tr>
<tr>
<td>Source Name</td>
</tr>
<tr>
<td>Number of Samples</td>
</tr>
<tr>
<td>Modulation Type</td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>Signal Span</td>
</tr>
<tr>
<td>Phase Type</td>
</tr>
<tr>
<td>-----------------------------</td>
</tr>
<tr>
<td>Number of Notches</td>
</tr>
<tr>
<td>Notch Location</td>
</tr>
<tr>
<td>Notch1 Span</td>
</tr>
<tr>
<td>Receiver RBW</td>
</tr>
<tr>
<td>Measurement Time</td>
</tr>
<tr>
<td>Create/Edit Mod File...</td>
</tr>
<tr>
<td>Feature</td>
</tr>
<tr>
<td>-------------------------------------</td>
</tr>
<tr>
<td>Autoset Frequencies and Coherence for current modulation</td>
</tr>
<tr>
<td>Immediate Autoset</td>
</tr>
<tr>
<td>Autoset NPR Markers</td>
</tr>
<tr>
<td>NPR Band Guard</td>
</tr>
<tr>
<td>Autoset ACPR Markers</td>
</tr>
<tr>
<td>ACPR Band Guard</td>
</tr>
<tr>
<td>Enable Modulation Correction</td>
</tr>
<tr>
<td>Calibrate Modulation...</td>
</tr>
<tr>
<td>Source Power</td>
</tr>
<tr>
<td>-------------</td>
</tr>
<tr>
<td>Source Path Amplification (= Power Offset)</td>
</tr>
<tr>
<td>For SA Analysis Marker commands for getting measurement data, refer to SA Analysis Tab Commands.</td>
</tr>
</tbody>
</table>
| RF source sweep order | SENSE:SA:SOURce:SWEep:FIRst[:DIMension] | SourceSweepFirstDimensio
<table>
<thead>
<tr>
<th>Path Configuration...</th>
<th>SENSE:PATH:CONFig:ELEMent[:STATe]</th>
<th>Element</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power and Attenuator.. .</td>
<td></td>
<td></td>
</tr>
<tr>
<td>External Devices...</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Coherence Tab</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable multitone</td>
<td>SENSE:SA:COHerence:MULTitone[:STATe]</td>
<td>MultiToneEnable</td>
</tr>
<tr>
<td>Tone Spacing</td>
<td>SENSE:SA:COHerence:MULTitone:SPACing</td>
<td>MultiToneSpacing</td>
</tr>
<tr>
<td>Waveform Period</td>
<td>SENSE:SA:COHerence:MULTitone:PERiod</td>
<td>MultiTonePeriod</td>
</tr>
<tr>
<td>Reference Tone</td>
<td>SENSE:SA:COHerence:MULTitone:REFerence</td>
<td>MultiToneReference</td>
</tr>
<tr>
<td>Feature</td>
<td>Command Details</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>------------------------------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>Reject up to harmonic</td>
<td>SENSE:SA:COHerence:MULTitone:HREJect</td>
<td>MultiToneHarmonicRejection</td>
</tr>
<tr>
<td>Nyquist protect order</td>
<td>SENSE:SA:COHerence:MULTitone:NYQReject</td>
<td>MultiToneNyquistProtection</td>
</tr>
<tr>
<td>Vector averaging</td>
<td>SENSE:SA:COHerence:VECTor:AVERage[:STATE]</td>
<td>VectorAverageEnable</td>
</tr>
<tr>
<td></td>
<td>SENSE:SA:COHerence:VECTor:AVERage:VALue</td>
<td>VectorAverageValue</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VectorAverageValueMax</td>
</tr>
<tr>
<td>Data Display</td>
<td>SENSE:SA:COHerence:MULTitone:DATA</td>
<td>MultiToneDataDisplay</td>
</tr>
<tr>
<td>Multitone settings are valid</td>
<td>SENSE:SA:COHerence:MULTitone:VALid</td>
<td>MultiToneSettingsValid</td>
</tr>
<tr>
<td>Compute Phases</td>
<td>SENSE:SA:COHerence:PHASe[:STATE]</td>
<td>PhaseProcessState</td>
</tr>
<tr>
<td>Display Phases if Tone Power &gt;</td>
<td>SENSE:SA:COHerence:PHASe:DISPLAY:LEVEL</td>
<td>PhaseDisplayMinLevel</td>
</tr>
</tbody>
</table>
### Trig. & Pulse Tab

<table>
<thead>
<tr>
<th>Advanced Trigger Mode</th>
<th>SENSE:SA:TRIGer:LEVel[:STATe]</th>
<th>TriggerADCLevelState</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC Level</td>
<td>SENSE:SA:TRIGer:LEVel[:STATe]</td>
<td>TriggerADCLevelState</td>
</tr>
<tr>
<td></td>
<td>SENSE:SA:TRIGer:LEVel:VALue</td>
<td>TriggerADCLevelValue</td>
</tr>
<tr>
<td>Periodic Counter</td>
<td>SENSE:SA:TRIGer:PERCounter[:STATe]</td>
<td>TriggerPeriodicCounter State</td>
</tr>
<tr>
<td></td>
<td>SENSE:SA:TRIGer:PERCounter:VALue</td>
<td>TriggerPeriodicCounter Value</td>
</tr>
</tbody>
</table>

### Advanced Tab

<table>
<thead>
<tr>
<th>RBW Shape</th>
<th>SENSE:SA:BANDwidth:SHAPe</th>
<th>BandwidthShape</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image Reject Type</td>
<td>SENSE:SA:IMAGe:REJect</td>
<td>ImageRejectMethod</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>Image Reject Strength</td>
<td>SENSe:SA:IMAGe:STRENgth</td>
<td>ImageRejectStrength</td>
</tr>
<tr>
<td>RBW/VBW</td>
<td>SENSe:SA:BANDwidth:VIDeo:RATio</td>
<td>ResolutionBWVideoBWRatio</td>
</tr>
<tr>
<td>Span/RBW</td>
<td>SENSe:SA:FREQuency:SPAN:BANDwidth[:RESolution]:RATio</td>
<td>SpanResolutionBWRatio</td>
</tr>
<tr>
<td>Occupied BW search min</td>
<td>SENSe:SA:BANDwidth:SEARch:OCCupied:MIN</td>
<td>SearchOccupiedBWMinFreq</td>
</tr>
<tr>
<td>Enable DC Outputs</td>
<td>SOURce:DC:ENABle</td>
<td>EnableAllOutput</td>
</tr>
<tr>
<td>Enable DC Sweep</td>
<td>SENSe:SA:SOURce:DC:SWEep[:STATe]</td>
<td>DCSourceSweepState</td>
</tr>
<tr>
<td>Number of DC levels</td>
<td>SENSe:SA:SOURce:DC:SWEep:POINt</td>
<td>DCSourcePointCount</td>
</tr>
<tr>
<td>DC Sources...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Embedded LO</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>IF Tab</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADC Filter</td>
<td><strong>SENSe:SA:ADC:FILTer</strong></td>
<td><strong>ADCFilter</strong></td>
</tr>
<tr>
<td>ADC Filter Auto</td>
<td><strong>SENSe:SA:ADC:FILTer:AUTO</strong></td>
<td><strong>EnableADCFilterAuto</strong></td>
</tr>
<tr>
<td>DFT Bandwidth Auto</td>
<td><strong>SENSe:SA:DFT:BANDwidth:AUTO</strong></td>
<td><strong>AutoBandwidth</strong></td>
</tr>
<tr>
<td>Narrow - DFT Min</td>
<td><strong>SENSe:SA:DFT:BANDwidth:NARRow:MIN</strong></td>
<td><strong>BandwidthNarrowMin</strong></td>
</tr>
<tr>
<td>Narrow - DFT Max</td>
<td><strong>SENSe:SA:DFT:BANDwidth:NARRow:MAX</strong></td>
<td><strong>BandwidthNarrowMax</strong></td>
</tr>
<tr>
<td>Wide - DFT Min</td>
<td><strong>SENSe:SA:DFT:BANDwidth:WIDE:MIN</strong></td>
<td><strong>BandwidthWideMin</strong></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Wide - DFT Max</td>
<td><code>SENSe:SA:DFT:BANDwidth:WIDE:MAX</code></td>
<td>BandwidthWideMax</td>
</tr>
<tr>
<td>IF Gain</td>
<td><code>SENSe:PATH:CONFig:ELEMent[:STATe]</code></td>
<td>Element</td>
</tr>
<tr>
<td>IF Config</td>
<td><code>SENSe:PATH:CONFig:ELEMent[:STATe]</code></td>
<td>Element</td>
</tr>
<tr>
<td><strong>Processing Tab</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFT Type</td>
<td><code>SENSe:SA:DFT:TYPE</code></td>
<td>Type</td>
</tr>
<tr>
<td></td>
<td><code>SENSe:SA:DATA:KEEP:RECeivers:COUNt?</code></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td><code>SENSe:SA:DATA:KEEP:RECeivers[:CURRENT]</code></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td><code>SENSe:SA:DATA:KEEP:RECeivers:LIST</code></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td><code>SENSe:SA:DATA:KEEP:SCALar:GET?</code></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td><code>SENSe:SA:DATA:KEEP[:STATe]</code></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td><code>SENSe:SA:DATA:KEEP:VECTor:GET?</code></td>
<td>None</td>
</tr>
<tr>
<td>Acq. Time for 1 LO</td>
<td><code>SENSe:SA:ADC:ACQTime?</code></td>
<td>AcquisitionTime</td>
</tr>
<tr>
<td>Span LOs count</td>
<td><code>SENSe:SA:LO:COUNt?</code></td>
<td>LOCount</td>
</tr>
<tr>
<td>Span bins count</td>
<td>SENSE:SA:SPAN:BINS:COUNt?</td>
<td>SpanBinsCount</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>DFT resolution</td>
<td>SENSE:SA:DFT:RESolution?</td>
<td>Resolution</td>
</tr>
<tr>
<td>DFT record size</td>
<td>SENSE:SA:DFT:RECord:SIZE?</td>
<td>RecordSize</td>
</tr>
<tr>
<td>Display image reject traces</td>
<td>SENSE:SA:TRACe:IMAGe[:STATe]</td>
<td>EnableImageRejectTraces</td>
</tr>
</tbody>
</table>

**ADC & LO Tab**

<table>
<thead>
<tr>
<th>Sample Frequency</th>
<th>SENSE:SA:ADC:SAMPLE:RATE</th>
<th>ADCSampleRate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SENSE:SA:ADC:SAMPLE:RATE:AUTO</td>
<td>EnableADCSampleRateAuto</td>
</tr>
<tr>
<td>Enable FIR for 25 MHz</td>
<td>SENSE:SA:ADC:SAMPLE:DECimation:FIR</td>
<td>ADCEnableFIRFor25MHz</td>
</tr>
<tr>
<td>Dithering</td>
<td>SENSE:SA:ADC:DITHer[:STATe]</td>
<td>EnableADCDither</td>
</tr>
<tr>
<td>Feature</td>
<td>Command Details</td>
<td>Default Values</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Multiple Recording</td>
<td>SENSE:SA:ADC:MREC[:STATe]</td>
<td>ADCMultRecState</td>
</tr>
<tr>
<td>Chunk size</td>
<td>SENSE:SA:ADC:MREC:SIZE</td>
<td>ADCMultRecSize</td>
</tr>
<tr>
<td>Chunk period</td>
<td>SENSE:SA:ADC:MREC:PERiod</td>
<td>ADCMultRecPeriod</td>
</tr>
<tr>
<td>Randomize LO</td>
<td>SENSE:SA:LO:RANDOM[:STATe]</td>
<td>EnableRandomizedLO</td>
</tr>
<tr>
<td>Enable baseband X-axis mode (LO independent sweep)</td>
<td>SENSE:SA:LO:BASEband[:STATe]</td>
<td>None</td>
</tr>
</tbody>
</table>
| Force LO to frequency | SENSE:SA:LO:FREQ:FORCEe  
SENSe:SA:LO:FREQ:VALue  
SENSe:SA:LO:FORCE:OFFSet:DIVider  
SENSe:SA:LO:FORCE:OFFSet:MULTiplier  
SENSe:SA:LO:FORCE:OFFSet:SOURce | EnableForce  
LOToFrequency  
ForceLOYToFrequency  
None  
None  
None |

**Data Tab**

| Data Format | SENSE:SA:DATA:TYPE | DataFormat |
| Export receivers | SENSE:SA:DATA:RECeivers:LIST  
SENSe:SA:DATA:RECeivers? | ExportReceiverSetList  
ExportReceiverList |
| Don't save data below threshold | SENSE:SA:DATA:THReshold[:STATe]  
SENSe:SA:DATA:THReshold:VALue | DataLevelThresholdEnabled  
DataLevelThreshold |
<table>
<thead>
<tr>
<th>Feature</th>
<th>Command/Query</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFT bins count</td>
<td><code>SENSe:SA:DATA:BINs:COUNT?</code></td>
<td>DataBinCount</td>
</tr>
<tr>
<td>Export to binary file</td>
<td><code>SENSe:SA:DATA:FILE:BINary[:STATe]</code></td>
<td>BinaryFileEnabled</td>
</tr>
<tr>
<td>File name prefix</td>
<td><code>SENSe:SA:DATA:FILE:PREFix</code></td>
<td>FilePrefix</td>
</tr>
<tr>
<td>Record size (bytes)</td>
<td><code>SENSe:SA:DATA:SIZE?</code></td>
<td>DataByteSize</td>
</tr>
<tr>
<td></td>
<td><code>SENSe:SA:DATA:SIZE:BIN?</code></td>
<td>DataBytesPerBin</td>
</tr>
<tr>
<td></td>
<td><code>SENSe:SA:DATA:SIZE:LOW?</code></td>
<td>DataByteSizeLOW</td>
</tr>
<tr>
<td></td>
<td><code>SENSe:SA:DATA:SIZE:HIGH?</code></td>
<td>DataByteSizeHIGH</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>--------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>with data files</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Export all markers to a single file</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Export to FIFO buffer</td>
<td>SENSE:SA:DATA:FIFO[:STATE]</td>
<td>FIFOEnabled</td>
</tr>
<tr>
<td>Share name</td>
<td>SENSE:SA:DATA:SHARed:NAME</td>
<td>MemShareName</td>
</tr>
<tr>
<td></td>
<td>SENSE:SA:DATA:START?</td>
<td>DataFirstRFBin</td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Quick Start...</td>
<td>S-Param</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td></td>
<td>Balanced</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
</tr>
<tr>
<td>----------</td>
<td>----------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>New Trace</td>
<td></td>
<td>DISPLAY:WINDow:TRA Ce[:STATE]</td>
</tr>
<tr>
<td>New Channel</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>New Window</td>
<td></td>
<td>DISPLAY:WINDow[:STATE]</td>
</tr>
<tr>
<td>New Sheet</td>
<td></td>
<td>DISPLAY:SHEet:STATE</td>
</tr>
<tr>
<td>Delete</td>
<td>TrN</td>
<td>DISPLAY:WINDow:TRA Ce:DELETE</td>
</tr>
<tr>
<td></td>
<td>ChN</td>
<td>SYSTEM:CHANnels:DELETE</td>
</tr>
<tr>
<td></td>
<td>WinN</td>
<td>DISPLAY:WINDow[:STATE]</td>
</tr>
<tr>
<td></td>
<td>Select</td>
<td>TrN</td>
</tr>
<tr>
<td>---</td>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>ChN</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>WinN</td>
<td></td>
<td>DISPlay:WINDow:TRACe:SELect</td>
</tr>
<tr>
<td>Measu re...</td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Meas Class...</td>
<td></td>
<td>CALCulate:MEASure:DEFine</td>
</tr>
</tbody>
</table>
CF_Setup Commands - SMC

Setup Commands - Scalar Mixer/Converter + Phase Measurement Class

Only the Main and Layout Setup commands corresponding to the Scalar Mixer/Converter + Phase measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking [here](#) or by clicking on the softtabs on the graphic below.

Click [here](#) to view links to Setup commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMC Setup...</td>
<td>Sweep</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sweep Type</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linear Frequency</td>
<td>SENSe:SWEep:TYPE</td>
<td>SweepType</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CW Time</td>
<td>SENSe:SWEep:TYPE</td>
<td>SweepType</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Segment Sweep</td>
<td>SENSe:SWEep:TYPE</td>
<td>SweepType</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power</td>
<td>SENSe:SWEep:TYPE</td>
<td>SweepType</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X-Axis Point Spacing</td>
<td>SENSe:SEGMenX:SPACing</td>
<td>XAxisPointSpacing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------</td>
<td>--------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reversed Port 2 Coupler</td>
<td>SENSE:MIXer:REVerse</td>
<td>IncludeReverseSweep</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of Points</td>
<td>SENSE:SWEEP:POINts</td>
<td>NumberOfPoints</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IF Bandwidth</td>
<td>SENSE:BWIDth[:RESolution]</td>
<td>IFBandwidth</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phase Reference Point</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable Phase</td>
<td>SENSE:MIXer:PHASE</td>
<td>EnablePhase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>First Point</td>
<td>SENSE:MIXer:NORMalize</td>
<td>NormalizePoint</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Middle Point</td>
<td>SENSE:MIXer:NORMalize</td>
<td>NormalizePoint</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Last Point</td>
<td>SENSE:MIXer:NORMalize</td>
<td>NormalizePoint</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify</td>
<td>SENSE:MIXer:NORMalize</td>
<td>NormalizePoint</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power On (All Channels)</td>
<td>OUTPUT[:,STATe]</td>
<td>SourcePowerState</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port Powers Coupled</td>
<td>SOURce:POWer:COUPle</td>
<td>CouplePorts</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DUT Input Port</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input Port</td>
<td>SENSe:MIXer:PMAP</td>
<td>SetDutPorts</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>------------------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power Level</td>
<td>SOURce:POWer[:LEVel][:IMMediate][:AMPLitude]</td>
<td>InputLinearPowerLevel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source Leveling Mode</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### DUT Output Port

<table>
<thead>
<tr>
<th>Output Port</th>
<th>SENSe:MIXer:PMAP</th>
<th>SetDutPorts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Level</td>
<td>SOURce:POWer[:LEVel][:IMMediate][:AMPLitude]</td>
<td>InputLinearPowerLevel</td>
</tr>
<tr>
<td>Source Leveling Mode</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
</tr>
</tbody>
</table>

### DUT Input Port Power Sweep

<table>
<thead>
<tr>
<th>Start Power</th>
<th>SOURce:POWer:PORT:STARt</th>
<th>StartPowerEx</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stop Power</td>
<td>SOURce:POWer:PORT:STOP</td>
<td>StopPowerEx</td>
</tr>
<tr>
<td>Points</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Power Step</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### DUT Output Port Power Sweep

7673
| Start Power | SOURce:POWer:PORT:STARt | StartPowerEx |
| Stop Power | SOURce:POWer:PORT:STOP | StopPowerEx |
| Path Configuration... | SENSE:PATH:CONFig:ELEMen[t][:STATe] | Element |

### Mixer Frequency

| Input | SENSE:MIXer:CALCulate | Calculate |
| LO1 | SENSE:MIXer:LO:FREQuency:FIXed | LOFixedFrequency |
| Input > LO | SENSE:MIXer:LO:FREQuency:ILTI | IsInputGreaterThanLO |
| Output | SENSE:MIXer:CALCulate | Calculate |

### Mixer Power

<p>| Power On (All Channels) | OPut[:STATe] | SourcePowerState |
| LO1 Power | SENSE:MIXer:LO:POWer | LOPower |
| LO2 Power | SENSE:MIXer:LO:POWer | LOPower |
| LO1 Swept Power |  |  |</p>
<table>
<thead>
<tr>
<th>Start</th>
<th>SENSe:MIXer:LO:FREQuency:STARt</th>
<th>LOStartFrequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stop</td>
<td>SENSe:MIXer:LO:FREQuency:STOP</td>
<td>LOStopFrequency</td>
</tr>
<tr>
<td>Step</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**LO2 Swept Power**

<table>
<thead>
<tr>
<th>Start</th>
<th>SENSe:MIXer:LO:FREQuency:STARt</th>
<th>LOStartFrequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stop</td>
<td>SENSe:MIXer:LO:FREQuency:STOP</td>
<td>LOStopFrequency</td>
</tr>
<tr>
<td>Step</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Path Configuration...**

| SENSe:PATH:CONFig:ELEMent[:STATe] | Element |

**Mixer Setup**

<table>
<thead>
<tr>
<th>Converter Stages</th>
<th>SENSe:MIXer:STAGe</th>
<th>LOStage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add Source...</td>
<td>SENSe:ROLE:DEVice</td>
<td>AssignSourceToRole</td>
</tr>
<tr>
<td>Path Configuration...</td>
<td>SENSe:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td>Save...</td>
<td>SENSe:MIXer:SAVE</td>
<td>SaveFile</td>
</tr>
<tr>
<td>Load...</td>
<td>SENSE:MIXer:LOAD</td>
<td>LoadFile</td>
</tr>
<tr>
<td>---------</td>
<td>------------------</td>
<td>----------</td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Quick Start...</td>
<td>S-Param</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Balanced</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BalSMeasurement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BBalMeasurement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SBalMeasurement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SSBMeasurement</td>
</tr>
<tr>
<td>Other</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
</tr>
</tbody>
</table>

**Layout Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Trace</td>
<td></td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>New Channel</td>
<td></td>
<td>None</td>
<td>chans.Add</td>
</tr>
<tr>
<td>New Window</td>
<td></td>
<td>DISPlay:WINDow[:STATe]</td>
<td>Add</td>
</tr>
<tr>
<td>New Sheet</td>
<td></td>
<td>DISPlay:SHEet:STATe</td>
<td>None</td>
</tr>
<tr>
<td>Action</td>
<td>Parameter</td>
<td>Command</td>
<td>Option</td>
</tr>
<tr>
<td>----------</td>
<td>-----------</td>
<td>--------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>Delete</td>
<td>TrN</td>
<td>DISPlay.WINDow:TRACe:DELeTe</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>ChN</td>
<td>SYSTem:CHANnels:DELeTe</td>
<td>RemoveChannelNumber</td>
</tr>
<tr>
<td></td>
<td>WinN</td>
<td>DISPlay.WINDow[:STATe]</td>
<td>Add</td>
</tr>
<tr>
<td>Select</td>
<td>TrN</td>
<td>DISPlay.WINDow:TRACe:SELect</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>ChN</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>WinN</td>
<td>DISPlay.WINDow:TRACe:SELect</td>
<td>None</td>
</tr>
<tr>
<td>Measure...</td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
</tr>
<tr>
<td>Meas Class...</td>
<td></td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
</tbody>
</table>
## CF_Setup Commands - Standard

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td>Sweep Setup...</td>
</tr>
<tr>
<td>Linear Frequency</td>
</tr>
<tr>
<td>Log Frequency</td>
</tr>
<tr>
<td>Power Sweep</td>
</tr>
<tr>
<td>CW Time</td>
</tr>
<tr>
<td>Segment Sweep</td>
</tr>
<tr>
<td>Phase Sweep</td>
</tr>
<tr>
<td><strong>Sweep Properties</strong></td>
</tr>
<tr>
<td>Start</td>
</tr>
<tr>
<td>Stop</td>
</tr>
<tr>
<td>Power</td>
</tr>
</tbody>
</table>

7678
<table>
<thead>
<tr>
<th>Points</th>
<th>SENSE:SWEep:POINts</th>
<th>NumberOfPoints</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF Bandwidth</td>
<td>SENSE:BWIDth[:RESolution]</td>
<td>IFBandwidth</td>
</tr>
<tr>
<td>Start Power</td>
<td>SOURce:POWer:STARt</td>
<td>chan.StartPower</td>
</tr>
<tr>
<td>Stop Power</td>
<td>SOURce:POWer:STOP</td>
<td>chan.StopPower</td>
</tr>
<tr>
<td>CW Freq</td>
<td>SENSE:FREQuency[:CW]</td>
<td>CWFrequency</td>
</tr>
<tr>
<td>Segment Table</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X-axis Point Spacing</td>
<td>SENSE:SEGMent:X:SPACing</td>
<td>chan.XAxisPointSpacing</td>
</tr>
<tr>
<td>Allow arbitrary Segments</td>
<td>SENSE:SEGMent:ARBitrary</td>
<td>segs.AllowArbitrarySegments</td>
</tr>
<tr>
<td>Display Center/Span Freq</td>
<td>SENSE:SEGMent:FREQuency:CENTer</td>
<td>chan.centerFrequency</td>
</tr>
<tr>
<td></td>
<td>SENSE:SEGMent:FREQuency:SPAN</td>
<td>chan.FrequencySpan</td>
</tr>
<tr>
<td>Timing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sweep Time</td>
<td>SENSE:SWEep:TIME</td>
<td>SweepTime</td>
</tr>
<tr>
<td>Dwell Time</td>
<td>SENSE:SWEep:DWELI</td>
<td>DwellTime</td>
</tr>
<tr>
<td>Feature</td>
<td>SCPI Command</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Sweep Delay</td>
<td>SENSe:SWEep:DWELI:SDELay</td>
<td>SweepDelay</td>
</tr>
<tr>
<td>Auto Sweep Time</td>
<td>SENSe:SWEep:TIME:AUTO</td>
<td>SweepTime</td>
</tr>
<tr>
<td>Fast Sweep - Reduce settling time</td>
<td>SENSe:SWEep:SPEed</td>
<td>SweepSpeedMode</td>
</tr>
<tr>
<td>Sweep Mode</td>
<td>SENSe:SWEep:GENeration</td>
<td>SweepGenerationMode</td>
</tr>
<tr>
<td>Sweep Sequence</td>
<td>SENSe:SWEep:GENeration:POINtsweep</td>
<td>PointSweepState</td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Quick Start...</td>
<td>S-Param</td>
<td>CreateSParameterEx</td>
</tr>
<tr>
<td>Balanced</td>
<td>CALCulate:MEASure:PARameter</td>
<td>BalSMeasurement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BBalMeasurement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SBalMeasurement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SSBMeasurement</td>
</tr>
<tr>
<td>Other</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
</tr>
</tbody>
</table>

**Layout Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Action</td>
<td>Command</td>
<td>Value</td>
<td>Result</td>
</tr>
<tr>
<td>------------</td>
<td>--------------------------------</td>
<td>---------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>New Trace</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
<td></td>
</tr>
<tr>
<td>New Channel</td>
<td>None</td>
<td>chans.Add</td>
<td></td>
</tr>
<tr>
<td>New Window</td>
<td>DISPlay:WINDow[:STATe]</td>
<td>Add</td>
<td></td>
</tr>
<tr>
<td>New Sheet</td>
<td>DISPlay:SHEet:STATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Delete</td>
<td>TrN DISPlay:WINDow:TRACe:DELeTe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ChN SYSTem:CHANnels:DELeTe</td>
<td>RemoveChannelNumber</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WinN DISPlay:WINDow[:STATe]</td>
<td>Add</td>
<td></td>
</tr>
<tr>
<td>Select</td>
<td>TrN DISPlay:WINDow:TRACe:SELect</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ChN None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WinN DISPlay:WINDow:TRACe:SELect</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Measure...</td>
<td>S-Parameter CALCulate:MEASure:PARameter</td>
<td>CreateSPParameterEx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Balanced CALCulate:MEASure:PARameter</td>
<td>BaisMeasurement</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>BBalMeasurement</td>
<td></td>
</tr>
<tr>
<td>Receptors</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------</td>
<td>-------------------</td>
<td></td>
</tr>
<tr>
<td>Meas Class...</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
<td></td>
</tr>
</tbody>
</table>

**Internal Hardware Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF Path Config...</td>
<td></td>
<td>SENSE:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td>IF Path Config...</td>
<td></td>
<td>SENSE:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td>Mechanic Devices...</td>
<td>Trigger...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interface Control...</td>
<td>Enable Interface Control</td>
<td>CONTrol:CHANnel:INTERface:CONTrol[:STATe]</td>
<td>IC.State</td>
</tr>
<tr>
<td>Channel</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>--------------</td>
<td>------</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>Channel Control Label</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Before Sweep Start</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Enable Control</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Handler I/O Control</td>
<td>CONTrol:HANDler[:DATA]</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Test Set I/O Control (addr. data)</td>
<td>CONTrol:EXTernal:TESTset:DATa</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Dwell After Command</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Aux I/O Output Voltage</td>
<td>CONTrol:AUXiliary:OUTPut:VOLTage</td>
<td>put_OutputVoltage</td>
<td>None</td>
</tr>
<tr>
<td>After Sweep End</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>------------------------------------------</td>
<td>---------------------------</td>
<td></td>
</tr>
<tr>
<td>Handler I/O Control</td>
<td><code>CONTrol:HANDler[:DATA]</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Test Set I/O Control (addr. data)</td>
<td><code>CONTrol:EXTernal:TESTset:DATa</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Dwell After Command</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Aux I/O Output Voltage</td>
<td><code>CONTrol:AUXiliary:OUTPut:VOLTage</code></td>
<td><code>put_OutputVoltage</code></td>
<td></td>
</tr>
<tr>
<td>Reset All</td>
<td><code>CALCulate:MEASure:DELete:ALL</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Save</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Recall</td>
<td><code>CONTrol:CHANnel:INTERface:CONTrol:CONFig:RECall[:STATE]</code></td>
<td><code>IC.ConfigurationFile</code></td>
<td></td>
</tr>
<tr>
<td>Reference ...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Auto-select Reference</td>
<td><code>SENSe:ROSCillator:CONT:AUTO</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Reference In</td>
<td><code>SENSe:ROSCillator:EXTernal:FREQuency</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Internal/External</td>
<td>Internal/External</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Reference Out</td>
<td>Reference Out</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>LF Extension</td>
<td>LF Extension</td>
<td>LowFrequencyExtension</td>
<td></td>
</tr>
</tbody>
</table>

### External Hardware Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>External Device...</td>
<td>New</td>
<td>SYSTem:CONFigure:EDEVice:ADD</td>
<td>Add (External Device)</td>
</tr>
<tr>
<td></td>
<td>Remove</td>
<td>SYSTem:CONFigure:EDEVice:REMove</td>
<td>Remove</td>
</tr>
<tr>
<td></td>
<td>Name</td>
<td>SYSTem:CONFigure:EDEVice:ADD</td>
<td>Add (External Device)</td>
</tr>
<tr>
<td></td>
<td>Device Type</td>
<td>SYSTem:CONFigure:EDEVice:DTYPe</td>
<td>DeviceType</td>
</tr>
<tr>
<td></td>
<td>Driver</td>
<td>SYSTem:CONFigure:EDEVice:DRIVer</td>
<td>Driver</td>
</tr>
<tr>
<td></td>
<td>Active - Show in UI</td>
<td>SYSTem:CONFigure:EDEVice:STATe</td>
<td>Active</td>
</tr>
</tbody>
</table>
### Device Properties...

#### DC Meter

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Timeout</strong></td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td><strong>Dwell Before Sweep</strong></td>
<td>System:CONFigure:EDEVice:DC:DSWeep</td>
<td>SweepDwell</td>
</tr>
<tr>
<td><strong>Dwell After Point Set</strong></td>
<td>System:CONFigure:EDEVice:DC:DPOint</td>
<td>PointDwell</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>System:CONFigure:EDEVice:DC:TYPE</td>
<td>DCTYPE</td>
</tr>
<tr>
<td><strong>Receiver Correction On</strong></td>
<td>System:CONFigure:EDEVice:DC:CORRection</td>
<td>DCCorrection</td>
</tr>
<tr>
<td><strong>Offset</strong></td>
<td>System:CONFigure:EDEVice:DC:OFFSet</td>
<td>DCOffset</td>
</tr>
<tr>
<td><strong>Scaling</strong></td>
<td>System:CONFigure:EDEVice:DC:SCALe</td>
<td>DCScale</td>
</tr>
<tr>
<td><strong>Edit Commands...</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ID Query</strong></td>
<td>System:CONFigure:EDEVice:DC:QUERy:ID</td>
<td>IDQuery</td>
</tr>
<tr>
<td>Error Query</td>
<td>SYSTEM:CONFigure:EDEvie:DC:QUery:ERRor</td>
<td>ErrorQuery</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Enable I/O</td>
<td>SYSTEM:CONFigure:EDEvie:DC:COMMand:INIT</td>
<td>InitCmd</td>
</tr>
<tr>
<td>Disable I/O</td>
<td>SYSTEM:CONFigure:EDEvie:DC:COMMand:EXIT</td>
<td>ExitCmd</td>
</tr>
<tr>
<td>Before Sweep</td>
<td>SYSTEM:CONFigure:EDEvie:DC:COMMand:SWeep:BEFor e</td>
<td>BeforeSweepCmd</td>
</tr>
<tr>
<td>After Sweep</td>
<td>SYSTEM:CONFigure:EDEvie:DC:COMMand:SWeep:AFTer</td>
<td>AfterSweepCmd</td>
</tr>
<tr>
<td>Abort Sweep</td>
<td>SYSTEM:CONFigure:EDEvie:DC:COMMand:SWeep:ABOR t</td>
<td>AbortSweepCmd</td>
</tr>
<tr>
<td>Point Set Commands</td>
<td>SYSTEM:CONFigure:EDEvie:DC:COMMand:POINt:SET</td>
<td>PointCmd</td>
</tr>
<tr>
<td>DC Source</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Timeout</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Dwell Before Sweep</td>
<td>SYSTEM:CONFigure:EDEvie:DC:DSWEEP</td>
<td>SweepDwell</td>
</tr>
<tr>
<td>Dwell After Point Set</td>
<td>SYSTEM:CONFigure:EDEvie:DC:DPOint</td>
<td>PointDwell</td>
</tr>
<tr>
<td>Type</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>SYSTem:CONFigure:EDEVice:DC:TYPE</td>
<td>DCTYPE</td>
<td></td>
</tr>
<tr>
<td>SYSTem:CONFigure:EDEVice:DC:LIMit:CURRent</td>
<td>CurrentLimit</td>
<td></td>
</tr>
<tr>
<td>SYSTem:CONFigure:EDEVice:DC:LIMit:VOLTage</td>
<td>VoltageLimit</td>
<td></td>
</tr>
<tr>
<td>Edit Commands...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYSTem:CONFigure:EDEVice:DC:QUERy:ID</td>
<td>IDQuery</td>
<td></td>
</tr>
<tr>
<td>SYSTem:CONFigure:EDEVice:DC:QUERy:ERRor</td>
<td>ErrorQuery</td>
<td></td>
</tr>
<tr>
<td>SYSTem:CONFigure:EDEVice:DC:COMMand:INIT</td>
<td>InitCmd</td>
<td></td>
</tr>
<tr>
<td>SYSTem:CONFigure:EDEVice:DC:COMMand:EXIT</td>
<td>ExitCmd</td>
<td></td>
</tr>
<tr>
<td>SYSTem:CONFigure:EDEVice:DC:MAX:VALue</td>
<td>MaxOutput</td>
<td></td>
</tr>
<tr>
<td>SYSTem:CONFigure:EDEVice:DC:MAX[:STATE]</td>
<td>MaxOutputState</td>
<td></td>
</tr>
<tr>
<td>SYSTem:CONFigure:EDEVice:DC:MAX:VALue</td>
<td>MaxOutput</td>
<td></td>
</tr>
<tr>
<td>SYSTem:CONFigure:EDEVice:DC:MIN:VALue</td>
<td>MinOutput</td>
<td></td>
</tr>
<tr>
<td>Define Min As</td>
<td>SYSTEM:CONFigure:EDEVice:DC:MIN[:STATE]</td>
<td>MinOutputState</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td>SYSTEM:CONFigure:EDEVice:DC:MIN:VALue</td>
<td>MinOutput</td>
</tr>
<tr>
<td>Before Sweep</td>
<td>SYSTEM:CONFigure:EDEVice:DC:COMMAND:SWEep:BEFor e</td>
<td>BeforeSweepCmd</td>
</tr>
<tr>
<td>After Sweep</td>
<td>SYSTEM:CONFigure:EDEVice:DC:COMMAND:SWEep:AFTer</td>
<td>AfterSweepCmd</td>
</tr>
<tr>
<td>Abort Sweep</td>
<td>SYSTEM:CONFigure:EDEVice:DC:COMMAND:SWEep:ABOR t</td>
<td>AbortSweepCmd</td>
</tr>
<tr>
<td>Point Set Commands</td>
<td>SYSTEM:CONFigure:EDEVice:DC:COMMAND:POINt:SET</td>
<td>PointCmd</td>
</tr>
<tr>
<td>Power Meter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sensor A/B</td>
<td>None</td>
<td>PowerMeterChannel</td>
</tr>
<tr>
<td>Tolerance</td>
<td>SOURce:POWer:CORRection:COLLect:AVERage:NTOLeran ce</td>
<td>ReadingsTolerance</td>
</tr>
<tr>
<td>Max Number of Readings</td>
<td>SOURce:POWer:CORRection:COLLect:AVERage:COUNt</td>
<td>ReadingsPerPoint</td>
</tr>
<tr>
<td><strong>Edit Table</strong></td>
<td>SOURce:POWer:CORRection:COLLect:TABLe:POINts?</td>
<td>Count</td>
</tr>
<tr>
<td></td>
<td>SOURce:POWer:CORRection:COLLect:TABLe:FREQuency</td>
<td>Frequency</td>
</tr>
<tr>
<td></td>
<td>SOURce:POWer:CORRection:COLLect:TABLe:DATA</td>
<td>Loss</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SegmentNumber</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Add</td>
</tr>
</tbody>
</table>

| **Uncertainty** | SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:CATalog? | UncertaintyModelCatalog |
| | SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:FILE | UncertaintyFile |
| | SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:MODEl | UncertaintyModel |
| | SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:PLEVel? | PowerForBestAccuracy |
| | SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:READ? | PowerMtrReadingUncertainty |

| **Pulse Generator** | |
| | |

| **Time out** | SYSTem:CONFigure:EDEVice:TOUT | TimeOut |
| | |

| **Primary Mode** | SYSTem:CONFigure:EDEVice:PULSe:PMODe | PrimaryMode |
| | |

| **Output** | SYSTem:CONFigure:EDEVice:PULSe:CHAN | OutputChannel |
| | |

| **High** | SYSTem:CONFigure:EDEVice:PULSe:HAMP | HighAmplitude |
| | |

<p>| <strong>Low</strong> | SYSTem:CONFigure:EDEVice:PULSe:LAMP | LowAmplitude |</p>
<table>
<thead>
<tr>
<th>Source Imp</th>
<th>SYST:CON Figure:EDEV:PU LSe:SIMP</th>
<th>SourceImpedance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load Imp</td>
<td>SYST:CON Figure:EDEV:PU LSe:LIMP</td>
<td>LoadImpedance</td>
</tr>
<tr>
<td>SMU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Setup</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chan N</td>
<td>SYST:CON Figure:EDEV:ADD</td>
<td>Add (External Device)</td>
</tr>
<tr>
<td>Trigger Mode</td>
<td>SYST:CON Figure:EDEV:SOURce:TMODe</td>
<td>TriggerMode</td>
</tr>
<tr>
<td>SMU Trigger In</td>
<td></td>
<td>TriggerInPin</td>
</tr>
<tr>
<td>SMU Trigger Out</td>
<td></td>
<td>TriggerPort</td>
</tr>
<tr>
<td>Source/Voltage Meter/Current Meter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMU Chan</td>
<td>SYST:CON Figure:EDEV:IOENable</td>
<td>IOEnable</td>
</tr>
<tr>
<td></td>
<td>SYST:CON Figure:EDEV:SMU:CHANnel[1-4]:STATe</td>
<td>ChanActive</td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Source type</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:TYPE</code></td>
<td>DCType</td>
</tr>
<tr>
<td>Timeout</td>
<td><code>SYSTem:CONFigure:EDEVice:TOUT</code></td>
<td>TimeOut</td>
</tr>
<tr>
<td>Dwell Before Sweep</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:DSWeep</code></td>
<td>SweepDwell</td>
</tr>
<tr>
<td>Dwell After Point Set</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:DPOint</code></td>
<td>PointDwell</td>
</tr>
<tr>
<td>Current Limit</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:LIMit:CURREnt</code></td>
<td>CurrentLimit</td>
</tr>
<tr>
<td>Voltage Limit</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:LIMit:VOLTage</code></td>
<td>VoltageLimit</td>
</tr>
<tr>
<td>Source Correction On</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:CORRection</code></td>
<td>DCCorrection</td>
</tr>
<tr>
<td>Offset</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:OFFSet</code></td>
<td>DCOffset</td>
</tr>
<tr>
<td>Scaling</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:SCALE</code></td>
<td>DCScale</td>
</tr>
<tr>
<td>Receiver Correction On</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:CORRection</code></td>
<td>DCCorrection</td>
</tr>
<tr>
<td>Offset</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:OFFSet</code></td>
<td>DCOffset</td>
</tr>
<tr>
<td><strong>Scaling</strong></td>
<td><strong>Edit Commands - DC Source</strong></td>
<td><strong>Edit Commands - DC Meter</strong></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Timeout</td>
<td><strong>SYSTem:CONFigure:EDEVice:TOUT</strong></td>
<td><strong>TimeOut</strong></td>
</tr>
<tr>
<td>Dwell Per Point</td>
<td><strong>SYSTem:CONFigure:EDEVice:SOURce:DPP</strong></td>
<td><strong>DwellPerPoint</strong></td>
</tr>
<tr>
<td>Enable Modulation Control</td>
<td><strong>SYSTem:CONFigure:EDEVice:SOURce:MODulation:CONTro[l[:STATe]]</strong></td>
<td><strong>EnableModulationControl</strong></td>
</tr>
<tr>
<td>Trigger Mode</td>
<td><strong>SYSTem:CONFigure:EDEVice:SOURce:TMODe</strong></td>
<td><strong>TriggerMode</strong></td>
</tr>
<tr>
<td>Trigger Port</td>
<td><strong>SYSTem:CONFigure:EDEVice:SOURce:TPORt</strong></td>
<td><strong>TriggerPort</strong></td>
</tr>
<tr>
<td>Edit Commands</td>
<td>Operation complete (*OPC)</td>
<td>Preset</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------</td>
<td>--------</td>
</tr>
<tr>
<td></td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power Meter Setup...</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SYSTem:CONFigure:EDEVice:PMAR:SENSor</td>
<td></td>
</tr>
<tr>
<td>Tolerance</td>
<td>SYSTEM:CONFigure:EDEVice:PMAR:READing:NTOLerance</td>
<td>ReadingsTolerance</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Max Number of Readings</td>
<td>SYSTEM:CONFigure:EDEVice:PMAR:READing:COUNt</td>
<td>ReadingsPerPoint</td>
</tr>
<tr>
<td>Use Loss Table</td>
<td>SYSTEM:CONFigure:EDEVice:PMAR:TABLE:LOSS:DATA</td>
<td>Loss</td>
</tr>
<tr>
<td>Edit Table</td>
<td>SYSTEM:CONFigure:EDEVice:PMAR:TABLE:LOSS:DATA</td>
<td>Loss</td>
</tr>
<tr>
<td>External Testset...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Select ID</td>
<td>SENSE:MULTiplexer:OUTPut[:DATa]</td>
<td>ControlLines</td>
</tr>
<tr>
<td>Test Set</td>
<td>SENSE:MULTiplexer:TYPe</td>
<td>ts.Add</td>
</tr>
<tr>
<td></td>
<td>SENSE:MULTiplexer:ADDRes</td>
<td></td>
</tr>
<tr>
<td>Enable Test Set Control</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Show Test Set Properties</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Port Control</td>
<td>SENSE:MULTiplexer:PORT:SEl ect</td>
<td>ts.SelectPort</td>
</tr>
<tr>
<td>Control Lines</td>
<td>SENSE:MULTiplexer:OUTPut[:DATa]</td>
<td>ControlLines</td>
</tr>
<tr>
<td>Line X</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>DC Source</td>
<td>SCPI Commands</td>
<td>COM Commands</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------</td>
<td>--------------</td>
</tr>
<tr>
<td>(E5092A only)</td>
<td>CONTrol:MULTiplexer Commands</td>
<td>None</td>
</tr>
</tbody>
</table>

**Multiport**

<table>
<thead>
<tr>
<th>Testset</th>
<th>SENSE:MULTiplexer:TYPE</th>
<th>ts.Add</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address/ID</td>
<td>SENSE:MULTiplexer:ADDRESS</td>
<td>None</td>
</tr>
</tbody>
</table>

**Restart as a standalone PNA**

<table>
<thead>
<tr>
<th>SYSTem:CONFigure</th>
<th>app.Configure</th>
</tr>
</thead>
</table>

**Restart as a multiport PNA with this testset**

<table>
<thead>
<tr>
<th>SYSTem:CONFigure</th>
<th>app.Configure</th>
</tr>
</thead>
</table>

**port count =**

<table>
<thead>
<tr>
<th>SENSE:MULTiplexer:COUNT</th>
<th>None</th>
</tr>
</thead>
</table>

**Millimeter Config...**

(See Supported Application s)

<table>
<thead>
<tr>
<th>Select Configuration</th>
<th>SYSTem:CONFigure:MWAVE:CONFigure:ACTive</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SYSTem:CONFigure:MWAVE:CONFigure:CATalog?</td>
<td>None</td>
</tr>
</tbody>
</table>

**New**

<table>
<thead>
<tr>
<th>SYSTem:CONFigure:MWAVE:CONFigure:ADD</th>
<th>None</th>
</tr>
</thead>
</table>

**Remove**

<table>
<thead>
<tr>
<th>SYSTem:CONFigure:MWAVE:CONFigure:REMove</th>
<th>None</th>
</tr>
</thead>
</table>

**Multiplier RF IN**

<table>
<thead>
<tr>
<th>ΣΨΣΤΕμ:ΧΟΝΦιγυρε:ΜΩΑζε:ΦΡΕΘονενχψ:ΡΦ:ΜΥΑΤιπλι</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting</td>
<td>M-code/M-function</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Multiplier LO IN</td>
<td>SYSTem:CONFigure:MWAVe:FREQuency:LO:MULTiplier</td>
</tr>
<tr>
<td>Test Port Frequency</td>
<td>SYSTem:CONFigure:MWAVe:FREQuency:STARt, SYSTem:CONFigure:MWAVe:FREQuency:STOP</td>
</tr>
<tr>
<td>Name</td>
<td>SYSTem:CONFigure:MWAVe:CONFigure:ACTive</td>
</tr>
<tr>
<td>Test Set</td>
<td>SYSTem:CONFigure:MWAVe:TSET:NAME, SYSTem:CONFigure:MWAVe:TSET:CATalog?</td>
</tr>
<tr>
<td>Mixer Mode</td>
<td>ΨΣΤεμ:ΧΟΝΦιγυρε:ΜΩΛάζε:ΤΣΕΤ:ΜΙΞερ</td>
</tr>
<tr>
<td>Module IF Gain</td>
<td></td>
</tr>
<tr>
<td>Test Set IF Switch settings for Channel N</td>
<td>SYSTem:CONFigure:MWAVe:TSET:RPANel</td>
</tr>
<tr>
<td>Enable Test Set RF ALC</td>
<td>ΨΣΤεμ:ΧΟΝΦιγυρε:ΜΩΛάζε:ΤΣΕΤ:ΑΛΧ</td>
</tr>
<tr>
<td>Max Power limit at Module RF IN</td>
<td>ΨΣΤεμ:ΧΟΝΦιγυρε:ΜΩΛάζε:ΤΣΕΤ:ΠΟΩερ:ΑΙΜιτ</td>
</tr>
<tr>
<td>Offset</td>
<td>ΨΣΤεμ:ΧΟΝΦιγυρε:ΜΩΛάζε:ΤΣΕΤ:ΠΟΩερ:ΟΦΦΣετ</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Slope</td>
<td><code>SYSTem:CONFigure:MWAVe:CONF:ACTive:MODel?</code></td>
</tr>
<tr>
<td>PNA RF Source...</td>
<td><code>SYSTem:CONFigure:MWAVe:CONF:ACTive:OPTion?</code></td>
</tr>
<tr>
<td>RF Stop</td>
<td><code>SYSTem:CONFigure:MWAVe:CONF:ACTive:PORT{1:4}:MODel?</code></td>
</tr>
<tr>
<td>PNA LO Source...</td>
<td><code>SYSTem:CONFigure:MWAVe:CONF:ACTive:PORT{1:4}:MODel?</code></td>
</tr>
</tbody>
</table>

7698
<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return the option number of the frequency extender module connected to the specified port number.</td>
<td><code>SYSTem:CONFigure:MWAVE:CONF:ACTive:PORT{1:4}:OPTion?</code></td>
<td>None</td>
</tr>
<tr>
<td>Read serial number of frequency extender module connected to specified port.</td>
<td><code>ΣΨΣΤεμ:ΧΟΝΦιγυρε:ΜΩΑζε:ΧΟΝΦ:ΑΧΤίσε:ΠΟΡΤ{1:4}:ΣΕΡταλ?</code></td>
<td>None</td>
</tr>
<tr>
<td>Read serial number of active test set.</td>
<td><code>ΣΨΣΤεμ:ΧΟΝΦιγυρε:ΜΩΑζε:ΧΟΝΦ:ΑΧΤίσε:ΣΕΡταλ?</code></td>
<td>None</td>
</tr>
</tbody>
</table>
CF_Setup Commands - TDR

Setup Commands - TDR Measurement Class

Click here to view links to Setup commands for all Measurement Classes.

SYST:TDR:PRESet launches TDR application without VNA-TDR GUI.

Tool Bar
Setup Pane
TDR/TDT Pane
Eye/Mask Pane

---

### Tool Bar Commands

![Tool Bar Commands Image]

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>Description</th>
<th>SCPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trace</td>
<td></td>
<td>Select active trace</td>
<td>CALC:PAR:MNUM:SELect</td>
</tr>
<tr>
<td>Auto Scale</td>
<td>Scale menu</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>X</td>
<td>Do horizontal auto scale</td>
<td>DISP:TDR:MEAS:X:SCAL:AUTO</td>
</tr>
<tr>
<td></td>
<td>Y</td>
<td>Do vertical auto scale</td>
<td>DISP:MEAS:Y:SCAL:AUTO</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------</td>
<td>--------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>All Traces</td>
<td>Do horizontal and vertical auto scale for all traces</td>
<td>DISP:TDR:MEAS:X:SCAL:AUTO DISP:MEAS:Y:SCAL:AUTO</td>
<td></td>
</tr>
<tr>
<td>Run</td>
<td>Trigger continuous</td>
<td>SENS:TDR:SWE:MODE RUN</td>
<td></td>
</tr>
<tr>
<td>Stop / Single</td>
<td>Trigger hold (when continuous)</td>
<td>SENS:TDR:SWE:MODE HOLD</td>
<td></td>
</tr>
<tr>
<td>Data / Mem</td>
<td>Data and Memory menu</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data -&gt; Mem</td>
<td>Save data trace to memory</td>
<td>CALC:MEAS:MATH:MEMorize</td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td>Display off</td>
<td>DISP:TDR:MEAS:DMEM:TYPE OFF</td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td>Display data trace only</td>
<td>DISP:TDR:MEAS:DMEM:TYPE DATA</td>
<td></td>
</tr>
<tr>
<td>Memory</td>
<td>Display memory trace only</td>
<td>DISP:TDR:MEAS:DMEM:TYPE MEMory</td>
<td></td>
</tr>
<tr>
<td>Data &amp; Memory</td>
<td>Display data and memory traces</td>
<td>DISP:TDR:MEAS:DMEM:TYPE DMEMory</td>
<td></td>
</tr>
<tr>
<td>Marker</td>
<td>Select active marker</td>
<td>CALC:TDR:MEAS:MARK:STAT</td>
<td></td>
</tr>
<tr>
<td>Marker Search</td>
<td>Marker search menu</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td>Marker search Off</td>
<td>CALC:MEAS:MARK:FUNC:TRACk OFF</td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>Set max marker search On</td>
<td>CALC:MEAS:MARK:FUNC:SELect MAX; TRACk ON</td>
<td></td>
</tr>
<tr>
<td>Min</td>
<td>Set min marker search On</td>
<td>CALC:MEAS:MARK:FUNC:SELect MIN; TRACk ON</td>
<td></td>
</tr>
<tr>
<td>Rise Time (10%-90%)</td>
<td>Select time threshold to 10-90%</td>
<td>CALC:TDR:MEAS:TTIM:THReshold T1_9</td>
<td></td>
</tr>
<tr>
<td>Rise Time (20%-80%)</td>
<td>Select time threshold to 20-80%</td>
<td>CALC:TDR:MEAS:TTIM:THReshold T2_8</td>
<td></td>
</tr>
<tr>
<td>ΔTime</td>
<td>Open ΔTime dialog</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>File</th>
<th>File menu</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Save State</td>
<td>Save TDR state file</td>
<td>MMEM:TDR:STOR:STATe</td>
</tr>
<tr>
<td>Recall State</td>
<td>Recall TDR state file</td>
<td>MMEM:TDR:LOAD:STATe</td>
</tr>
<tr>
<td>Save Active Trace Data</td>
<td>Save the active trace data to csv file</td>
<td>MMEM:TDR:STOR:FDATa</td>
</tr>
<tr>
<td>Save All Trace Data</td>
<td>Save all trace data to csv file</td>
<td>MMEM:TDR:STOR:FDAT:ALL</td>
</tr>
<tr>
<td>Softkey</td>
<td>Description</td>
<td>SCPI</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Setup Wizard</td>
<td>Launch Setup Wizard</td>
<td>SYST:TDR:PRESet</td>
</tr>
<tr>
<td>Preset</td>
<td>Preset TDR state</td>
<td>SYST:TDR:PRESet</td>
</tr>
</tbody>
</table>

**Save Touchstone**

Save trace data to touchstone file

**MMEM:TDR:STOR:SNP**

**Save Image**

Save display image to file

**HCOP:FILE**

**Invert Color**

Set invert color state

**DISP:TDR:IMAGe**

**Minimize (Up/Down Arrow)**

Minimize TDR GUI

**DISP:TDR:MINimize**

**Help (?)**

Open TDR help

None

**Close (X)**

Exit TDR application

**:SYST:PRESet**

**Setup Pane Commands**

![TDR GUI](image)

**Softkey**

**Sub-item**

**Description**

**SCPI**
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>Description</th>
<th>SCPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ref. Z</td>
<td></td>
<td>Set reference impedance value</td>
<td>SENS:CORR:TDR:RIMPedance</td>
</tr>
</tbody>
</table>

**DUT Topology**

Select DUT topology

**CALC:TDR:DEVice**

**Stim. Ampl.**

Set stimulus amplitude level

**CALC:TDR:TIME:STEP:AMPLitude**

**DUT Length (Auto)**

Automatically estimate DUT length

**SENS:TDR:DLEN:AUTO:IMMediate**

**DUT Length (Numeric Edit)**

Set DUT length

**SENS:TDR:DLEN:DATA**

**Deskew**

Open Deskew dialog

None

**Deskew&Loss**

Open Deskew & Loss Compensation dialog

None

**ECal**

Open Full Calibration (ECal) dialog

None

**TDR**

![TDR Interface](image)

**Setup**

**TDR/TDT**

**Eye/Mask**

**Softkey**

**Sub-item**

**Description**

**SCPI**

```for loop
```
<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dielectric Const.</td>
<td>SENS:CORR:TDR:DCONstant</td>
</tr>
<tr>
<td>Set dielectric constant</td>
<td></td>
</tr>
<tr>
<td>value</td>
<td></td>
</tr>
<tr>
<td>Velocity Factor</td>
<td>SENS:CORR:TDR:DCONstant</td>
</tr>
<tr>
<td>Set velocity factor (=</td>
<td></td>
</tr>
<tr>
<td>1/sqrt(dielectric const))</td>
<td></td>
</tr>
<tr>
<td>Source Power</td>
<td>SOUR:TDR:POWER:LEV:IMM:AMPLitude</td>
</tr>
<tr>
<td>Set source power level</td>
<td></td>
</tr>
<tr>
<td>Bal Port Config</td>
<td>None</td>
</tr>
<tr>
<td>Open Balanced Source</td>
<td></td>
</tr>
<tr>
<td>setting dialog</td>
<td></td>
</tr>
<tr>
<td>Freq Limits Config</td>
<td>None</td>
</tr>
<tr>
<td>Open Frequency Limits</td>
<td></td>
</tr>
<tr>
<td>dialog to set start and</td>
<td></td>
</tr>
<tr>
<td>stop frequency limits</td>
<td></td>
</tr>
<tr>
<td>Advanced Mode</td>
<td>None</td>
</tr>
<tr>
<td>Move to advanced mode</td>
<td></td>
</tr>
</tbody>
</table>

![TDR diagram]

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>Description</th>
<th>SCPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Averaging (Check Box)</td>
<td></td>
<td>Set averaging</td>
<td>SENS:AVER:STATe</td>
</tr>
<tr>
<td></td>
<td></td>
<td>state</td>
<td></td>
</tr>
</tbody>
</table>

7705
<table>
<thead>
<tr>
<th>Averaging (Spin Control)</th>
<th>Set averaging count</th>
<th>SENS:AVER:COUNt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Trigger</td>
<td>Set average trigger state</td>
<td>SENS:TDR:SWE:AVERage</td>
</tr>
<tr>
<td>IF Bandwidth</td>
<td>Set resolution bandwidth</td>
<td>SENS:TDR:BWIDth:RESolution</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TDR</th>
<th>Setup</th>
<th>TDR/TDT</th>
<th>Eye/Mask</th>
<th>More Functions</th>
<th>Average</th>
<th>Adv Waveform</th>
<th>Hot TDR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Basic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Emphasis</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>De-embedding</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Equalization</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>Description</th>
<th>SCPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emphasis</td>
<td></td>
<td>Open Advanced Waveform Emphasis dialog</td>
<td>None</td>
</tr>
<tr>
<td>De-embedding</td>
<td></td>
<td>Open Advanced Waveform De-embedding dialog</td>
<td>Enable de-embedding CALC:TDR:DEEM:STATe</td>
</tr>
</tbody>
</table>

Enable the SNP file de-embedding for the selected port.
CALC:TDR:DEEM:PORT:STAT (for single end)
CALC:TDR:DEEM:BPOR:STAT (for balance)

Specify the SNP file de-embedding for the selected port.
CALC:TDR:DEEM:PORT:FILename (for single end)
CALC:TDR:DEEM:BPOR:FILename (for balance)
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>Description</th>
<th>SCPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Rate</td>
<td></td>
<td>Set spurious data rate</td>
<td>SENS:TDR:SPUR:INP:DRATE</td>
</tr>
<tr>
<td>Avoid Spurious</td>
<td></td>
<td>Execute avoid spurious</td>
<td>SENS:TDR:SPURious:AVO:IMMEDIATE</td>
</tr>
<tr>
<td>Avoid Spurious</td>
<td>(Check Mark)</td>
<td>Show avoid spurious state</td>
<td>SENS:TDR:SPURious:AVO:STATE?</td>
</tr>
</tbody>
</table>

**TDR/TDT Pane Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>Description</th>
<th>SCPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal - Left Knob</td>
<td>Set horizontal scale/div</td>
<td>DISP:TDR:MEAS:X:SCAL:PDIvision</td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------------------</td>
<td>---------------------------------</td>
<td></td>
</tr>
<tr>
<td>Horizontal - Right Knob</td>
<td>Set horizontal reference level</td>
<td>DISP:TDR:MEAS:X:SCAL:RLEVel</td>
<td></td>
</tr>
<tr>
<td>Horizontal - Center Button</td>
<td>Select horizontal reference position</td>
<td>DISP:TDR:X:SCAL:RPOSition</td>
<td></td>
</tr>
<tr>
<td>Vertical - Left Knob</td>
<td>Set vertical scale/div</td>
<td>DISP:MEAS:Y:SCAL:PDIvision</td>
<td></td>
</tr>
<tr>
<td>Vertical - Right Knob</td>
<td>Set vertical reference level</td>
<td>DISP:MEAS:Y:SCAL:RLEVel</td>
<td></td>
</tr>
</tbody>
</table>

![TDR Diagram](image)

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>Description</th>
<th>SCPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter Selector Table</td>
<td>Select measurement parameter</td>
<td>CALC:TDR:MEAS:PARameter</td>
<td></td>
</tr>
<tr>
<td>Measure</td>
<td>Change selections of parameter table</td>
<td>CALC:TDR:MEAS:PARameter</td>
<td></td>
</tr>
<tr>
<td>Format</td>
<td>Select measurement format</td>
<td>CALC:TDR:MEAS:FORMat</td>
<td></td>
</tr>
<tr>
<td>Stimulus</td>
<td>Select time stimulus type</td>
<td>CALC:TDR:MEAS:TIME:TYPE</td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>Description</td>
<td>SCPI</td>
</tr>
<tr>
<td>------------</td>
<td>--------------</td>
<td>-----------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Start</td>
<td></td>
<td>Set gate start time</td>
<td>CALC:MEAS:FILT:GATE:TIME:STARt</td>
</tr>
<tr>
<td>Stop</td>
<td></td>
<td>Set gate stop time</td>
<td>CALC:MEAS:FILT:GATE:TIME:STOP</td>
</tr>
<tr>
<td>Gating</td>
<td></td>
<td>Set gating state</td>
<td>CALC:MEAS:FILT:GATE:TIME:STATe</td>
</tr>
<tr>
<td>Type</td>
<td></td>
<td>Select gating type</td>
<td>CALC:MEAS:FILT:GATE:TIME:TYPE</td>
</tr>
<tr>
<td>Gate Coupling</td>
<td></td>
<td>Open Gate Coupling dialog</td>
<td>None</td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>Description</td>
<td>SCPI</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Allocation - Mixed</td>
<td>Allocate measurements to mixed T/S-parameters</td>
<td>CALC:TDR:ALLoc MIX</td>
<td></td>
</tr>
<tr>
<td>Allocation - All T</td>
<td>Allocate measurements to all T-parameters</td>
<td>CALC:TDR:ALLoc TPAR</td>
<td></td>
</tr>
<tr>
<td>Allocation - All S</td>
<td>Allocate measurements to all S-parameters</td>
<td>CALC:TDR:ALLoc SPAR</td>
<td></td>
</tr>
<tr>
<td>Coupling - Time</td>
<td>Set time couple state</td>
<td>CALC:TDR:TIME:COUPle</td>
<td></td>
</tr>
<tr>
<td>Coupling - Marker</td>
<td>Set marker couple state</td>
<td>CALC:MARK:COUP:STATe</td>
<td></td>
</tr>
<tr>
<td>Coupling - Rise Time</td>
<td>Set rise time couple state</td>
<td>CALC:TDR:MEAS:TIME:STEP:COUPle</td>
<td></td>
</tr>
</tbody>
</table>
### Trace Annotation

Show the trace annotation on the active trace only

**SCPI:** `DISP:WIND:ANN:TRAC:SCOPe`

### Marker Readout

Show the marker readout on the active trace only

**SCPI:** `DISP:WIND:ANN:MARK:VISIble`

### Trace Settings Copy

Open Trace Settings Copy dialog

**SCPI:** None

### Eye/Mask Pane Commands

#### SCPI

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>Description</th>
<th>SCPI</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Draw Eye</strong></td>
<td></td>
<td>Measure and draw eye diagram</td>
<td><code>CALC:TDR:EYE:EXECute</code></td>
</tr>
<tr>
<td><strong>Abort</strong></td>
<td></td>
<td>Abort eye drawing</td>
<td>None</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td></td>
<td>Select eye input bit pattern</td>
<td><code>CALC:TDR:EYE:INP:BPAT:TYPE</code></td>
</tr>
<tr>
<td>Length</td>
<td>Select eye input bit length</td>
<td>CALC:TDR:EYE:INP:BPAT:LENTh</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------</td>
<td>-----------------------------</td>
<td></td>
</tr>
<tr>
<td>Advanced Waveform</td>
<td>Open Advanced Waveform Jitter dialog</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>User Pattern</td>
<td>Open Bit Pattern Editor dialog</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>One Lv.</td>
<td>Set eye input one level</td>
<td>CALC:TDR:EYE:INP:OLEVel</td>
<td></td>
</tr>
<tr>
<td>Zero Lv.</td>
<td>Set eye input zero level</td>
<td>CALC:TDR:EYE:INP:ZLEVel</td>
<td></td>
</tr>
<tr>
<td>Data Rate</td>
<td>Set eye input data rate</td>
<td>CALC:TDR:EYE:INP:DRATe</td>
<td></td>
</tr>
<tr>
<td>Rise Time (Pulldown)</td>
<td>Select eye input rise time threshold</td>
<td>CALC:TDR:EYE:INP:RTIM:THReshold</td>
<td></td>
</tr>
<tr>
<td>Rise Time (Numeric Edit)</td>
<td>Set eye input rise time value</td>
<td>CALC:TDR:EYE:IMP:RTIM:DATA</td>
<td></td>
</tr>
</tbody>
</table>

TDR

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>Description</th>
<th>SCPI</th>
</tr>
</thead>
</table>

7712
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>Description</th>
<th>SCPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto Scale</td>
<td>Set eye auto scale state On</td>
<td>DISP:TDR:EYE:Y:SCAL:AUTO:STAT ON</td>
<td></td>
</tr>
<tr>
<td>Scale / Div</td>
<td>Set eye vertical scale/div</td>
<td>DISP:TDR:EYE:Y:SCAL:PDIV</td>
<td></td>
</tr>
<tr>
<td>Offset</td>
<td>Set eye vertical reference level</td>
<td>DISP:TDR:EYE:Y:SCAL:RLEVel</td>
<td></td>
</tr>
<tr>
<td>Mask Test</td>
<td>Set eye mask test state</td>
<td>CALC:TDR:EYE:MASK:STATe</td>
<td></td>
</tr>
<tr>
<td>Mask Pattern</td>
<td>Open Mask Pattern dialog</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
CF_Setup Commands - VMC

Setup Commands - Vector Mixer/Converter Measurement Class

Only the Main and Layout Setup commands corresponding to the Vector Mixer/Converter measurement class are documented here. All other commands are identical to the Setup commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtabs on the graphic below.

<table>
<thead>
<tr>
<th>Main</th>
<th>System Setup</th>
<th>Internal Hardware</th>
<th>External Hardware</th>
</tr>
</thead>
</table>

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMC Setup...</td>
<td>Sweep</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Linear Frequency</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
<td></td>
</tr>
<tr>
<td>CW Time</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
<td></td>
</tr>
<tr>
<td>Segment Sweep</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
<td></td>
</tr>
<tr>
<td>Power</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
<td></td>
</tr>
<tr>
<td>X-Axis Point Spacing</td>
<td>SENSE:SEGment:X:SPACing</td>
<td>XAxisPointSpacing</td>
<td></td>
</tr>
<tr>
<td>Number of Points</td>
<td>SENSE:SWEep:POINts</td>
<td>NumberOfPoints</td>
<td></td>
</tr>
<tr>
<td>Section</td>
<td>Command/Tag</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------------</td>
<td>----------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>IF Bandwidth</strong></td>
<td>SENSE:BWIDth[:RESolution]</td>
<td>IF Bandwidth</td>
<td></td>
</tr>
<tr>
<td><strong>Power</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power On (All Channels)</td>
<td>OUTPut[:STATe]</td>
<td>SourcePowerState</td>
<td></td>
</tr>
<tr>
<td>Port Powers Coupled</td>
<td>SOURce:POWer:COUPle</td>
<td>CouplePorts</td>
<td></td>
</tr>
<tr>
<td><strong>DUT Input Port</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input Port</td>
<td>SENSE:MIXer:PMAP</td>
<td>SetDutPorts</td>
<td></td>
</tr>
<tr>
<td>Power Level</td>
<td>SOURce:POWer[:LEVEL][:IMMediate][:AMPLitude]</td>
<td>InputLinearPowerLevel</td>
<td></td>
</tr>
<tr>
<td>Source Leveling Mode</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
<td></td>
</tr>
<tr>
<td><strong>DUT Output Port</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output Port</td>
<td>SENSE:MIXer:PMAP</td>
<td>SetDutPorts</td>
<td></td>
</tr>
<tr>
<td>Power Level</td>
<td>SOURce:POWer[:LEVEL][:IMMediate][:AMPLitude]</td>
<td>InputLinearPowerLevel</td>
<td></td>
</tr>
<tr>
<td>Source Leveling Mode</td>
<td>SOURce:POWer:ALC[:MODE]</td>
<td>ALCLevelingMode</td>
<td></td>
</tr>
<tr>
<td>DUT Input Port Power Sweep</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start Power</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOURce:POWer:PORT:STARt</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>StartPowerEx</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stop Power</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOURce:POWer:PORT:STOP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>StopPowerEx</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Points</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>None</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>None</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power Step</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>None</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>None</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DUT Output Port Power Sweep</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Power</td>
</tr>
<tr>
<td>SOURce:POWer:PORT:STARt</td>
</tr>
<tr>
<td>StartPowerEx</td>
</tr>
<tr>
<td>Stop Power</td>
</tr>
<tr>
<td>SOURce:POWer:PORT:STOP</td>
</tr>
<tr>
<td>StopPowerEx</td>
</tr>
<tr>
<td>Path Configuration...</td>
</tr>
<tr>
<td>SENSe:PATH:CONFig:ELEMent[:STATe]</td>
</tr>
<tr>
<td>Element</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mixer Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
</tr>
<tr>
<td>SENSe:MIXer:CALCulate</td>
</tr>
<tr>
<td>Calculate</td>
</tr>
<tr>
<td>LO1</td>
</tr>
<tr>
<td>SENSe:MIXer:LO:FREQuency:FIXed</td>
</tr>
<tr>
<td>LOFixedFrequency</td>
</tr>
<tr>
<td>Input &gt; LO</td>
</tr>
<tr>
<td>SENSe:MIXer:LO:FREQuency:ILTI</td>
</tr>
<tr>
<td>IsInputGreaterThanLO</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>---------------------</td>
</tr>
<tr>
<td>Output</td>
</tr>
<tr>
<td>Mixer Power</td>
</tr>
<tr>
<td>Power On (All</td>
</tr>
<tr>
<td>Channels)</td>
</tr>
<tr>
<td>LO1 Power</td>
</tr>
<tr>
<td>LO2 Power</td>
</tr>
<tr>
<td>LO1 Swept Power</td>
</tr>
<tr>
<td>Start</td>
</tr>
<tr>
<td>Stop</td>
</tr>
<tr>
<td>Step</td>
</tr>
<tr>
<td>LO2 Swept Power</td>
</tr>
<tr>
<td>Start</td>
</tr>
<tr>
<td>Stop</td>
</tr>
<tr>
<td>Step</td>
</tr>
<tr>
<td>Path Configuration...</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td><strong>Mixer Setup</strong></td>
</tr>
<tr>
<td>Converter Stages</td>
</tr>
<tr>
<td>Add Source...</td>
</tr>
<tr>
<td>Path Configuration...</td>
</tr>
<tr>
<td>Save...</td>
</tr>
<tr>
<td>Load...</td>
</tr>
<tr>
<td><strong>Meas Class...</strong></td>
</tr>
<tr>
<td>Quick Start...</td>
</tr>
<tr>
<td>Other</td>
</tr>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>New Trace</td>
</tr>
<tr>
<td>New Channel</td>
</tr>
<tr>
<td>New Window</td>
</tr>
<tr>
<td>New Sheet</td>
</tr>
<tr>
<td>Delete</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Select</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Measure...</td>
</tr>
<tr>
<td>Measure Class...</td>
</tr>
</tbody>
</table>
CF_Sweep Commands

The Sweep softkeys vary depending on which measurement class is currently active. Click on a measurement class link below to view the corresponding softkeys.

Standard
Active Hot Parameters
Gain Compression
Differential I/Q
IM Spectrum
Swept IMD

Modulation Distortion and Modulation Distortion Converters
Noise Figure Cold Source
Phase Noise
Gain Compression Converters
IM Spectrum Converters
Swept IMD Converters
Noise Figure Converters
Scalar Mixer/Converter + Phase
Vector Mixer/Converter
Spectrum Analyzer
# CF_Sweep Commands - DIQ

**Sweep Commands - Differential I/Q Measurement Class**

Click [here](#) to view links to Sweep commands for all Measurement Classes.

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td></td>
<td>SENSE:SWEep:POINts</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>X-axis Type</td>
<td></td>
<td>CALCulate:MEASure:X:AXIS:DOMain</td>
<td>XAxisDomain</td>
</tr>
</tbody>
</table>

### DIQ Setup...

### Sweep Timing Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sweep Time</td>
<td>Auto</td>
<td>SENSE:SWEep:TIME:AUTO</td>
<td>chan.SweepTime</td>
</tr>
<tr>
<td>Dwell Time</td>
<td></td>
<td>SENSE:SWEep:DWELI</td>
<td>DwellTime</td>
</tr>
<tr>
<td>Sweep Delay</td>
<td></td>
<td>SENSE:SWEep:DWELI:SDELay</td>
<td>SweepDelay</td>
</tr>
<tr>
<td>Sweep Mode</td>
<td>AUTO</td>
<td>SENSE:SWEep:GENeration</td>
<td>SweepGenerationMode</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>STEPPED</td>
<td>SENSE:SWEep:GENeration</td>
<td>SweepGenerationMode</td>
<td></td>
</tr>
<tr>
<td>Fast Sweep</td>
<td>ON</td>
<td>SENSE:SWEep:SPEed</td>
<td>SweepSpeedMode</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>SENSE:SWEep:SPEed</td>
<td>SweepSpeedMode</td>
</tr>
</tbody>
</table>

### Source Control Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse Setup...</td>
<td>Pulse Measurement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Standard Pulse</td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Pulse Profile</td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
</tbody>
</table>

### Pulse Timing

<table>
<thead>
<tr>
<th>Pulse Width</th>
<th>SENSE:SWEep:PULSe:PRIMary:WIDth</th>
<th>PrimaryWidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse Period</td>
<td>SENSE:SWEep:PULSe:PRIMary:PERiod</td>
<td>PrimaryPeriod</td>
</tr>
<tr>
<td>Pulse Frequency</td>
<td>SENSE:SWEep:PULSe:PRIMary:FREQuency</td>
<td>PrimaryFrequency</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td></td>
<td>7725</td>
<td></td>
</tr>
</tbody>
</table>
CF_Sweep Commands - GCA

Sweep Commands - Gain Compression Measurement Class

Only the Main, Sweep Timing, and Source Control commands corresponding to the Gain Compression measurement classes are documented here. The Segment Table commands are identical to the Sweep commands for the Standard measurement class and can be accessed by clicking [here](#) or by clicking on the softtab on the graphic below.

Click [here](#) to view links to Sweep commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td></td>
<td>SENSe:GCSetup:SWEep:FREQuency:POINTS</td>
<td>NumberOfFrequencyPoints</td>
</tr>
<tr>
<td>Sweep Type</td>
<td>Linear Frequency</td>
<td>SENSe:SWEep:TYPe</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Log Frequency</td>
<td>SENSe:SWEep:TYPe</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Power Sweep</td>
<td>SENSe:SWEep:TYPe</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>CW Time</td>
<td>SENSe:SWEep:TYPe</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Segment Sweep</td>
<td>SENSe:SWEep:TYPe</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Phase Sweep</td>
<td>SENSe:SWEep:TYPe</td>
<td>SweepType</td>
</tr>
<tr>
<td>Start</td>
<td>SENSE:FREQuency:STARt</td>
<td>StartFrequency</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>Stop</td>
<td>SENSE:FREQuency:STOP</td>
<td>StopFrequency</td>
<td></td>
</tr>
<tr>
<td>X-axis Type</td>
<td>SENSE:FOM:DISPlay:SElECT</td>
<td>DisplayRange</td>
<td></td>
</tr>
<tr>
<td>GCA Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Sweep Timing Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>SENSE:SWEep:TIME:AUTO</td>
<td>chan.SweepTime</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SENSE:SWEep:TIME[:STOP]</td>
<td>chan.SweepTime</td>
</tr>
<tr>
<td>Sweep Time</td>
<td>Auto</td>
<td>SENSE:SWEep:TIME:AUTO</td>
<td>chan.SweepTime</td>
</tr>
<tr>
<td>Dwell Time</td>
<td></td>
<td>SENSE:SWEep:DWELI</td>
<td>DwellTime</td>
</tr>
<tr>
<td>Sweep Delay</td>
<td></td>
<td>SENSE:SWEep:DWELI:SDElay</td>
<td>SweepDelay</td>
</tr>
<tr>
<td>Fast Sweep</td>
<td>ON</td>
<td>SENSE:SWEep:SPEeed</td>
<td>SweepSpeedMode</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>SENSE:SWEep:SPEeed</td>
<td>SweepSpeedMode</td>
</tr>
</tbody>
</table>

**Source Control Tab Commands**
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency Offset...</td>
<td>Frequency Offset (ON/OFF)</td>
<td>SENSE:FOM[:STATe]</td>
<td>State</td>
</tr>
<tr>
<td></td>
<td>Mode - Coupled and Uncoupled</td>
<td>SENSE:FOM:RANGe:COUPled</td>
<td>Coupled</td>
</tr>
<tr>
<td>Sweep Type</td>
<td></td>
<td>SENSE:FOM:RANGe:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Settings</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td></td>
<td>SENSE:FOM:RANGe:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td></td>
<td>SENSE:FOM:RANGe:FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>Annotation - Primary, Source, and Receivers</td>
<td></td>
<td>SENSE:FOM:RANGe:NAME?</td>
<td>Name</td>
</tr>
<tr>
<td>X-Axis Point Spacing</td>
<td></td>
<td>SENSE:FOM:RANGe:SEGMent:SWEep:POINts</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td><strong>Pulse Setup...</strong></td>
<td><strong>Pulse Measurement</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td></td>
<td>SENSE:SWEep:PULSE:MODE</td>
<td>PulseMeasMode</td>
</tr>
<tr>
<td>Standard Pulse</td>
<td></td>
<td>SENSE:SWEep:PULSE:MODE</td>
<td>PulseMeasMode</td>
</tr>
<tr>
<td>Pulse Profile</td>
<td>SENSe:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td><strong>Pulse Timing</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Width</td>
<td>SENSe:SWEep:PULSe:PRIMary:WIDth</td>
<td>PrimaryWidth</td>
<td></td>
</tr>
<tr>
<td>Pulse Period</td>
<td>SENSe:SWEep:PULSe:PRIMary:PERiod</td>
<td>PrimaryPeriod</td>
<td></td>
</tr>
<tr>
<td>Pulse Frequency</td>
<td>SENSe:SWEep:PULSe:PRIMary:FREQuency</td>
<td>PrimaryFrequency</td>
<td></td>
</tr>
<tr>
<td>DC Source...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Global Source</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# CF_Sweep Commands - GCX

## Sweep Commands - Gain Compression Converters Measurement Class

Click [here](#) to view links to Sweep commands for all Measurement Classes.

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td></td>
<td>SENSE:SWEep:POINTs</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>X-Axis Type</td>
<td></td>
<td>CALCulate:MEASure:MIXer:XAXis</td>
<td>ActiveXAxisRange</td>
</tr>
<tr>
<td>Sweep Type</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GCX Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Sweep Timing Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sweep Time</td>
<td>Auto</td>
<td>SENSE:SWEep:TIME:AUTO</td>
<td>chan.SweepTime</td>
</tr>
<tr>
<td>Dwell Time</td>
<td></td>
<td>SENSE:SWEep:DWELI</td>
<td>DwellTime</td>
</tr>
<tr>
<td>Sweep Delay</td>
<td>SENSE:SWEep:DWELI:SDElay</td>
<td>SweepDelay</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>Fast Sweep</td>
<td>SENSE:SWEep:SPEed</td>
<td>SweepSpeedMode</td>
<td></td>
</tr>
<tr>
<td>ON</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OFF</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Source Control Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse Setup...</td>
<td>Pulse Measurement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Standard Pulse</td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Pulse Profile</td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
</tbody>
</table>

**Pulse Timing**

<table>
<thead>
<tr>
<th>Pulse Width</th>
<th>SENSE:SWEep:PULSe:PRIMary:WIDth</th>
<th>PrimaryWidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse Period</td>
<td>SENSE:SWEep:PULSe:PRIMary:PERiod</td>
<td>PrimaryPeriod</td>
</tr>
<tr>
<td>Pulse Frequency</td>
<td>SENSE:SWEep:PULSe:PRIMary:FREQuency</td>
<td>PrimaryFrequency</td>
</tr>
<tr>
<td>Embedded LO...</td>
<td>Embedded LO Mode On</td>
<td>SENSE:MIXer:LO:FREQuency:MODE</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Tuning Point</td>
<td>SENSE:MIXer:ELO:NORMalize:POINT</td>
<td>NormalizePoint</td>
</tr>
<tr>
<td>LO Frequency Delta</td>
<td>SENSE:MIXer:ELO:LO:DELTA</td>
<td>LOFrequencyDelta</td>
</tr>
<tr>
<td>Tuning Settings</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Broadband and precise</td>
<td>SENSE:MIXer:ELO:TUNing:MODE</td>
<td>TuningMode</td>
</tr>
<tr>
<td>Precise only</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Disable tuning</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Sweep Span</td>
<td>SENSE:MIXer:ELO:TUNing:SPAN</td>
<td>BroadbandTuningSpan</td>
</tr>
<tr>
<td>Max Iterations</td>
<td>SENSE:MIXer:ELO:TUNing:ITERations</td>
<td>MaxPreciseTuningIterations</td>
</tr>
<tr>
<td>Tolerance</td>
<td>SENSE:MIXer:ELO:TUNing:TOLerance</td>
<td>PreciseTuningTolerance</td>
</tr>
<tr>
<td>Tuning IFBW</td>
<td>SENSE:MIXer:ELO:TUNing:IFBW</td>
<td>TuningIFBW</td>
</tr>
<tr>
<td>Tune every</td>
<td>SENSE:MIXer:ELO:TUNing:INTerval</td>
<td>TuningSweepInterval</td>
</tr>
</tbody>
</table>
**CF_Sweep Commands - HotS22**

**Sweep Commands - Active Hot Parameters**

Only the Main Sweep commands corresponding to the Active Hot Parameters measurement class are documented here. The other commands are identical to the Sweep commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtab on the graphic below.

Click here to view links to Sweep commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td></td>
<td>SENSe:SWEep:POINts</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>Sweep Type</td>
<td>Linear Frequency</td>
<td>SENSe:ACTive:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Log Frequency</td>
<td>SENSe:ACTive:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Power Sweep</td>
<td>SENSe:ACTive:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Multisweep</td>
<td>SENSe:ACTive:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Start</td>
<td></td>
<td>SENSe:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td></td>
<td>SENSe:FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>X-axis Type</td>
<td></td>
<td>CALCulate:MEASure:X:AXIS:DOMain</td>
<td>DisplayDomain</td>
</tr>
</tbody>
</table>
AHP Setup...
CF_Sweep Commands - IMD

Sweep Commands - Swept IMD Measurement Class

Only the Main, Sweep Timing, and Source Control Sweep commands corresponding to the Swept IMD measurement class are documented here. The Segment Table commands are identical to the Sweep commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtab on the graphic below.

Click here to view links to Sweep commands for all Measurement Classes.

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td>Sweep fc</td>
<td>SENSe:SWEep:POINTs</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>Sweep Type</td>
<td>Sweep fc</td>
<td>SENSe:IMD:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Sweep DeltaF</td>
<td>SENSe:IMD:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Power Sweep</td>
<td>SENSe:IMD:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>CW Time</td>
<td>SENSe:IMD:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Segment Sweep</td>
<td>SENSe:IMD:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Start</td>
<td>Set DeltaF</td>
<td>SENS:IMD:FREQ:DFR</td>
<td>DeltaFrequency</td>
</tr>
<tr>
<td>Set Center Freq</td>
<td>SENS:IMD:FREQ:FCEN</td>
<td>FrequencyCenter</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>Start for Center Freq Sweep</td>
<td>SENS:IMD:FREQ:FCEN:STAR</td>
<td>FrequencyCenterStart</td>
<td></td>
</tr>
<tr>
<td>Center for Center Freq Sweep</td>
<td>SENS:IMD:FREQ:FCEN:CENT</td>
<td>FrequencyCenterCenter</td>
<td></td>
</tr>
<tr>
<td>Span for Center Freq Sweep</td>
<td>SENS:IMD:FREQ:FCEN:SPAN</td>
<td>FrequencyCenterSpan</td>
<td></td>
</tr>
<tr>
<td>Start for DeltaF Sweep</td>
<td>SENS:IMD:FREQ:DFR:STAR</td>
<td>DeltaFrequencyStart</td>
<td></td>
</tr>
<tr>
<td>Set F1 for CW and Power Sweep</td>
<td>SENS:IMD:FREQ:F1</td>
<td>F1Frequency</td>
<td></td>
</tr>
<tr>
<td>Set F2 for CW and Power Sweep</td>
<td>SENS:IMD:FREQ:F2</td>
<td>F2Frequency</td>
<td></td>
</tr>
<tr>
<td>Stop</td>
<td>Stop for Center Freq Sweep</td>
<td>SENS:IMD:FREQ:FCEN:STOP</td>
<td>FrequencyCenterStop</td>
</tr>
<tr>
<td>Stop for DeltaF Sweep</td>
<td>SENS:IMD:FREQ:DFR:STOP</td>
<td>DeltaFrequencyStop</td>
<td></td>
</tr>
<tr>
<td>X-axis Type</td>
<td>SENSe:FOM:DISPlay:SELect</td>
<td>DisplayRange</td>
<td></td>
</tr>
<tr>
<td>IMD Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Sweep Timing Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sweep Time</td>
<td>Auto</td>
<td><code>SENSe:SWEep:TIME:AUTO</code></td>
<td><code>chan.SweepTime</code></td>
</tr>
<tr>
<td>-------------</td>
<td>------------------</td>
<td>-------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Dwell Time</td>
<td><code>SENSe:SWEep:DWELI</code></td>
<td></td>
<td><code>DwellTime</code></td>
</tr>
<tr>
<td>Sweep Delay</td>
<td><code>SENSe:SWEep:DWELI:SDELay</code></td>
<td></td>
<td><code>SweepDelay</code></td>
</tr>
<tr>
<td>Fast Sweep</td>
<td>ON</td>
<td><code>SENSe:SWEep:SPEed</code></td>
<td><code>SweepSpeedMode</code></td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td><code>SENSe:SWEep:SPEed</code></td>
<td><code>SweepSpeedMode</code></td>
</tr>
</tbody>
</table>

**Source Control Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse Setup...</td>
<td>Pulse Measurement</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Off</td>
<td><code>SENSe:SWEep:PULSe:MODE</code></td>
<td><code>PulseMeasMode</code></td>
</tr>
<tr>
<td></td>
<td>Standard Pulse</td>
<td><code>SENSe:SWEep:PULSe:MODE</code></td>
<td><code>PulseMeasMode</code></td>
</tr>
<tr>
<td></td>
<td>Pulse Profile</td>
<td><code>SENSe:SWEep:PULSe:MODE</code></td>
<td><code>PulseMeasMode</code></td>
</tr>
<tr>
<td></td>
<td>Pulse Timing</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:SWEep:PULSe:PRIMary:WIDth</td>
<td>PrimaryWidth</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td>Pulse Width</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Period</td>
<td>SENSE:SWEep:PULSe:PRIMary:PERiod</td>
<td>PrimaryPeriod</td>
<td></td>
</tr>
<tr>
<td>Pulse Frequency</td>
<td>SENSE:SWEep:PULSe:PRIMary:FREQuency</td>
<td>PrimaryFrequency</td>
<td></td>
</tr>
<tr>
<td>DC Source...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Global Source</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## CF_Sweep Commands - IMDX

### Sweep Commands - Swept IMD Converters Measurement Class

Click [here](#) to view links to Sweep commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main</th>
<th>Sweep Timing</th>
<th>Source Control</th>
</tr>
</thead>
</table>

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td></td>
<td>SENSE:SWEep:POINts</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>Sweep Type</td>
<td></td>
<td>SENSE:IMD:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Sweep DeltaF</td>
<td></td>
<td>SENSE:IMD:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Power Sweep</td>
<td></td>
<td>SENSE:IMD:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>CW</td>
<td></td>
<td>SENSE:IMD:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>LO Power Sweep</td>
<td></td>
<td>SENSE:IMD:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>X-Axis Type</td>
<td></td>
<td>SENSE:MIXer:XAXis</td>
<td>ActiveXAxisRange</td>
</tr>
<tr>
<td>IMDX Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Sweep Timing Tab Commands

---

7739
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sweep Time</td>
<td>Auto</td>
<td>SENSE:SWEep:TIME:AUTO</td>
<td>chan.SweepTime</td>
</tr>
<tr>
<td>Dwell Time</td>
<td></td>
<td>SENSE:SWEep:DWELl</td>
<td>DwellTime</td>
</tr>
<tr>
<td>Sweep Delay</td>
<td></td>
<td>SENSE:SWEep:DWELl:SDElay</td>
<td>SweepDelay</td>
</tr>
<tr>
<td>Fast Sweep</td>
<td>ON</td>
<td>SENSE:SWEep:SPEed</td>
<td>SweepSpeedMode</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>SENSE:SWEep:SPEed</td>
<td>SweepSpeedMode</td>
</tr>
</tbody>
</table>

### Source Control Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse Setup...</td>
<td>Pulse Measurement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td></td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
</tr>
<tr>
<td>Standard Pulse</td>
<td></td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
</tr>
<tr>
<td>Pulse Profile</td>
<td></td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
</tr>
<tr>
<td>Pulse Timing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>Pulse Width</td>
<td>SENSE:SWEep:PULSe:PRIMary:WIDth</td>
<td>PrimaryWidth</td>
<td></td>
</tr>
<tr>
<td>Pulse Period</td>
<td>SENSE:SWEep:PULSe:PRIMary:PERiod</td>
<td>PrimaryPeriod</td>
<td></td>
</tr>
<tr>
<td>Pulse Frequency</td>
<td>SENSE:SWEep:PULSe:PRIMary:FREQuency</td>
<td>PrimaryFrequency</td>
<td></td>
</tr>
<tr>
<td>Embedded LO...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Embedded LO Mode On</td>
<td>SENSE:MIXer:LO:FREQuency:MODE</td>
<td>LORangeMode</td>
<td></td>
</tr>
<tr>
<td>Tuning Point</td>
<td>SENSE:MIXer:ELO:NORMalize:POINT</td>
<td>NormalizePoint</td>
<td></td>
</tr>
<tr>
<td>LO Frequency Delta</td>
<td>SENSE:MIXer:ELO:LO:DELTA</td>
<td>LOFrequencyDelta</td>
<td></td>
</tr>
<tr>
<td>Tuning Settings</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Broadband and precise</td>
<td>SENSE:MIXer:ELO:TUNing:MODE</td>
<td>TuningMode</td>
<td></td>
</tr>
<tr>
<td>Precise only</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Disable tuning</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sweep Span</td>
<td>SENSE:MIXer:ELO:TUNing:SPAN</td>
<td>BroadbandTuningSpan</td>
<td></td>
</tr>
<tr>
<td>Max Iterations</td>
<td>SENSE:MIXer:ELO:TUNing:ITERations</td>
<td>MaxPreciseTuningIterations</td>
<td></td>
</tr>
<tr>
<td>Tolerance</td>
<td>SENSE:MIXer:ELO:TUNing:TOLerance</td>
<td>PreciseTuningTolerance</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------</td>
<td>------------------------</td>
<td></td>
</tr>
<tr>
<td>Tuning IFBW</td>
<td>SENSE:MIXer:ELO:TUNing:IFBW</td>
<td>TuningIFBW</td>
<td></td>
</tr>
<tr>
<td>Tune every</td>
<td>SENSE:MIXer:ELO:TUNing:INTerval</td>
<td>TuningSweepInterval</td>
<td></td>
</tr>
</tbody>
</table>
## CF_Sweep Commands - IMS

**Sweep Commands - IM Spectrum Measurement Class**

Click [here](#) to view links to Sweep commands for all Measurement Classes.

### Main Source Control

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td></td>
<td>SENSe:SWEep:POINts</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>Sweep Type</td>
<td>Linear</td>
<td>SENSe:IMS:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>2nd Order Spectrum</td>
<td>SENSe:IMS:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>3rd Order Spectrum</td>
<td>SENSe:IMS:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Nth Order Spectrum</td>
<td>SENSe:IMS:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Start</td>
<td></td>
<td>SENSe:IMS:RESPonse:STARt</td>
<td>SpectrumStartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td></td>
<td>SENSe:IMS:RESPonse:STOP</td>
<td>SpectrumStopFrequency</td>
</tr>
<tr>
<td>X-axis Type</td>
<td></td>
<td>SENSe:FOM:DISPlay:SELect</td>
<td>DisplayRange</td>
</tr>
<tr>
<td>IMS Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>-----------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td>Pulse Measurement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td>SENSE:SWEP:PULSE:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Pulse Profile</td>
<td>SENSE:SWEP:PULSE:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Pulse Timing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Width</td>
<td>SENSE:SWEP:PULSE:PRIMARY:WIDth</td>
<td>PrimaryWidth</td>
<td></td>
</tr>
<tr>
<td>Pulse Period</td>
<td>SENSE:SWEP:PULSE:PRIMARY:PERiod</td>
<td>PrimaryPeriod</td>
<td></td>
</tr>
<tr>
<td>Pulse Frequency</td>
<td>SENSE:SWEP:PULSE:PRIMARY:FREQuency</td>
<td>PrimaryFrequency</td>
<td></td>
</tr>
<tr>
<td>DC Source...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Global Source</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## CF_Sweep Commands - IMSX

**Sweep Commands - IM Spectrum Converters Measurement Class**

Click [here](#) to view links to Sweep commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Source Control</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main Tab Commands</strong></td>
</tr>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>Number of Points</td>
</tr>
<tr>
<td>Sweep Type</td>
</tr>
<tr>
<td>Start</td>
</tr>
<tr>
<td>Stop</td>
</tr>
<tr>
<td>X-axis Type</td>
</tr>
<tr>
<td>IMSX Setup...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Control Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Softkey</td>
</tr>
<tr>
<td>Pulse Setup...</td>
</tr>
<tr>
<td>Mode</td>
</tr>
<tr>
<td>---------------------------</td>
</tr>
<tr>
<td>Off</td>
</tr>
<tr>
<td>Standard Pulse</td>
</tr>
<tr>
<td>Pulse Profile</td>
</tr>
<tr>
<td><strong>Pulse Timing</strong></td>
</tr>
<tr>
<td>Pulse Width</td>
</tr>
<tr>
<td>Pulse Period</td>
</tr>
<tr>
<td>Pulse Frequency</td>
</tr>
<tr>
<td>Embedded LO...</td>
</tr>
<tr>
<td>Embedded LO Mode On</td>
</tr>
<tr>
<td>Tuning Point</td>
</tr>
<tr>
<td>LO Frequency Delta</td>
</tr>
<tr>
<td>Tuning Settings</td>
</tr>
<tr>
<td>Broadband and precise</td>
</tr>
<tr>
<td>Precise only</td>
</tr>
<tr>
<td>Feature</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>Disable tuning</td>
</tr>
<tr>
<td>Sweep Span</td>
</tr>
<tr>
<td>Max Iterations</td>
</tr>
<tr>
<td>Tolerance</td>
</tr>
<tr>
<td>Tuning IFBW</td>
</tr>
<tr>
<td>Tune every</td>
</tr>
</tbody>
</table>
CF_Sweep Commands - NF

Sweep Commands - Noise Figure Cold Source Measurement Class

Only the Main, Sweep Timing, and Source Control Sweep commands corresponding to the Noise Figure Cold Source measurement class are documented here. The Segment Table commands are identical to the Sweep commands for the Standard measurement class and can be accessed by clicking here or by clicking on the softtab on the graphic below.

Click here to view links to Sweep commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td></td>
<td>SENSE:SWEep:POINTS</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>Sweep Type</td>
<td>Linear Frequency</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Log Frequency</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>CW Time</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Segment Sweep</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Start</td>
<td></td>
<td>SENSE:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td></td>
<td>SENSE:FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>X-axis Type</td>
<td>SENSE:FOM:DISPLAY:SELECT</td>
<td>DisplayRange</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>--------------------------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td>NF Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Sweep Timing Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sweep Time</td>
<td>Auto</td>
<td>SENSE:SWEep:TIME:AUTO</td>
<td>chan.SweepTime</td>
</tr>
<tr>
<td>Dwell Time</td>
<td></td>
<td>SENSE:SWEep:DWELi</td>
<td>DwellTime</td>
</tr>
<tr>
<td>Sweep Delay</td>
<td></td>
<td>SENSE:SWEep:DWELi:SDELay</td>
<td>SweepDelay</td>
</tr>
<tr>
<td>Fast Sweep</td>
<td>ON</td>
<td>SENSE:SWEep:SPEed</td>
<td>SweepSpeedMode</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>SENSE:SWEep:SPEed</td>
<td>SweepSpeedMode</td>
</tr>
</tbody>
</table>

### Source Control Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency Offset (ON/OFF)</td>
<td>SENSE:FOM[:STATE]</td>
<td>State</td>
<td></td>
</tr>
<tr>
<td><strong>Frequency Offset...</strong></td>
<td><strong>Mode</strong> - Coupled and Un-coupled</td>
<td>SENSE:FOM:RANGE:COUPled</td>
<td>Coupled</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------</td>
<td>--------------------------</td>
<td>---------</td>
</tr>
<tr>
<td><strong>Sweep Type</strong></td>
<td></td>
<td>SENSE:FOM:RANGE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td><strong>Settings</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Start</strong></td>
<td></td>
<td>SENSE:FOM:RANGE:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td><strong>Stop</strong></td>
<td></td>
<td>SENSE:FOM:RANGE:FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td><strong>Annotation - Primary, Source, and Receivers</strong></td>
<td>SENSE:FOM:RANGE:NAME?</td>
<td>Name</td>
<td></td>
</tr>
<tr>
<td><strong>X-Axis Point Spacing</strong></td>
<td></td>
<td>SENSE:FOM:RANGE:SEGMent:SWEep:POINts</td>
<td>NumberOfPoints</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Pulse Setup...</strong></th>
<th><strong>Pulse Measurement</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Off</strong></td>
<td>SENSE:SWEep:PULSE:MODE</td>
</tr>
<tr>
<td><strong>Standard Pulse</strong></td>
<td>SENSE:SWEep:PULSE:MODE</td>
</tr>
<tr>
<td><strong>Pulse Profile</strong></td>
<td>SENSE:SWEep:PULSE:MODE</td>
</tr>
<tr>
<td><strong>Pulse Timing</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Pulse Width</strong></td>
<td>SENSE:SWEep:PULSE:PRIMary:WIDth</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Pulse Period</td>
<td>SENSe:SWEep:PULSe:PRIMary:PERiod</td>
</tr>
<tr>
<td>Pulse Frequency</td>
<td>SENSe:SWEep:PULSe:PRIMary:FREQuency</td>
</tr>
<tr>
<td>DC Source...</td>
<td></td>
</tr>
<tr>
<td>Global Source</td>
<td></td>
</tr>
</tbody>
</table>
## CF_Sweep Commands - NFX

Sweep Commands - Noise Figure Converters Measurement Class

Click [here](#) to view links to Sweep commands for all Measurement Classes.

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td></td>
<td>SENSE:SWEP:POINTS</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>Sweep Type</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X-Axis Type</td>
<td></td>
<td>SENSE:MIXer:XAXis</td>
<td>ActiveXAxisRange</td>
</tr>
<tr>
<td>NFX Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Sweep Timing Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dwell Time</td>
<td></td>
<td>SENSE:SWEP:DWELI</td>
<td>DwellTime</td>
</tr>
<tr>
<td>Source Control Tab Commands</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Softkey</strong></td>
<td><strong>Sub-item</strong></td>
<td><strong>SCPI</strong></td>
<td><strong>COM</strong></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td>Pulse Measurement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td></td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
</tr>
<tr>
<td>Standard Pulse</td>
<td></td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
</tr>
<tr>
<td>Pulse Profile</td>
<td></td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
</tr>
<tr>
<td>Pulse Timing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Width</td>
<td></td>
<td>SENSE:SWEep:PULSe:PRIMary:WIDth</td>
<td>PrimaryWidth</td>
</tr>
<tr>
<td>Pulse Period</td>
<td></td>
<td>SENSE:SWEep:PULSe:PRIMary:PERiod</td>
<td>PrimaryPeriod</td>
</tr>
<tr>
<td>Pulse Frequency</td>
<td></td>
<td>SENSE:SWEep:PULSe:PRIMary:FREQuency</td>
<td>PrimaryFrequency</td>
</tr>
<tr>
<td>DC Source...</td>
<td>Embedded LO Mode On</td>
<td>SENSe:MIXer:LO:FREQuency:MODE</td>
<td>LORangeMode</td>
</tr>
<tr>
<td>Embedded LO...</td>
<td>Tuning Point</td>
<td>SENSe:MIXer:ELO:NORMalize:POINT</td>
<td>NormalizePoint</td>
</tr>
<tr>
<td>LO Frequency Delta</td>
<td>LO Frequency Delta</td>
<td>SENSe:MIXer:ELO:LO:DELTA</td>
<td>LOFrequencyDelta</td>
</tr>
<tr>
<td>Tuning Settings</td>
<td>Broadband and precise</td>
<td>SENSe:MIXer:ELO:TUNing:MODE</td>
<td>TuningMode</td>
</tr>
<tr>
<td></td>
<td>Precise only</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Disable tuning</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Sweep Span</td>
<td>SENSe:MIXer:ELO:TUNing:SPAN</td>
<td>BroadbandTuningSpan</td>
</tr>
<tr>
<td></td>
<td>Max Iterations</td>
<td>SENSe:MIXer:ELO:TUNing:ITERations</td>
<td>MaxPreciseTuningIterations</td>
</tr>
<tr>
<td></td>
<td>Tolerance</td>
<td>SENSe:MIXer:ELO:TUNing:TOLerance</td>
<td>PreciseTuningTolerance</td>
</tr>
<tr>
<td></td>
<td>Tuning IFBW</td>
<td>SENSe:MIXer:ELO:TUNing:IFBW</td>
<td>TuningIFBW</td>
</tr>
<tr>
<td></td>
<td>Tune every</td>
<td>SENSe:MIXer:ELO:TUNing:INTerval</td>
<td>TuningSweepInterval</td>
</tr>
</tbody>
</table>
# CF_Sweep Commands - PN

## Sweep Commands - Phase Noise Measurement Class

Click [here](#) to view links to Sweep commands for all Measurement Classes.

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td></td>
<td>SENSE:SWEep:POINts</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>Start</td>
<td></td>
<td>SENSE:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td></td>
<td>SENSE:FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>Phase Noise Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Sweep Timing Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sweep Time</td>
<td>Auto</td>
<td>SENSE:SWEep:TIME:AUTO</td>
<td>chan.SweepTime</td>
</tr>
<tr>
<td>Dwell Time</td>
<td>SENSE:SWEep:DWELLI</td>
<td>DwellTime</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>--------------------</td>
<td>----------</td>
<td></td>
</tr>
<tr>
<td>Sweep Delay</td>
<td>SENSE:SWEep:DWELLI:SDELay</td>
<td>SweepDelay</td>
<td></td>
</tr>
<tr>
<td>Sweep Mode</td>
<td>AUTO</td>
<td>SENSE:SWEep:GENeration</td>
<td>SweepGenerationMode</td>
</tr>
<tr>
<td></td>
<td>STEPPED</td>
<td>SENSE:SWEep:GENeration</td>
<td>SweepGenerationMode</td>
</tr>
<tr>
<td>Sweep Sequence</td>
<td>STD</td>
<td>SENSE:SWEep:GENeration:POINtsweep</td>
<td>PointSweepState</td>
</tr>
<tr>
<td></td>
<td>POINT</td>
<td>SENSE:SWEep:GENeration:POINtsweep</td>
<td>PointSweepState</td>
</tr>
<tr>
<td>Fast Sweep</td>
<td>ON</td>
<td>SENSE:SWEep:SPEed</td>
<td>SweepSpeedMode</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>SENSE:SWEep:SPEed</td>
<td>SweepSpeedMode</td>
</tr>
</tbody>
</table>

**Source Control Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse Setup...</td>
<td></td>
<td>Pulse Measurement</td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td></td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
</tr>
<tr>
<td>Standard Pulse</td>
<td></td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
</tr>
<tr>
<td>Property</td>
<td>Command</td>
<td>State</td>
<td></td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>Pulse Profile</td>
<td>SENSE:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Pulse Timing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Width</td>
<td>SENSE:SWEep:PULSe:PRIMary:WIDth</td>
<td>PrimaryWidth</td>
<td></td>
</tr>
<tr>
<td>Pulse Period</td>
<td>SENSE:SWEep:PULSe:PRIMary:PERiod</td>
<td>PrimaryPeriod</td>
<td></td>
</tr>
<tr>
<td>Pulse Frequency</td>
<td>SENSE:SWEep:PULSe:PRIMary:FREQuency</td>
<td>PrimaryFrequency</td>
<td></td>
</tr>
<tr>
<td>Properties</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Narrowband</td>
<td>SENSE:SWEep:PULSe:WIDeband[:STATe]</td>
<td>WideBandDectionState</td>
<td></td>
</tr>
<tr>
<td>Wideband</td>
<td>SENSE:SWEep:PULSe:WIDeband[:STATe]</td>
<td>WideBandDectionState</td>
<td></td>
</tr>
<tr>
<td>SW Gating</td>
<td>SENSE:SWEep:PULSe:SWGate</td>
<td>SoftwareGateState</td>
<td></td>
</tr>
<tr>
<td>Autoselect IF Path Gain and</td>
<td>SENSE:SWEep:PULSe:IFGain[:AUTO]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Loss</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IF Path...</td>
<td>SENSE:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Setting</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------------</td>
<td>------------------</td>
<td></td>
</tr>
<tr>
<td>Optimize Pulse Frequency</td>
<td>SENSe:SWEep:PULSe:PRF[:AUTO]</td>
<td>AutoOptimizePRF</td>
<td></td>
</tr>
<tr>
<td>Autoselect Profile Sweep Time</td>
<td>SENSe:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>IFBW</td>
<td>SENSe:SWEep:PULSe:CWTime[:AUTO]</td>
<td>AutoIFBandWidth</td>
<td></td>
</tr>
<tr>
<td>Sweep Time</td>
<td>SENSe:SWEep:TIME</td>
<td>chan.SweepTime</td>
<td></td>
</tr>
<tr>
<td>Number of Points</td>
<td>SENSe:SWEep:POINts</td>
<td>NumberOfPoints</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Measurement Timing</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Width</td>
<td>SENSe:SWEep:PULSe:TIMing</td>
<td>AutoPulseTiming</td>
<td></td>
</tr>
<tr>
<td>Delay</td>
<td>SENSe:SWEep:PULSe:TIMing</td>
<td>AutoPulseTiming</td>
<td></td>
</tr>
<tr>
<td>Pulse Gen</td>
<td>SENSe:SWEep:PULSe:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Master Pulse Trigger</td>
<td>SENSe:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
<td></td>
</tr>
<tr>
<td>Autoselect Width &amp; Delay</td>
<td>SENSe:SWEep:PULSe:TIMing</td>
<td>AutoPulseTiming</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------------</td>
<td>---------------------------</td>
<td></td>
</tr>
<tr>
<td>Autoselect Pulse Generators</td>
<td>SENSe:SWEep:PULSe:DRIVe[:AUTO]</td>
<td>AutoSelectPulseGen</td>
<td></td>
</tr>
<tr>
<td>Pulse Generators...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Width</td>
<td>SENSe:PULSe:WIDTh</td>
<td>Width</td>
<td></td>
</tr>
<tr>
<td>Delay</td>
<td>SENSe:PULSe:DELay</td>
<td>Delay</td>
<td></td>
</tr>
<tr>
<td>Invert</td>
<td>SENSe:PULSe:INVert</td>
<td>Invert</td>
<td></td>
</tr>
<tr>
<td>Enable</td>
<td>SENSe:PULSe[:STATe]</td>
<td>State</td>
<td></td>
</tr>
<tr>
<td>Trigger</td>
<td>SENSe:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
<td></td>
</tr>
<tr>
<td>Frequency</td>
<td>SENSe:SWEep:PULSe:PRIMary:FREQuency</td>
<td>PrimaryFrequency</td>
<td></td>
</tr>
<tr>
<td>Period</td>
<td>SENSe:SWEep:PULSe:PRIMary:PERiod</td>
<td>PrimaryPeriod</td>
<td></td>
</tr>
<tr>
<td>Enable Source x Modulator</td>
<td>SENSe:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
<td></td>
</tr>
<tr>
<td>Modulator Drive</td>
<td>SENSe:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
<td></td>
</tr>
<tr>
<td>Offset Pulse using Modulator and ADC Delays</td>
<td>SENSe:PULSe:HWDelay[:STATe]</td>
<td>EnableOffsetDelays</td>
<td></td>
</tr>
<tr>
<td>Modulator Delay</td>
<td><code>SENSe:PULSe:HWDelay:MODulator</code></td>
<td>ModulatorDelay</td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>Fixed ADC Delay</td>
<td><code>SENSe:PULSe:HWDelay:ADC?</code></td>
<td>FixedADCDelay</td>
<td></td>
</tr>
<tr>
<td>Synchronize ADCs using pulse trigger</td>
<td><code>SENSe:PATH:CONFig:ELEMENT[:STATe]</code></td>
<td>Element</td>
<td></td>
</tr>
<tr>
<td>Pulse4 Output Indicates</td>
<td><code>SENSe:PULSe4:OPTion</code></td>
<td>Pulse4OutAsADCActivity</td>
<td></td>
</tr>
<tr>
<td>All ADC Activity</td>
<td><code>SENSe:PULSe4:OPTion:MODE ALL</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Trace ADC Activity</td>
<td><code>SENSe:PULSe4:OPTion:MODE TRACe</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Pulse Trigger...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trigger Source</td>
<td><code>SENSe:PATH:CONFig:ELEMENT[:STATe]</code></td>
<td>Element</td>
<td></td>
</tr>
<tr>
<td>Trigger Level/Edge</td>
<td><code>SENSe:PULSe:TTYPE</code></td>
<td>TriggerInType</td>
<td></td>
</tr>
<tr>
<td>Synchronize ADCs using pulse trigger</td>
<td><code>SENSe:PULSe[:STATe]</code></td>
<td>State</td>
<td></td>
</tr>
<tr>
<td>Trigger...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DC Source...</td>
<td><code>SOURce:DC:ENABle</code></td>
<td>EnableAllOutput</td>
<td></td>
</tr>
<tr>
<td>Enable DC Outputs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SOURce:DC:STATe</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------</td>
<td>----------------------</td>
<td></td>
</tr>
<tr>
<td>State</td>
<td>State</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start DC</td>
<td>SOURce:DC:STARt</td>
<td>Start</td>
<td></td>
</tr>
<tr>
<td>Stop DC</td>
<td>SOURce:DC:STOP</td>
<td>Stop</td>
<td></td>
</tr>
<tr>
<td><strong>Limits...</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>State</td>
<td>SOURce:DC:STATe</td>
<td>State</td>
<td></td>
</tr>
<tr>
<td>Min</td>
<td>SOURce:DC:LIMIt:MINimum</td>
<td>LimitMin</td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>SOURce:DC:LIMIt:MAXimum</td>
<td>LimitMax</td>
<td></td>
</tr>
</tbody>
</table>
## CF_Sweep Commands - SA

Sweep Commands - Spectrum Analyzer Measurement Class

Click [here](#) to view links to Sweep commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
<td><strong>Sub-item</strong></td>
</tr>
<tr>
<td>Number of Points</td>
<td></td>
</tr>
<tr>
<td>Sweep Type</td>
<td>Linear Frequency</td>
</tr>
<tr>
<td></td>
<td>Segment Sweep</td>
</tr>
<tr>
<td>Start</td>
<td></td>
</tr>
<tr>
<td>Stop</td>
<td></td>
</tr>
<tr>
<td>X-axis Type</td>
<td></td>
</tr>
<tr>
<td>SA Setup...</td>
<td></td>
</tr>
</tbody>
</table>

### Sweep Timing Tab Commands
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>SENSe:SWEep:TIME:AUTO</strong></td>
<td><strong>chan.SweepTime</strong></td>
</tr>
<tr>
<td>Sweep Time</td>
<td>Auto</td>
<td><strong>SENSe:SWEep:TIME[:STOP]</strong></td>
<td><strong>chan.SweepTime</strong></td>
</tr>
<tr>
<td></td>
<td>Manual</td>
<td><strong>SENSe:SWEep:DWELI</strong></td>
<td><strong>DwellTime</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>SENSe:SWEep:DWELI:SDELay</strong></td>
<td><strong>SweepDelay</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>SENSe:SWEep:SPeed</strong></td>
<td><strong>SweepSpeedMode</strong></td>
</tr>
<tr>
<td></td>
<td>ON</td>
<td><strong>SENSe:SWEep:SPeed</strong></td>
<td><strong>SweepSpeedMode</strong></td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td><strong>SENSe:SWEep:SPeed</strong></td>
<td><strong>SweepSpeedMode</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>SENSe:SWEep:PULSe:MODE</strong></td>
<td><strong>PulseMeasMode</strong></td>
</tr>
<tr>
<td></td>
<td>Off</td>
<td><strong>SENSe:SWEep:PULSe:MODE</strong></td>
<td><strong>PulseMeasMode</strong></td>
</tr>
<tr>
<td></td>
<td>Standard Pulse</td>
<td><strong>SENSe:SWEep:PULSe:MODE</strong></td>
<td><strong>PulseMeasMode</strong></td>
</tr>
<tr>
<td></td>
<td>Pulse Profile</td>
<td><strong>SENSe:SWEep:PULSe:MODE</strong></td>
<td><strong>PulseMeasMode</strong></td>
</tr>
</tbody>
</table>

**Source Control Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse Setup</td>
<td></td>
<td><strong>SENSe:SWEep:PULSe:MODE</strong></td>
<td><strong>PulseMeasMode</strong></td>
</tr>
<tr>
<td></td>
<td>Off</td>
<td><strong>SENSe:SWEep:PULSe:MODE</strong></td>
<td><strong>PulseMeasMode</strong></td>
</tr>
<tr>
<td></td>
<td>Standard Pulse</td>
<td><strong>SENSe:SWEep:PULSe:MODE</strong></td>
<td><strong>PulseMeasMode</strong></td>
</tr>
<tr>
<td></td>
<td>Pulse Profile</td>
<td><strong>SENSe:SWEep:PULSe:MODE</strong></td>
<td><strong>PulseMeasMode</strong></td>
</tr>
</tbody>
</table>
### Pulse Timing

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command Scheme</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse Width</td>
<td>SENSE:SWEP:PULSE:PRIMary:WIDth</td>
<td>PrimaryWidth</td>
</tr>
<tr>
<td>Pulse Period</td>
<td>SENSE:SWEP:PULSE:PRIMary:PERiod</td>
<td>PrimaryPeriod</td>
</tr>
<tr>
<td>Pulse Frequency</td>
<td>SENSE:SWEP:PULSE:PRIMary:FREQuency</td>
<td>PrimaryFrequency</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command Scheme</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autoselect IF Path Gain and Loss</td>
<td>SENSE:SWEP:PULSE:IFGain[:AUTO]</td>
<td>None</td>
</tr>
<tr>
<td>IF Path...</td>
<td>SENSE:PATH:CONFig:ELEMENT[:STATE]</td>
<td>Element</td>
</tr>
<tr>
<td>Autoselect Profile Sweep Time</td>
<td>SENSE:SWEP:PULSE:MODE</td>
<td>PulseMeasMode</td>
</tr>
</tbody>
</table>

### Measurement Timing

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command Scheme</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting</td>
<td>Command</td>
<td>Mode</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Width</td>
<td>SENSE:SWEep:PULSE:TIMing AutoPulseTiming</td>
<td></td>
</tr>
<tr>
<td>Delay</td>
<td>SENSE:SWEep:PULSE:TIMing AutoPulseTiming</td>
<td></td>
</tr>
<tr>
<td>Pulse Gen</td>
<td>SENSE:SWEep:PULSE:MODE PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Master Pulse Trigger</td>
<td>SENSE:PATH:CONFig:ELEMENT[:STATe] Element</td>
<td></td>
</tr>
<tr>
<td>Autoselect Width &amp; Delay</td>
<td>SENSE:SWEep:PULSE:TIMing AutoPulseTiming</td>
<td></td>
</tr>
</tbody>
</table>

## Pulse Generators...##

<table>
<thead>
<tr>
<th>Setting</th>
<th>Command</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Width</td>
<td>SENSE:PULSE:WIDTH Width</td>
<td></td>
</tr>
<tr>
<td>Delay</td>
<td>SENSE:PULSE:DELay Delay</td>
<td></td>
</tr>
<tr>
<td>Invert</td>
<td>SENSE:PULSE:INVert Invert</td>
<td></td>
</tr>
<tr>
<td>Enable</td>
<td>SENSE:PULSE[:STATe] State</td>
<td></td>
</tr>
<tr>
<td>Trigger</td>
<td>SENSE:PATH:CONFig:ELEMENT[:STATe] Element</td>
<td></td>
</tr>
<tr>
<td>Frequency</td>
<td>SENSE:SWEep:PULSE:MASTER:Frequency MasterFrequency</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>--------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Period</td>
<td>SENSE:SWEp:PUlS:e:MASTer:PERiod</td>
<td>MasterPeriod</td>
</tr>
<tr>
<td>Enable Source x Modulator</td>
<td>SENSE:PATH:CONFig:ELEMENT[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td>Modulator Drive</td>
<td>SENSE:PATH:CONFig:ELEMENT[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td>Offset Pulse using Modulator and ADC Delays</td>
<td>SENSE:PUlS:e:HWDelay[:STATe]</td>
<td>EnableOffsetDelays</td>
</tr>
<tr>
<td>Modulator Delay</td>
<td>SENSE:PUlS:e:HWDelay:MODulator</td>
<td>ModulatorDelay</td>
</tr>
<tr>
<td>Fixed ADC Delay</td>
<td>SENSE:PUlS:e:HWDelay:ADC?</td>
<td>FixedADCDelay</td>
</tr>
<tr>
<td>Synchronize ADCs using pulse trigger</td>
<td>SENSE:PATH:CONFig:ELEMENT[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td>Pulse4 Output Indicates ADC Activity</td>
<td>SENSE:PUlS:e:OPTion</td>
<td>Pulse4OutAsADCActivity</td>
</tr>
<tr>
<td>Pulse Trigger...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trigger Source</td>
<td>SENSE:PATH:CONFig:ELEMENT[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td>Trigger Level/Edge</td>
<td>SENSE:PUlS:e:TTYPe</td>
<td>TriggerInType</td>
</tr>
<tr>
<td>Synchronize ADCs using pulse trigger</td>
<td>SENSe:PULSe[:STATe]</td>
<td>State</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>----------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Trigger...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DC Source...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Global Source</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Segment Table Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Segment</td>
<td>SENSe:SEGment:ADD</td>
<td>Add</td>
</tr>
<tr>
<td>Insert</td>
<td>Segment</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Delete</td>
<td>Segment</td>
<td>SENSe:SEGment:DELete</td>
<td>Remove</td>
</tr>
<tr>
<td>Delete All</td>
<td>Segments</td>
<td>SENSe:SEGment:DELete:ALL</td>
<td>None</td>
</tr>
<tr>
<td>X-Axis Point Spacing</td>
<td>SENSe:SEGment:X:SPACing</td>
<td>XAxisPointSpacing</td>
<td></td>
</tr>
<tr>
<td>Segment Table...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td></td>
</tr>
<tr>
<td>Allow Arbitrary Segments</td>
<td>SENSE:SEGMENT:ARBitrary</td>
<td>AllowArbitrarySegments</td>
<td></td>
</tr>
<tr>
<td>Display Center/Span Freq</td>
<td>SENSE:SEGMENT:FREQuency:CENTer</td>
<td>CenterFrequency</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:SEGMENT:FREQuency:SPAN</td>
<td>FrequencySpan</td>
<td></td>
</tr>
<tr>
<td>Save Table</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Load Table</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Independent Settings Per Segment</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vector Averaging</td>
<td>SENSE:SEGMENT:SA:VAverage</td>
<td>SAVectorAverage</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:SEGMENT:SA:VAverage:CONTrol</td>
<td>SAVectorAverageOption</td>
<td></td>
</tr>
<tr>
<td>Video Bandwidth</td>
<td>SENSE:SEGMENT:SA:VIDeobw</td>
<td>SAVideoBandwidth</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:SEGMENT:SA:VIDeobw:CONTrol</td>
<td>SAVideoAverageOption</td>
<td></td>
</tr>
<tr>
<td>Reference Tone</td>
<td>SENSE:SEGMENT:SA:MTReference</td>
<td>SAMTReference</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:SEGMENT:SA:MTReference:CONTrol</td>
<td>SAMTReferenceFreqOption</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:SEGMENT:SA:MTReference:MAX?</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:SEGMENT:SA:MTReference:MIN?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SA Data Threshold</td>
<td>SENSE:SEGMENT:SA:DTHReshold</td>
<td>SADataThreshold</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:SEGMENT:SA:DTHReshold:CONTrol</td>
<td>SADataThresholdOption</td>
<td></td>
</tr>
<tr>
<td>Show Table</td>
<td>Auto</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

7769
<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tone</td>
<td>Spacing</td>
<td>SENSE:SA:IMAGE:COHerence:SPACing</td>
<td>MultiToneImageRejectSpacing</td>
</tr>
<tr>
<td>Reference Tone</td>
<td></td>
<td>SENSE:SA:IMAGE:COHerence:REFERENCE</td>
<td>MultiToneImageRejectReference</td>
</tr>
<tr>
<td>Data Display</td>
<td>Show All</td>
<td>SENSE:SA:IMAGE:COHerence:DATA</td>
<td>MultiToneImageRejectDataDisplay</td>
</tr>
<tr>
<td></td>
<td>Zero the Non-Tones</td>
<td>SENSE:SA:IMAGE:COHerence:DATA</td>
<td>MultiToneImageRejectDataDisplay</td>
</tr>
<tr>
<td>Detector Bypass</td>
<td>On/Off</td>
<td>SENSE:SA:DETEctor:BYPass[:STATE]</td>
<td>EnableDetectorBypass</td>
</tr>
<tr>
<td></td>
<td>Reject up to harmonic</td>
<td>SENSE:SA:IMAGE:COHerence:HREJect</td>
<td>MultiToneImageRejectHarmonic</td>
</tr>
</tbody>
</table>
CF_Sweep Commands - SMC
Sweep Commands - Scalar Mixer/Converter + Phase Measurement Class

Click [here](#) to view links to Sweep commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
<td><strong>Sub-item</strong></td>
<td><strong>SCPI</strong></td>
<td><strong>COM</strong></td>
</tr>
<tr>
<td>Number of Points</td>
<td></td>
<td>SENSE:SWEep:POINts</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>Sweep Type</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X-Axis Type</td>
<td></td>
<td>CALCulate:MEASure:MIXer:XAXis</td>
<td>ActiveXAxisRange</td>
</tr>
<tr>
<td>SMC Setup...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sweep Timing Tab Commands</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
<td><strong>Sub-item</strong></td>
<td><strong>SCPI</strong></td>
<td><strong>COM</strong></td>
</tr>
<tr>
<td>Sweep Time</td>
<td>Auto</td>
<td>SENSE:SWEep:TIME:AUTO</td>
<td>chan.SweepTime</td>
</tr>
<tr>
<td>Dwell Time</td>
<td></td>
<td>SENSE:SWEep:DWELI</td>
<td>DwellTime</td>
</tr>
<tr>
<td>Source Control Tab Commands</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>Pulse Setup...</td>
<td>Pulse Measurement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td>SENSE:SWEep:PULSE:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Standard Pulse</td>
<td>SENSE:SWEep:PULSE:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Pulse Profile</td>
<td>SENSE:SWEep:PULSE:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Pulse Timing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Width</td>
<td>SENSE:SWEep:PULSE:PRIMary:WIDth</td>
<td>PrimaryWidth</td>
<td></td>
</tr>
<tr>
<td>Pulse Period</td>
<td>SENSE:SWEep:PULSE:PRIMary:PERiod</td>
<td>PrimaryPeriod</td>
<td></td>
</tr>
<tr>
<td>Pulse Frequency</td>
<td>SENSE:SWEep:PULSE:PRIMary:FREQuency</td>
<td>PrimaryFrequency</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Command</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>---------------------------</td>
<td>--------------------------------</td>
<td></td>
</tr>
<tr>
<td>Embedded LO</td>
<td>Enable Embedded LO</td>
<td>SENSe:MIXer:LO:FREQuency:MODE</td>
<td></td>
</tr>
<tr>
<td>Tuning Method</td>
<td>Tuning Method</td>
<td>SENSe:MIXer:ELO:TUNing:MODE</td>
<td></td>
</tr>
<tr>
<td>Tuning Point</td>
<td>Tune every</td>
<td>SENSe:MIXer:ELO:TUNing:INTerval</td>
<td></td>
</tr>
<tr>
<td>Broadband Search</td>
<td>Broadband Search</td>
<td>SENSe:MIXer:ELO:TUNing:SPAN</td>
<td></td>
</tr>
<tr>
<td>IFBW</td>
<td>IFBW</td>
<td>SENSe:MIXer:ELO:TUNing:IFBW</td>
<td></td>
</tr>
<tr>
<td>Max Iterations</td>
<td>Max Iterations</td>
<td>SENSe:MIXer:ELO:TUNing:ITERations</td>
<td></td>
</tr>
<tr>
<td>Tolerance</td>
<td>Tolerance</td>
<td>SENSe:MIXer:ELO:TUNing:TOLerance</td>
<td></td>
</tr>
<tr>
<td>LO Frequency Delta</td>
<td>LO Frequency Delta</td>
<td>SENSe:MIXer:ELO:LO:DELTA</td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>Default</td>
<td>SENSe:MIXer:ELO:RESet</td>
<td></td>
</tr>
</tbody>
</table>
## CF_Sweep Commands - Standard

Sweep Commands - Standard Measurement Class

Click [here](#) to view links to Sweep commands for all Measurement Classes.

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td></td>
<td>SENSE:SWEep:POINts</td>
<td>NumberOfPoints</td>
</tr>
<tr>
<td>Sweep Type</td>
<td>Linear Frequency</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Log Frequency</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Power Sweep</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>CW Time</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Segment Sweep</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Phase Sweep</td>
<td>SENSE:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Start</td>
<td></td>
<td>SENSE:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop</td>
<td></td>
<td>SENSE:FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>X-axis Type</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:FOM:DISPlay:SELect</td>
<td>DisplayRange</td>
<td></td>
</tr>
<tr>
<td><strong>Sweep Setup...</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Sweep Timing Tab Commands</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sweep Time</strong></td>
<td>Auto</td>
<td>SENSE:SWEep:TIME:AUTO</td>
<td>chan.SweepTime</td>
</tr>
<tr>
<td><strong>Dwell Time</strong></td>
<td></td>
<td>SENSE:SWEep:DWELI</td>
<td>DwellTime</td>
</tr>
<tr>
<td><strong>Sweep Delay</strong></td>
<td></td>
<td>SENSE:SWEep:DWELI:SDElay</td>
<td>SweepDelay</td>
</tr>
<tr>
<td><strong>Sweep Mode</strong></td>
<td>AUTO</td>
<td>SENSE:SWEep:GENeration</td>
<td>SweepGenerationMode</td>
</tr>
<tr>
<td></td>
<td>STEPPED</td>
<td>SENSE:SWEep:GENeration</td>
<td>SweepGenerationMode</td>
</tr>
<tr>
<td><strong>Sweep Sequence</strong></td>
<td>STD</td>
<td>SENSE:SWEep:GENeration:POINtsweep</td>
<td>PointSweepState</td>
</tr>
<tr>
<td></td>
<td>POINT</td>
<td>SENSE:SWEep:GENeration:POINtsweep</td>
<td>PointSweepState</td>
</tr>
</tbody>
</table>

7775
<table>
<thead>
<tr>
<th>Fast Sweep</th>
<th>ON</th>
<th>SENSE:SWEep:SPEed</th>
<th>SweepSpeedMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td>SENSE:SWEep:SPEed</td>
<td></td>
<td>SweepSpeedMode</td>
</tr>
</tbody>
</table>

**Source Control Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Frequency Offset</td>
<td>SENSE:FOM[:STATe]</td>
<td>State</td>
</tr>
<tr>
<td></td>
<td>(ON/OFF)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Mode - Coupled and Un-coupled</td>
<td>SENSE:FOM:RANGe:COUPled</td>
<td>Coupled</td>
</tr>
<tr>
<td></td>
<td>Sweep Type</td>
<td>SENSE:FOM:RANGe:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td></td>
<td>Settings</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Start</td>
<td>SENSE:FOM:RANGe:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td></td>
<td>Stop</td>
<td>SENSE:FOM:RANGe:FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td></td>
<td>Annotation - Primary, Source, Receivers, Source2, Source3</td>
<td>SENSE:FOM:RANGe:NAME?</td>
<td>Name</td>
</tr>
<tr>
<td>X-Axis Point Spacing</td>
<td>SENSE:FORM:RANGE:SEGMENT:SWEep:POINTS</td>
<td>NumberOfPoints</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------------------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td><strong>Pulse Setup</strong></td>
<td><strong>Pulse Measurement</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Off</td>
<td>SENSE:SWEep:PULSE:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Standard Pulse</td>
<td>SENSE:SWEep:PULSE:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>Pulse Profile</td>
<td>SENSE:SWEep:PULSE:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td><strong>Pulse Timing</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse Width</td>
<td>SENSE:SWEep:PULSE:PRIMARY:WIDTH</td>
<td>PrimaryWidth</td>
<td></td>
</tr>
<tr>
<td>Pulse Period</td>
<td>SENSE:SWEep:PULSE:PRIMARY:PERIOD</td>
<td>PrimaryPeriod</td>
<td></td>
</tr>
<tr>
<td>Pulse Frequency</td>
<td>SENSE:SWEep:PULSE:PRIMARY:FREQUENCY</td>
<td>PrimaryFrequency</td>
<td></td>
</tr>
<tr>
<td><strong>Properties</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>State/Value</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------------</td>
<td>---------------------------------------</td>
<td></td>
</tr>
<tr>
<td>SW Gating</td>
<td>SENSE:SWEP:pulse:swgate</td>
<td>SoftwareGateState</td>
<td></td>
</tr>
<tr>
<td>Autoselect IF Path Gain and Loss</td>
<td>SENSE:SWEP:pulse:ifgain[:AUTO]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>IF Path...</td>
<td>SENSE:PATH:CONFIG:ELEMENT[:STATE]</td>
<td>Element</td>
<td></td>
</tr>
<tr>
<td>Optimize Pulse Frequency</td>
<td>SENSE:SWEP:pulse:prf[:AUTO]</td>
<td>AutoOptimizePRF</td>
<td></td>
</tr>
<tr>
<td>Autoselect Profile Sweep Time</td>
<td>SENSE:SWEP:pulse:MODE</td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td>IFBW</td>
<td>SENSE:SWEP:pulse:ctime[:AUTO]</td>
<td>AutoIFBandWidth</td>
<td></td>
</tr>
<tr>
<td>Number of Points</td>
<td>SENSE:SWEP:POINTS</td>
<td>NumberOfPoints</td>
<td></td>
</tr>
<tr>
<td>Measurement Timing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Width</strong></td>
<td><code>SENSe:SWEep:PULSe:TIMing</code></td>
<td>AutoPulseTiming</td>
<td></td>
</tr>
<tr>
<td><strong>Delay</strong></td>
<td><code>SENSe:SWEep:PULSe:TIMing</code></td>
<td>AutoPulseTiming</td>
<td></td>
</tr>
<tr>
<td><strong>Pulse Gen</strong></td>
<td><code>SENSe:SWEep:PULSe:MODE</code></td>
<td>PulseMeasMode</td>
<td></td>
</tr>
<tr>
<td><strong>Master Pulse Trigger</strong></td>
<td><code>SENSe:PATH:CONFig:ELEMent[:STATE]</code></td>
<td>Element</td>
<td></td>
</tr>
<tr>
<td><strong>Primary Clock</strong></td>
<td><code>SENSe:SWEep:PULSe:PRIMary:CLOCk</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>Autoselect Width &amp; Delay</strong></td>
<td><code>SENSe:SWEep:PULSe:TIMing</code></td>
<td>AutoPulseTiming</td>
<td></td>
</tr>
<tr>
<td><strong>Autoselect Pulse Generators</strong></td>
<td><code>SENSe:SWEep:PULSe:DRIVe[:AUTO]</code></td>
<td>AutoSelectPulseGen</td>
<td></td>
</tr>
</tbody>
</table>

### Pulse Generators...

<p>| <strong>Width</strong> | <code>SENSe:PULSe:WIDTh</code> | Width |
| <strong>Delay</strong> | <code>SENSe:PULSe:DELaY</code> | Delay |</p>
<table>
<thead>
<tr>
<th>Invert</th>
<th>SENSe:PULSe:INVVert</th>
<th>Invert</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable</td>
<td>SENSe:PULSe[:STATe]</td>
<td>State</td>
</tr>
<tr>
<td>Trigger</td>
<td>SENSe:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td>Frequency</td>
<td>SENSe:SWEep:PULSe:MASTer:FREQuency</td>
<td>MasterFrequency</td>
</tr>
<tr>
<td>Period</td>
<td>SENSe:SWEep:PULSe:MASTer:PERiod</td>
<td>MasterPeriod</td>
</tr>
<tr>
<td>Enable Source x Modulator</td>
<td>SENSe:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td>Modulator Drive</td>
<td>SENSe:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
</tr>
<tr>
<td>Offset Pulse using Modulator and ADC Delays</td>
<td>SENSe:PULSe:HWDelay[:STATe]</td>
<td>EnableOffsetDelays</td>
</tr>
<tr>
<td>Modulator Delay</td>
<td>SENSe:PULSe:HWDelay:MODulator</td>
<td>ModulatorDelay</td>
</tr>
<tr>
<td><strong>Fixed ADC Delay</strong></td>
<td>SENSE:PULSE:HWDelay:ADC?</td>
<td>FixedADCDelay</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td><strong>Synchronize ADCs using pulse trigger</strong></td>
<td>SENSE:PATH:CONFig:ELEMent[:STATE]</td>
<td>Element</td>
</tr>
<tr>
<td><strong>Pulse4 Output</strong></td>
<td>SENSE:PULSE4:OPTion</td>
<td>Pulse4OutAsADCActivity</td>
</tr>
<tr>
<td><strong>Indicates</strong></td>
<td>SENSE:PULSE4:OPTion:MODE ALL</td>
<td>None</td>
</tr>
<tr>
<td><strong>All ADC Activity</strong></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>Trace ADC Activity</strong></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>Pulse Trigger...</strong></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>Trigger Source</strong></td>
<td>SENSE:PATH:CONFig:ELEMent[:STATE]</td>
<td>Element</td>
</tr>
<tr>
<td><strong>Trigger Level/Edge</strong></td>
<td>SENSE:PULSE:TTYPe</td>
<td>TriggerInType</td>
</tr>
<tr>
<td>Synchronize ADCs using pulse trigger</td>
<td>SENS:PULSe[:STATe]</td>
<td>State</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>---------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Trigger...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Balanced Source...</td>
<td>Topology</td>
<td></td>
</tr>
<tr>
<td>BAL</td>
<td>CALCulate:FSIMulator:BALun:TOPology:BALanced</td>
<td></td>
</tr>
<tr>
<td>BAL-BAL</td>
<td>CALCulate:FSIMulator:BALun:TOPology:BBALanced</td>
<td>SetBBPorts</td>
</tr>
<tr>
<td>BAL-SE</td>
<td>CALCulate:FSIMulator:BALun:TOPology:BSBalanced</td>
<td>SetBSBPorts</td>
</tr>
<tr>
<td>SE-BAL</td>
<td>CALCulate:FSIMulator:BALun:TOPology:SSBalanced</td>
<td>SetSSBPorts</td>
</tr>
<tr>
<td>Stimulus</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single Ended</td>
<td>CALCulate:FSIMulator:BALun:STIMulus:MODE</td>
<td>Mode</td>
</tr>
<tr>
<td>True Mode</td>
<td>CALCulate:FSIMulator:BALun:STIMulus:MODE</td>
<td>Mode</td>
</tr>
<tr>
<td>Forward True Mode</td>
<td>CALCulate:FSIMulator:BALun:STIMulus:MODE</td>
<td>Mode</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>Reverse True Mode</td>
<td>CALCulate:FSIMulator:BALun:STIMulus:MODE</td>
<td>Mode</td>
</tr>
<tr>
<td>Source Only Mode</td>
<td>CALCulate:FSIMulator:BALun:STIMulus:MODE</td>
<td>Mode</td>
</tr>
<tr>
<td><strong>Balanced Port Offset</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phase Offset</td>
<td>CALCulate:FSIMulator:BALun:BPORT:OFFSet:PHASe</td>
<td>BalPort1PhaseOffset</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BalPort2PhaseOffset</td>
</tr>
<tr>
<td>Offset as fixture</td>
<td>CALCulate:FSIMulator:BALun:FIXTure:OFFSet:PHASe</td>
<td>PhaseAsFixture</td>
</tr>
<tr>
<td>Power Offset</td>
<td>CALCulate:FSIMulator:BALun:BPORT:OFFSet:POWer</td>
<td>BalPort1PowerOffset</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BalPort2PowerOffset</td>
</tr>
<tr>
<td>Offset as fixture</td>
<td>CALCulate:FSIMulator:BALun:FIXTure:OFFSet:POWer</td>
<td>PowerAsFixture</td>
</tr>
<tr>
<td><strong>Phase Sweep (CW Time Only)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable Phase Sweep</td>
<td>CALCulate:FSIMulator:BALun:PHASe:SWEep:STATe</td>
<td>PhaseSwpState</td>
</tr>
<tr>
<td>Sweep Phase On</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Start Phase</td>
<td>CALCulate:FSIMulator:BALun:BPORT:SWEep:PHASe:STARt</td>
<td>BalPort1StartPhase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BalPort2StartPhase</td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>Stop Phase</td>
<td>CALCulate:FSIMulator:BA lun:BPORt:SW Eep:PHAse:STOP</td>
<td>BalPort1StopPhase BalPort2StopPhase</td>
</tr>
<tr>
<td>Offset as fixture</td>
<td>CALCulate:FSIMulator:BA lun:FIXTure:PHASe</td>
<td>PhaseSwpAsFixture</td>
</tr>
<tr>
<td>Phase Control</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sweep Type</td>
<td>SENSe:SW Eep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Phase</td>
<td>SOURce:PH ASe[:FIXed]</td>
<td>FixedPhase</td>
</tr>
<tr>
<td>Start Phase</td>
<td>SOURce:PH Ase:STARt</td>
<td>StartPhase</td>
</tr>
<tr>
<td>Stop Phase</td>
<td>SOURce:PH Ase:STOP</td>
<td>StopPhase</td>
</tr>
<tr>
<td>Phase Control</td>
<td>SOURce:PH Ase:MODE:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SOURce:PH Ase:MODE[:VALue]</td>
<td>PhaseControlMode</td>
</tr>
<tr>
<td>Reference Port</td>
<td>SOURce:PH Ase:REFerence:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SOURce:PH Ase:REFerence:PORT</td>
<td>PhaseReferencePort</td>
</tr>
<tr>
<td>Ext. Source Port</td>
<td>SOURce:PH Ase:EX Ternal:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SOURce:PH Ase:EX Ternal:PORT</td>
<td>None</td>
</tr>
<tr>
<td>Control Parameter</td>
<td>SOURce:PH Ase:PAR a meter:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SOURce:PH Ase:PAR a meter[:VALue]</td>
<td>PhaseParameter</td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Phase Leveling Tolerance</td>
<td>SOURce:PHASE:CONTro1:TOlerance</td>
<td>PhaseTolerance</td>
</tr>
<tr>
<td>Leveling Max Iterations</td>
<td>SOURce:PHASE:CONTro1:ITERation</td>
<td>PhaseIterationNumber</td>
</tr>
<tr>
<td>Power Leveling Tolerance</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:TOlerance</td>
<td>Tolerance</td>
</tr>
<tr>
<td>Leveling IFBW</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:IFBW</td>
<td>LevelingIFBW</td>
</tr>
<tr>
<td>Start Power</td>
<td>SOURce:POWer:PORT:STARt</td>
<td>StartPower</td>
</tr>
<tr>
<td>Stop Power</td>
<td>SOURce:POWer:PORT:STOP</td>
<td>StopPower</td>
</tr>
<tr>
<td>Receiver Leveling...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phase Control Setup Dialog</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Setup...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Referenced to</td>
<td>SOURce:PHASE:PARameter:PORT</td>
<td>PhaseReferencePort</td>
</tr>
<tr>
<td>Control Parameter</td>
<td>SOURce:PHAse:PARameter</td>
<td>PhaseParameter</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Background Sweep Properties</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Apply Settings to All Ports</td>
<td>SOURce:PHAse:PARameter:MODE</td>
<td>PhaseControlMode</td>
</tr>
<tr>
<td>Use Leveling IFBW</td>
<td>SOURce:POWer:ALC[:MODE]:RECeiver:IFBW</td>
<td>LevelingIFBW</td>
</tr>
<tr>
<td>Tolerance</td>
<td>SOURce:PHASe:CONTrol:TOLerance</td>
<td>PhaseTolerance</td>
</tr>
<tr>
<td>Max Iterations</td>
<td>SOURce:PHASe:CONTrol:ITERation</td>
<td>PhasIterationNumber</td>
</tr>
<tr>
<td>DC Source... Outputs</td>
<td>SOURce:DC:ENABle</td>
<td>EnableAllOutput</td>
</tr>
<tr>
<td>State</td>
<td>SOURce:DC:STATe</td>
<td>State</td>
</tr>
<tr>
<td>Start DC</td>
<td>SOURce:DC:STARt</td>
<td>Start</td>
</tr>
<tr>
<td>Stop DC</td>
<td>SOURce:DC:STOP</td>
<td>Stop</td>
</tr>
<tr>
<td>Limits...</td>
<td>State</td>
<td>Min</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>Shift LO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LF Extension</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Global Source</td>
<td>Power On (All Channels)</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SYSTem:PREFerences:SOURce:GLOBAL:POFF:IGNORE[:STATE]</td>
<td></td>
</tr>
<tr>
<td>Global Sources</td>
<td>Ignore &quot;Power ON&quot; Setting</td>
<td>SYSTem:PREFerences:SOURce:GLOBAL[:STATE]</td>
</tr>
<tr>
<td>State</td>
<td>SYSTem:PREFerences:SOURce:GLOBAL:OUTPUT[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Frequency</td>
<td>SYSTem:PREFerences:SOURce:GLOBal:FREQuency</td>
<td>None</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Power</td>
<td>SYSTem:PREFerences:SOURce:GLOBal:POWer</td>
<td>None</td>
</tr>
</tbody>
</table>

### Segment Table Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Sub-Add Item</td>
<td>SENSE:SEGMENT:ADD</td>
<td>Add</td>
</tr>
<tr>
<td>Insert</td>
<td>Segment</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Delete</td>
<td>Segment</td>
<td>SENSE:SEGMENT:DELETE</td>
<td>Remove</td>
</tr>
<tr>
<td>Delete All Segments</td>
<td>SENSE:SEGMENT:DELETE:ALL</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Segment Table...</td>
<td>X-Axis Point Spacing</td>
<td>SENSE:SEGMENT:X:SPACING</td>
<td>XAxisPointSpacing</td>
</tr>
<tr>
<td></td>
<td>Allow Arbitrary Segments</td>
<td>SENSE:SEGMENT:ARBitrary</td>
<td>AllowArbitrarySegments</td>
</tr>
<tr>
<td>Feature</td>
<td>Command Description</td>
<td>Setting Type</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------------------------------------------------------------</td>
<td>----------------------</td>
<td></td>
</tr>
<tr>
<td>Display</td>
<td>SENSe:SEGment:FREQuency:CENTer SENSe:SEGment:FREQuency:SPAN</td>
<td>CenterFrequency</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>FrequencySpan</td>
<td></td>
</tr>
<tr>
<td>Center/Span</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Freq</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Save Table</td>
<td>None</td>
<td>ExportCSVfile</td>
<td></td>
</tr>
<tr>
<td>Load Table</td>
<td>None</td>
<td>ImportCSVfile</td>
<td></td>
</tr>
<tr>
<td>Independent Settings Per Segment</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power Level</td>
<td>SENSe:SEGment:POWer[:LEVel]</td>
<td>TestPortPower</td>
<td></td>
</tr>
<tr>
<td>IF Bandwidth</td>
<td>SENSe:SEGment:BWIDth[:RESolution]</td>
<td>IFBBandwidth</td>
<td></td>
</tr>
<tr>
<td>IF Bandwidth Per Port</td>
<td>SENSe:SEGment:BWIDth</td>
<td>BANDwidth:PORT[:RESolution]:CONTrol</td>
<td>PortIFBBandwidthOption PortIFBBandwidth</td>
</tr>
<tr>
<td>Sweep Time</td>
<td>SENSe:SEGment:SWEep:TIME</td>
<td>SweepTimeOption</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SweepTime</td>
<td></td>
</tr>
<tr>
<td>Dwell Time</td>
<td>SENSe:SWEep:DWELI</td>
<td>DwellTime</td>
<td></td>
</tr>
<tr>
<td>Delay</td>
<td>SENSe:SEGment:SWEep:DELay</td>
<td>DelayOption</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Delay</td>
<td></td>
</tr>
<tr>
<td>Sweep Mode</td>
<td>SENSe:SEGment:SWEep:GENeration:CONTrol</td>
<td>SweepModeOption</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SweepMode</td>
<td></td>
</tr>
<tr>
<td>Shift LO</td>
<td>SENSE:SEGMENT:SHLO</td>
<td>ShiftLO</td>
<td></td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------------------</td>
<td>--------------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:SEGMENT:SHLO:CONTrol</td>
<td>ShiftLOOption</td>
<td></td>
</tr>
<tr>
<td>Receiver Atten Per Port</td>
<td>SENSE:SEGMENT:POWer:ATTenuation:RECeiver:REFERENCE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:SEGMENT:POWer:ATTenuation:RECeiver:CONTrol</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Show Table</td>
<td>Auto</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>On</td>
<td>DISPlay:WINDow:TABLE</td>
<td>ShowTable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

7790
## CF_Sweep Commands - VMC

**Sweep Commands - Vector Mixer/Converter**

Click [here](#) to view links to Sweep commands for all Measurement Classes.

<table>
<thead>
<tr>
<th>Main Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td>Number of Points</td>
</tr>
<tr>
<td>Sweep Type</td>
</tr>
<tr>
<td>X-Axis Type</td>
</tr>
<tr>
<td>VMC Setup...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sweep Timing Tab Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td>Sweep Time</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Source Control Tab Commands</td>
</tr>
<tr>
<td>----------------------------</td>
</tr>
<tr>
<td><strong>Softkey</strong></td>
</tr>
<tr>
<td>Pulse Setup...</td>
</tr>
<tr>
<td>Off</td>
</tr>
<tr>
<td>Standard Pulse</td>
</tr>
<tr>
<td>Pulse Profile</td>
</tr>
<tr>
<td>Pulse Timing</td>
</tr>
<tr>
<td>Pulse Width</td>
</tr>
<tr>
<td>Pulse Period</td>
</tr>
<tr>
<td>Pulse Frequency</td>
</tr>
<tr>
<td>Embedded LO...</td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>Tuning Point</td>
</tr>
<tr>
<td>LO Frequency Delta</td>
</tr>
<tr>
<td>Tuning Settings</td>
</tr>
<tr>
<td>Broadband and precise</td>
</tr>
<tr>
<td>Precise only</td>
</tr>
<tr>
<td>Disable tuning</td>
</tr>
<tr>
<td>Sweep Span</td>
</tr>
<tr>
<td>Max Iterations</td>
</tr>
<tr>
<td>Tolerance</td>
</tr>
<tr>
<td>Tuning IFBW</td>
</tr>
<tr>
<td>Tune every</td>
</tr>
</tbody>
</table>

7793
## CF_System Commands

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show Taskbar</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Move App to Back</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Minimize Application</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Exit</td>
<td>None</td>
<td>app.Quit</td>
<td></td>
</tr>
<tr>
<td>Security...</td>
<td>SYSTem:SECurity[:LEVel]</td>
<td>app.SecurityLevel</td>
<td></td>
</tr>
<tr>
<td>Control Panel...</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Manage Files...</td>
<td>List Files</td>
<td>MMEMory:CATalog</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Copy Files</td>
<td>MMEMory:COPY</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Move Files</td>
<td>MMEMory:MOVE</td>
<td>None</td>
</tr>
<tr>
<td>Softkey</td>
<td>Sub-item</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>------</td>
<td>-----</td>
</tr>
<tr>
<td>Next/Prev Keys</td>
<td>Trace</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Channel</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sheet</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Cal: Always use Internal trigger during cal</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Cal: ECAL Extrapolation for IMD</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Cal: For Frequency Offset, use Primary frequencies</td>
<td>SENSE:CORRection:PREFerences:CALibration[:FOM]:RANGe</td>
<td>FrequencyOffsetRangeForCalCompu tations</td>
<td></td>
</tr>
<tr>
<td>Cal: (SCPI only) Auto-generate a User Cal Set</td>
<td>SENSE:CORRection:CSET:CREate</td>
<td>calMgr.CreateCalSet</td>
<td></td>
</tr>
<tr>
<td>Shunt-L fixtures</td>
<td>Display: Selected trace changes width briefly</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>Display: Selected trace is wider</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Display: Touchscreen On</td>
<td>SYSTem:TOUCHscreen[:STATe]</td>
<td>Touchscreen</td>
<td></td>
</tr>
<tr>
<td>Ext Device: De-activate on Preset and Recall</td>
<td>SYSTem:PREFerences:ITEM:EDEV:DPOLicy</td>
<td>ExternalDeviceDeActivatePolicy</td>
<td></td>
</tr>
<tr>
<td>Limit: Test the nearest measurement point</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>Marker: Coupling controls on/off state of markers</td>
<td>SYSTem:PREFerences:ITEM:MCControl</td>
<td>MarkCoupControlsMkrState</td>
<td></td>
</tr>
<tr>
<td>Marker: On Preset, Coupled Markers is On</td>
<td>SYSTem:PREFerences:ITEM:MCPreset</td>
<td>MarkCoupControlsMkrState</td>
<td></td>
</tr>
<tr>
<td>Marker: On Preset, Coupling Method is Channel</td>
<td>SYSTem:PREFerences:ITEM:MCMethod</td>
<td>MarkCoupMethPresetIsChan</td>
<td></td>
</tr>
<tr>
<td>Marker: Programmin</td>
<td>SYSTem:PREFerences:ITEM:REFMarker</td>
<td>TreatMkr10AsReference</td>
<td></td>
</tr>
<tr>
<td>g treats Mkr 10 as Reference</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>Marker: Use single marker for marker search</td>
<td>SYSTem:PREFerences:ITEM:MARKer:SINGle</td>
<td>SingleMarkerSearch</td>
<td></td>
</tr>
<tr>
<td>Meas: Mathematica l offset for receiver attenuation</td>
<td>SYSTem:PREFerences:ITEM:OFFSet:RCV</td>
<td>OffsetReceiverAttenuator</td>
<td></td>
</tr>
<tr>
<td>Meas: Mathematica l offset for source attenuation</td>
<td>SYSTem:PREFerences:ITEM:OFFSet:SRC</td>
<td>OffsetSourceAttenuator</td>
<td></td>
</tr>
<tr>
<td>Memory: Data Math 8510 Mode</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Memory: Interpolate ON is default condition</td>
<td>SYSTem:PREFerences:ITEM:MIINTERpolate</td>
<td>InterpolateMemoryIsDefault</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>Preset turn power on</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power: Report source unleveled events as errors</td>
<td>SYSTem:ERRor:REPort:SUNLeveled</td>
<td>EnableSourceUnleveledEvents</td>
<td></td>
</tr>
<tr>
<td>Power: Report when receiver is overloaded</td>
<td>SYSTem:PREFerences:ITEM:RECeivers:CERRor</td>
<td>ReportReceiverOverload</td>
<td></td>
</tr>
<tr>
<td>Power: Force RF power Off at the end of sweep</td>
<td>SYSTem:PREFerences:ITEM:RETRace:POWer</td>
<td>PowerOnDuringRetraceMode</td>
<td></td>
</tr>
<tr>
<td>Power: Turn Source Power Off when receiver is overloaded</td>
<td>SYSTem:PREFerences:ITEM:RECeivers:OVERload:POWer</td>
<td>RFOffOnReceiverOverload</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Setting</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------</td>
<td>--------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>Sweep retrace</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>show Keys toolbar</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recall:</td>
<td>Softkey order is most recently used</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Scale:</td>
<td>On Preset</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Couple</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Scale to Window</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sweep:</td>
<td>On Preset set</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Setting</td>
<td>Command</td>
<td>Value</td>
<td></td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>--------------------------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>Sweep Mode to Stepped</td>
<td>SYSTem:PREFerences:ITEM:ASMRamp</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sweep: Use only ramp sweeps for Auto Sweep Mode</td>
<td>SYSTem:BEEP:STATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>System: Enable sound</td>
<td>SYSTem:BEEP:STATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>System: On Power-on show dialog if detect mm testset</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>System: Use keyboard to navigate softkeys</td>
<td>SYSTem:PREFerences:ITEM:SOFTkeys:NAVigation</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>System: Use parallel processing</td>
<td>SYSTem:PREFerences:ITEM:CORRection:PARallel:PROCess</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>System: Set front panel remote state when a SCPI command is received</td>
<td>SYSTem:PREFerences:ITEM:REMote:AUTO[:STATe]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Trigger: External Trigger Out is Global</td>
<td>TRIGger:PREFerences:AIGlobal</td>
<td>AuxTriggerScopeIsGlobal</td>
<td></td>
</tr>
<tr>
<td>Ext Reference: Modify Settings on Preset and Recall</td>
<td>SYSTem:PREFerences:ITEM:ROSCillator:RECall</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Data Saves...</td>
<td>MMEMory:STORe:DATA</td>
<td>SaveData</td>
<td></td>
</tr>
<tr>
<td>Power Limit...</td>
<td>SYSTem:POWer:LIMit</td>
<td>Limit</td>
<td></td>
</tr>
<tr>
<td>Transparency...</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------</td>
<td>-------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>User Preset...</strong></td>
<td>SYSTem:UPReset</td>
<td>UserPreset</td>
<td></td>
</tr>
<tr>
<td><strong>Page Setup...</strong></td>
<td>Hardcopy</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>Disp Colors...</strong></td>
<td>See Display</td>
<td>See Display</td>
<td></td>
</tr>
<tr>
<td><strong>Print Colors...</strong></td>
<td>See Display</td>
<td>See Display</td>
<td></td>
</tr>
<tr>
<td><strong>Global Sources...</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Power On (All Channels)</strong></td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>Global Sources Ignore &quot;Power ON&quot; Setting</strong></td>
<td>SYSTem:PREFERENCES:SOURce:GLOBAL:POFF:IGNORE[:STATE]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>Global</strong></td>
<td>SYSTem:PREFERENCES:SOURce:GLOBAL[:STATE]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>State</strong></td>
<td>SYSTem:PREFERENCES:SOURce:GLOBAL:OUTPUT[:STATE]</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td><strong>Frequency</strong></td>
<td>SYSTem:PREFERENCES:SOURce:GLOBAL:FREQuency</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Setting</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------</td>
<td>---------------------------</td>
<td></td>
</tr>
<tr>
<td>Power</td>
<td>SYSTEM:PREFERENCES:SOURce:GLOBAL:POWER</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sound</td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Remote Interface...</td>
<td>GPIB Address</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>app.GPIBAddress</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SICL Enabled</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>app.SICL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SICL Address</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>app.SICLAddress</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Automatically Enable on Startup</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sockets Enabled</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Telnet Enabled</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HiSLIP Enabled</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>HiSLIPPort</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HiSLIP Address</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>HiSLIPAddress</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command/Argument</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>Enable Remote Drive Access</td>
<td>SYSTem:COMMunicate:DRIVE:ENABle?</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Show SCPI Parser Console</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Monitor GPIB Bus</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>LAN Status...</td>
<td>LAN Reset</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Code Emulation</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Print Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Print...</td>
<td></td>
<td>HCOPy:DPRinter</td>
<td>DoPrint</td>
</tr>
<tr>
<td>Print to File...</td>
<td></td>
<td>HCOPy:FILE</td>
<td>PrintToFile</td>
</tr>
<tr>
<td>Page Setup...</td>
<td></td>
<td>Hardcopy</td>
<td>None</td>
</tr>
<tr>
<td>Print Colors...</td>
<td>Pen</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>-----</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Background</strong></td>
<td>DISPlay:COLor:BACKground</td>
<td>Background</td>
<td></td>
</tr>
<tr>
<td><strong>Active Background</strong></td>
<td>DISPlay:COLor:ABACkground</td>
<td>ActiveBackground</td>
<td></td>
</tr>
<tr>
<td><strong>Grid</strong></td>
<td>DISPlay:COLor:GRAT2</td>
<td>Grid</td>
<td></td>
</tr>
<tr>
<td><strong>Active Labels, Grid frame</strong></td>
<td>DISPlay:COLor:GRAT1</td>
<td>ActiveLabels</td>
<td></td>
</tr>
<tr>
<td><strong>Inactive Window Labels</strong></td>
<td>DISPlay:COLor:ILABel</td>
<td>InactiveLabels</td>
<td></td>
</tr>
<tr>
<td><strong>Failed Trace</strong></td>
<td>DISPlay:COLor:LIM1</td>
<td>FailedTraces</td>
<td></td>
</tr>
<tr>
<td><strong>N Trace: Data and Limits</strong></td>
<td>DISPlay:COLor:TRACe:DATA</td>
<td>DataAndLimits</td>
<td></td>
</tr>
<tr>
<td><strong>N Trace: Memory</strong></td>
<td>DISPlay:COLor:TRACe:MEMory</td>
<td>Memory</td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------------</td>
<td>---------------------</td>
<td></td>
</tr>
<tr>
<td>N Trace: Markers</td>
<td>DISPlay:COLor:TRACe:MARKer</td>
<td>Markers</td>
<td></td>
</tr>
<tr>
<td>N Trace: Memory Markers</td>
<td>DISPlay:COLor:TRACe:MMARker</td>
<td>MemoryMarkers</td>
<td></td>
</tr>
<tr>
<td>Change Color...</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Reset Color</td>
<td>DISPlay:COLor:RESet</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Color Theme</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Save Theme...</td>
<td>DISPlay:COLor:STORE</td>
<td>StoreTheme</td>
<td></td>
</tr>
<tr>
<td>Recall Theme...</td>
<td>DISPlay:COLor:LOAD</td>
<td>LoadTheme</td>
<td></td>
</tr>
<tr>
<td>Reset Theme</td>
<td>DISPlay:COLor:RESet</td>
<td>ResetTheme</td>
<td></td>
</tr>
</tbody>
</table>

**Help Tab Commands**

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
</table>

7808
<table>
<thead>
<tr>
<th>NA Help...</th>
<th>None</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>On The Web...</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Error Display...</td>
<td>Enable Messages</td>
<td>DISPLAY:ANNotation:MESSage:STATe</td>
</tr>
<tr>
<td>Status bar Display</td>
<td>DISPLAY:ANNotation[:STATus]</td>
<td>app.ShowStatusBar</td>
</tr>
<tr>
<td>Confirmation Dialog boxes</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>View Error Log...</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>About NA...</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Service Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update Firmware</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Category</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>--------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>Verification</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Adjustment Routines...</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Diagnostics</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Option Enable</td>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
## CF_Trace Commands

### Trace 1 - 7 Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trace 1</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>Trace 2</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>Trace 3</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>Trace 4</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>Trace 5</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>Trace 6</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>Trace 7</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>New Traces...</td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
</tr>
</tbody>
</table>

### Trace 8 - 15 Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trace 8</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>Trace 9</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>---------</td>
<td>--------</td>
<td>-------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Trace 10</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>Trace 11</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>Trace 12</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>Trace 13</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>Trace 14</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
<tr>
<td>Trace 15</td>
<td>On/Off</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
<td>View</td>
</tr>
</tbody>
</table>

### Trace Setup Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>TrN</td>
<td>DISPlay:WINDow:TRACe:SELect</td>
<td>View</td>
</tr>
<tr>
<td>Measure...</td>
<td></td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSParameterEx</td>
</tr>
<tr>
<td>Trace Title...</td>
<td>Title</td>
<td>DISPlay:WINDow:TRACe:TITLe:DATA</td>
<td>Title</td>
</tr>
<tr>
<td></td>
<td>Enable</td>
<td>DISPlay:WINDow:TRACe:TITLe[:STATe]</td>
<td>TraceTitleState</td>
</tr>
<tr>
<td>Add Trace</td>
<td>New Trace</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------------</td>
<td>-------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>New Trace + Channel</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>New Trace + Window</td>
<td>DISPLAY:WINDow:TRACe[:STATE]</td>
<td>View</td>
<td></td>
</tr>
<tr>
<td>New Trace + Channel + Window</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>New Traces...</td>
<td>S-Parameter Balanced Receivers</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSPParameterEx</td>
</tr>
<tr>
<td>Delete Trace</td>
<td>TrN</td>
<td>DISPLAY:WINDow:TRACe:DELete</td>
<td>None</td>
</tr>
<tr>
<td>Move Trace...</td>
<td></td>
<td>DISPLAY:WINDow:TRACe:MOVE</td>
<td>Move</td>
</tr>
<tr>
<td>Trace Manager...</td>
<td>Trace</td>
<td>DISPLAY:WINDow:TRACe[:STATE]</td>
<td>View</td>
</tr>
<tr>
<td></td>
<td>Meas</td>
<td>CALCulate:MEASure:PARameter</td>
<td>CreateSPParameterEx</td>
</tr>
<tr>
<td></td>
<td>Class</td>
<td>CALCulate:MEASure:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td></td>
<td>Channel</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Window</td>
<td>DISPLAY:WINDow[:STATE]</td>
<td>Add</td>
</tr>
<tr>
<td>Sheet</td>
<td>DISPlay:SHEet:STATE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>---------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>Trace Hold</td>
<td>Off</td>
<td>CALCulate:MEASure:HOLD:TYPE</td>
<td>TraceHoldType</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>CALCulate:MEASure:HOLD:TYPE</td>
<td>TraceHoldType</td>
</tr>
<tr>
<td></td>
<td>Min</td>
<td>CALCulate:MEASure:HOLD:TYPE</td>
<td>TraceHoldType</td>
</tr>
<tr>
<td></td>
<td>Restart</td>
<td>CALCulate:MEASure:HOLD:CLEar</td>
<td>TraceHoldClear</td>
</tr>
</tbody>
</table>
## CF_Trigger Commands

### Main Tab Commands

<table>
<thead>
<tr>
<th>Softkey</th>
<th>Sub-item</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hold</td>
<td></td>
<td>SENSE:SWEep:MODE</td>
<td>Hold</td>
</tr>
<tr>
<td>Single</td>
<td></td>
<td>SENSE:SWEep:MODE</td>
<td>chan.Single</td>
</tr>
<tr>
<td>Groups</td>
<td></td>
<td>SENSE:SWEep:MODE</td>
<td>chan.NumberOfGroups</td>
</tr>
<tr>
<td>Continuous</td>
<td></td>
<td>SENSE:SWEep:MODE</td>
<td>chan.Continuous</td>
</tr>
<tr>
<td>Restart</td>
<td></td>
<td>INITiate[:IMMediate]</td>
<td>ManualTrigger</td>
</tr>
<tr>
<td>Trigger Source</td>
<td>Internal</td>
<td>TRIGger[:SEQuence]:SOURce</td>
<td>Source</td>
</tr>
<tr>
<td></td>
<td>Manual</td>
<td>TRIGger[:SEQuence]:SOURce</td>
<td>Source</td>
</tr>
<tr>
<td></td>
<td>External</td>
<td>TRIGger[:SEQuence]:SOURce</td>
<td>Source</td>
</tr>
<tr>
<td>Setup</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trigger Source</td>
<td>TRIGger[:SEQUence]:SOURce</td>
<td>Setup.Source</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td>Trigger Scope</td>
<td>TRIGger[:SEQUence]:SCOPE</td>
<td>Setup.SCOPE</td>
<td></td>
</tr>
<tr>
<td>Channel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trigger State</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trigger...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trigger Mode</td>
<td>SENSE:SWEep:TRIGger:MODE</td>
<td>chan.TriggerMode</td>
<td></td>
</tr>
<tr>
<td>CHANel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SWEep</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POINT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRACe</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Continuous</td>
<td>SENSE:SWEep:MODE</td>
<td>chan.Continuous</td>
<td></td>
</tr>
<tr>
<td>Groups</td>
<td>SENSE:SWEep:GROups:COUNt</td>
<td>chan.NumberOfGroups</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSE:SWEep:MODE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single</td>
<td>SENSE:SWEep:MODE</td>
<td>chan.Single</td>
<td></td>
</tr>
<tr>
<td>Hold</td>
<td>SENSE:SWEep:MODE</td>
<td>Hold</td>
<td></td>
</tr>
<tr>
<td>Meas Trigger</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-----------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Global Trigger Delay</td>
<td>TRIGger:DELay</td>
<td>TriggerDelay</td>
<td></td>
</tr>
<tr>
<td>Meas Trig In BNC</td>
<td>TRIGger[:SEQUence]:ROUTE:INPut</td>
<td>ExternalTriggerConnectionBehavior</td>
<td></td>
</tr>
<tr>
<td>Handler I/O Pin 18</td>
<td>TRIGger[:SEQUence]:ROUTE:INPut</td>
<td>ExternalTriggerConnectionBehavior</td>
<td></td>
</tr>
<tr>
<td>Pulse3</td>
<td>TRIGger[:SEQUence]:ROUTE:INPut</td>
<td>ExternalTriggerConnectionBehavior</td>
<td></td>
</tr>
<tr>
<td>Rear SMB</td>
<td>TRIGger[:SEQUence]:ROUTE:INPut CONTrol:SIGNal:STReamline:RTRigger[:STATe]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Backplane</td>
<td>CONTrol:SIGNal:PXI:RTRigger[:STATe]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Level/Edge</td>
<td>TRIGger[:SEQUence]:TYPE</td>
<td>ExternalTriggerConnectionBehavior</td>
<td></td>
</tr>
<tr>
<td>Accept trigger before armed</td>
<td>CONTrol:SIGNal:TRIGger:ATBA</td>
<td>AcceptTriggerBeforeArmed</td>
<td></td>
</tr>
<tr>
<td>Meas Trig Ready</td>
<td>TRIGger:STATus:READY?</td>
<td>ReadyForTriggerStatus</td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Handler I/O Pin 21</td>
<td>TRIGger[:SEQUence]:ROUTE:READy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ready High</td>
<td>TRIGger:READY:POLarity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ready Low</td>
<td>TRIGger:READY:POLarity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aux Trig 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable</td>
<td>TRIGger:CHANnel:AUXiliary:ENABLE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Positive Pulse</td>
<td>TRIGger:CHANnel:AUXiliary:OPOLarity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Negative Pulse</td>
<td>TRIGger:CHANnel:AUXiliary:OPOLarity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Before Acquisition</td>
<td>TRIGger:CHANnel:AUXiliary:POSition</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>After Acquisition</td>
<td>Per Point</td>
<td>Rear SMB</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------</td>
<td>--------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>TRIGger:CHANnel:AUXiliary:POSition</td>
<td>TriggerOutPosition</td>
<td>TriggerOutInterval</td>
<td>HandshakeEnable</td>
</tr>
<tr>
<td>Aux Trig 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Enable</strong></td>
<td>TRIGger:CHANnel:AUXiliary:ENABle</td>
<td>Enable</td>
<td></td>
</tr>
<tr>
<td><strong>Positive Pulse</strong></td>
<td>TRIGger:CHANnel:AUXiliary:OPOLarity</td>
<td>TriggerOutPolarity</td>
<td></td>
</tr>
<tr>
<td><strong>Negative Pulse</strong></td>
<td>TRIGger:CHANnel:AUXiliary:OPOLarity</td>
<td>TriggerOutPolarity</td>
<td></td>
</tr>
<tr>
<td><strong>Before Acquisition</strong></td>
<td>TRIGger:CHANnel:AUXiliary:POSition</td>
<td>TriggerOutPosition</td>
<td></td>
</tr>
<tr>
<td><strong>After Acquisition</strong></td>
<td>TRIGger:CHANnel:AUXiliary:POSition</td>
<td>TriggerOutPosition</td>
<td></td>
</tr>
<tr>
<td><strong>Per Point</strong></td>
<td>TRIGger:CHANnel:AUXiliary:INTerval</td>
<td>TriggerOutInterval</td>
<td></td>
</tr>
<tr>
<td><strong>Pulse Duration</strong></td>
<td>TRIGger:CHANnel:AUXiliary:DURation</td>
<td>TriggerOutDuration</td>
<td></td>
</tr>
<tr>
<td><strong>Enable Wait-for-Device Handshake</strong></td>
<td>TRIGger:CHANnel:AUXiliary:HANDshake</td>
<td>HandshakeEnable</td>
<td></td>
</tr>
<tr>
<td><strong>Positive Edge</strong></td>
<td>TRIGger:CHANnel:AUXiliary:IPOLarity</td>
<td>TriggerInPolarity</td>
<td></td>
</tr>
<tr>
<td>Negative Edge</td>
<td>TRIGger:CHANnel:AUXiliary:IPOLarity</td>
<td>TriggerInPolarity</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------</td>
<td>------------------</td>
<td></td>
</tr>
<tr>
<td>Delay</td>
<td>TRIGger:CHANnel:AUXiliary:DElay</td>
<td>Delay</td>
<td></td>
</tr>
<tr>
<td>Pulse Trigger</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trigger Source</td>
<td>SENSe:PATH:CONFig:ELEMent[:STATe]</td>
<td>Element</td>
<td></td>
</tr>
<tr>
<td>High Level</td>
<td>SENSe:PULSe:TTYPE</td>
<td>TriggerInType</td>
<td></td>
</tr>
<tr>
<td>Low Level</td>
<td>SENSe:PULSe:TTYPE</td>
<td>TriggerInType</td>
<td></td>
</tr>
<tr>
<td>Positive Edge</td>
<td>SENSe:PULSe:TPOLarity</td>
<td>TriggerInPolarity</td>
<td></td>
</tr>
<tr>
<td>Negative Edge</td>
<td>SENSe:PULSe:TPOLarity</td>
<td>TriggerInPolarity</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SENSe:PULSe[:STATe]</td>
<td>State</td>
<td></td>
</tr>
<tr>
<td>Synchronize ADCs using pulse trigger</td>
<td>SENSe:PULSe:DElay</td>
<td>Delay</td>
<td></td>
</tr>
<tr>
<td>ADC trigger delay</td>
<td>SENSe:PULSe:DElay</td>
<td>Delay</td>
<td></td>
</tr>
</tbody>
</table>
## COM Example Intro

### COM Example Programs

<table>
<thead>
<tr>
<th>Setup Basic Measurements</th>
<th>VBScript</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Calibrations</strong></td>
<td></td>
</tr>
<tr>
<td>Cal All Independent Calibration Channels</td>
<td>VBScript</td>
</tr>
<tr>
<td>Cal All Multi-Channel Independent Calibration Channels</td>
<td>VBScript</td>
</tr>
<tr>
<td>Cal All SMC Split Cal</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create Multiple Instances of Calibrate All Channels</td>
<td></td>
</tr>
<tr>
<td>Perform a Guided Cal</td>
<td>VBScript</td>
</tr>
<tr>
<td>Perform a Comprehensive Guided 2-Port Cal</td>
<td>VBScript</td>
</tr>
<tr>
<td>Perform a Guided Cal using C++</td>
<td>C++</td>
</tr>
<tr>
<td>Perform a Guided Cal with C#</td>
<td>C#</td>
</tr>
<tr>
<td>Perform a Guided Power Cal using Multiple Power Sensors</td>
<td>VBScript</td>
</tr>
<tr>
<td>Perform an ECal</td>
<td>VBScript</td>
</tr>
<tr>
<td>Perform an ECAL Confidence Check</td>
<td>Visual Basic</td>
</tr>
<tr>
<td>Perform an Unknown Thru or TRL Cal (apply a Delta Match Cal)</td>
<td>VBScript</td>
</tr>
<tr>
<td>Perform an Unguided Cal</td>
<td>VBScript</td>
</tr>
<tr>
<td>Perform Global Delta Match Cal</td>
<td>VBScript</td>
</tr>
<tr>
<td>Perform an Independent Power Calibration</td>
<td>VBScript</td>
</tr>
<tr>
<td>Reading Cal Set Data</td>
<td>VBScript</td>
</tr>
<tr>
<td>Writing Cal Set Data</td>
<td>VBScript</td>
</tr>
<tr>
<td>Perform a Source Power Cal</td>
<td>VBScript</td>
</tr>
<tr>
<td>Upload a Source Power Cal</td>
<td>VBScript</td>
</tr>
<tr>
<td>Perform an ECal User Characterization</td>
<td>VBScript</td>
</tr>
<tr>
<td>Show Custom Window during Calibration</td>
<td>VBScript</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>Perform a CalAllChannels Calibration</td>
<td>VBScript</td>
</tr>
<tr>
<td>Dynamic Uncertainty</td>
<td>VBScript</td>
</tr>
<tr>
<td>Perform a Guided Cal with Sliding Load</td>
<td>VBScript</td>
</tr>
<tr>
<td>Uncertainty on Power Meter</td>
<td>VBScript</td>
</tr>
<tr>
<td>User Defined Power Meter Uncertainty File</td>
<td>VBScript</td>
</tr>
</tbody>
</table>

### Applications

#### Active (Hot) Parameters

<table>
<thead>
<tr>
<th>Create and Cal a Noise Figure Measurement</th>
<th>VBScript</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessing the NoiseFigure and NoiseCal Object</td>
<td>C#</td>
</tr>
<tr>
<td>Create and Cal an NFX Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Setup Noise Figure Port Mapping</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create and Cal a Gain Compression Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create and Cal a GCX Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create an iTMSA Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create and Cal a Swept IMD Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create an IM Spectrum Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create a Diff IQ Measurement.htm</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create a Spectrum Analyzer Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create a Multi-Dimensional Sweep for a Spectrum Analyzer Channel</td>
<td>VBScript</td>
</tr>
</tbody>
</table>

#### FCA

<table>
<thead>
<tr>
<th>Create and Cal an SMC Measurement</th>
<th>VBScript</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create and Cal a VMC Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Perform a VMC Mixer Cal</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create an SMC Fixed Output Measurement</td>
<td>Visual Basic</td>
</tr>
<tr>
<td>Create a Segmented Sweep for Mixers</td>
<td>VBScript</td>
</tr>
<tr>
<td>Task</td>
<td>Language</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Perform an SMC Phase Reference Cal</td>
<td>VBScript</td>
</tr>
<tr>
<td>Set Up Embedded LO Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Use Existing Power Cal for SMC</td>
<td>VBScript</td>
</tr>
<tr>
<td><strong>Pulsed Measurements</strong></td>
<td></td>
</tr>
<tr>
<td>Create a Narrowband Pulsed Measurement (PNA-X)</td>
<td>Visual Basic</td>
</tr>
<tr>
<td>Create a Wideband Pulsed Measurement (PNA-X)</td>
<td>Visual Basic</td>
</tr>
<tr>
<td><strong>Miscellaneous</strong></td>
<td></td>
</tr>
<tr>
<td>IF Path Configuration</td>
<td>VBScript</td>
</tr>
<tr>
<td>RF Path Configuration</td>
<td>Visual Basic, C#</td>
</tr>
<tr>
<td>Frequency Offset Mode</td>
<td>VBScript</td>
</tr>
<tr>
<td>Basic Instrument Control</td>
<td>VEE</td>
</tr>
<tr>
<td>E5091 Testset Control</td>
<td>VBScript</td>
</tr>
<tr>
<td>External Testset Control</td>
<td>VBScript</td>
</tr>
<tr>
<td>Getting Trace Data from the VNA</td>
<td>Visual Basic</td>
</tr>
<tr>
<td>Upload Segment Table</td>
<td>VBScript</td>
</tr>
<tr>
<td>Upload Segment Table</td>
<td>C++</td>
</tr>
<tr>
<td>Create a Balanced Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Create a PMAR Device and Measurement</td>
<td>VBScript</td>
</tr>
<tr>
<td>Limit Line Testing</td>
<td>Visual Basic</td>
</tr>
<tr>
<td>Modify Display Colors</td>
<td>VBScript</td>
</tr>
<tr>
<td>C++ Example</td>
<td>C++</td>
</tr>
<tr>
<td>Errors and the SCPIStringParser Object</td>
<td>C++</td>
</tr>
<tr>
<td>Events Example</td>
<td>Visual Basic</td>
</tr>
<tr>
<td>Events Example</td>
<td>C++</td>
</tr>
<tr>
<td>Setup Phase Control</td>
<td>VBScript</td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>Setup Receiver Leveling</td>
<td>VBScript</td>
</tr>
<tr>
<td>Setup Compression Marker</td>
<td>VBScript</td>
</tr>
<tr>
<td>Setup PNOP and PSAT Marker Search</td>
<td>VBScript</td>
</tr>
<tr>
<td>Using C#</td>
<td>C#</td>
</tr>
<tr>
<td>Get Data</td>
<td>VB.NET</td>
</tr>
<tr>
<td>Power Range</td>
<td>VBScript</td>
</tr>
</tbody>
</table>

See more VNA programming information and examples at: [http://na.support.keysight.com/pna/programming/](http://na.support.keysight.com/pna/programming/)
Create and Cal a Diff IQ Measurement

Create a Differential IQ Measurement

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as DIQ.vbs. Learn how to setup and run the macro.

See Also

DIQ Object

See other COM Examples

```vbs
Set app = CreateObject("AgilentPNA835x.Application")
app.CreateCustomMeasurementEx 2, "Differential I/Q", "IPwrF1"
Set Diq = app.ActiveChannel.CustomChannelConfiguration
Diq.AddRange
Diq.RangeStartFrequency(1) = 1e9
Diq.RangeStopFrequency(1) = 2e9
Diq.RangeCoupleState(2) = 1
Diq.RangeCoupleId(2)=1
Diq.RangeMultiplier(2) = 2
Diq.RangeIFBW(2) = 1000
Diq.SourceRange("Port 1") = 1
Diq.SourceState("Port 1") = naPortON
Diq.MatchState("Port 1") = 1
Diq.MatchFrequencyRange("Port 1") = 1
Diq.MatchTestReceiver("Port 1") = "b3"
Diq.MatchRefReceiver("Port 1") = "a3"
Diq.PowerSweepState("Port 1")=0
Diq.PortStartPower("Port 1")=-5
Diq.PortAttenuator("Port 1") = 5
Diq.AutoRangeState("Port 1") = 1
```
Dynamic Uncertainty

This example does the following in separate subprograms:

- Setup Uncertainty Manager Workspace
- Perform a standard 2-port Guided Cal
- Perform a Cable Characterization
- Perform a Noise Characterization
- Perform a 2-Port Uncertainty Cal
- Make Uncertainty Trace Settings

The cable repeatability and noise characterizations can be performed once, and then they are good for possibly months.

However, the uncertainty calibrations should be performed just as often as traditional non-uncertainty measurement calibrations.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as *.vbs.

See Also

Learn how to setup and run the macro.
See Dynamic Uncertainty commands
Learn about Dynamic Uncertainty (Option S93015A/B)

```vbnet
Dim app
' Create / Get the VNA application.
Set pna = CreateObject("AgilentPNA835x.Application")
' Preset the analyzer
pna.preset

SetupWorkspace
TwoPortGuidedCal
PerformCableChar
PerformNoiseChar
TwoPortUncertCal
MakeTraceSettings
MsgBox "The example program has now completed"
'*******************************************************************************
```
' Setup Uncertainty Workspace
Sub SetupWorkspace
Set cables = pna.UncertaintyManager.Cables
Dim strCableCatalog
For i = 1 To cables.Count
    strCableCatalog = strCableCatalog & cables(i).Name
    If i < cables.Count Then strCableCatalog = strCableCatalog & Chr(10)
    ' Uncomment the following line to reset (clear) repeatability data for the cable
    ' cables(i).ResetRepeatability
Next
MsgBox "Cables:" & Chr(10) & Chr(10) & strCableCatalog
Set ports = pna.UncertaintyManager.Ports
Dim strPortCatalog
For i = 1 To ports.Count
    strPortCatalog = strPortCatalog & "Port " & ports(i).Number & ": " & ports(i).Cable
    If i < ports.Count Then strPortCatalog = strPortCatalog & Chr(10)
    ' Uncomment the following line to reset (clear) noise data for the port
    ' ports(i).ResetNoise
Next
MsgBox "Port Cables:" & Chr(10) & Chr(10) & strPortCatalog
' Uncomment the following line to select the particular cable for all ports
' ports.SelectCableForAllPorts cables(1).Name
' Uncomment this next line to copy the noise data for Port 1 to be used for all ports
' ports.CopyNoiseToAllPorts 1
' Uncomment the following line to reset (clear) the noise data for all ports
' ports.ResetNoiseForAllPorts
' The next lines toggle uncertainty manager properties on
pna.UncertaintyManager.PortNoiseEnabled = True
MsgBox "Noise enabled for cals = " & pna.UncertaintyManager.PortNoiseEnabled
pna.UncertaintyManager.CableRepeatabilityEnabled = True
MsgBox "Cable repeatability enabled for cals = " & pna.UncertaintyManager.CableRepeatabilityEnabled
pna.UncertaintyManager.StandardDefinitionsEnabled = True
MsgBox "Standards definition uncertainties enabled for cals = " & pna.UncertaintyManager.StandardDefinitionsEnabled
' Uncomment the following line to change the max number of uncertainty points for cals to be 500
' pna.UncertaintyManager.MaximumUncertaintyPoints = 500
MsgBox "Max number of uncertainty points for cals = " & pna.UncertaintyManager.MaximumUncertaintyPoints
' Uncomment the following line to perform a Save of the currently loaded uncertainty workspace (".ml4") file
' pna.UncertaintyManager.Save
' Uncomment the following line to save the currently loaded uncertainty workspace to a specific ('.ml4') filename
'pna.UncertaintyManager.Save "myUncertaintyWorkspace.ml4"

' Uncomment the following line to recall the uncertainty workspace ('.ml4') file that we just saved
'pna.UncertaintyManager.Recall "myUncertaintyWorkspace.ml4"

End Sub

'******************************************************************************
' Measure the steps of a cable characterization,
' noise characterization, or a calibration.

Sub MeasureStepsOfCharacterizationOrCal(objCharOrCal, isNoiseCharacterization, numSteps)

For i = 1 To numSteps
    step = "Step " + CStr(i) + " of " + CStr(numSteps)
        ' A "connection adjustment" means the user has to change something about the connection,
        ' such as wiggling cable and re-connecting standard during a repeatability characterization,
        ' or to move the slide position of a sliding load if a connection step in a calibration
        ' were to involve a sliding load.
    If isNoiseCharacterization = False Then
        numConnectionAdjustmentsPerStep = objCharOrCal.MinimumIterationsForStep(i)
    Else ' noise characterization
        ' in this case, "iterations" are just repetitive measurements, not re-adjust/re-measure the standard
        numConnectionAdjustmentsPerStep = 1
    End If

    For j = 1 To numConnectionAdjustmentsPerStep
        strPrompt = objCharOrCal.GetStepDescription(i)
        retVal = MsgBox(strPrompt, vbOKCancel, step)
        If retVal = vbCancel Then WScript.Quit
        objCharOrCal.AcquireStep i
    Next

Next

objCharOrCal.GenerateErrorTerms

End Sub

Function FormatList(tokens)

    For i = 0 To UBound(tokens)
        list = list & tokens(i) & Chr(10)
    Next

    FormatList = list

End Function

'******************************************************************************
'Performing a NON-UNCERTAINTY Guided 2-port cal (Ports 1 and 2)

Sub TwoPortGuidedCal

message = "A non-uncertainty 2-port calibration will now be performed, "
message = message & "because a calibration is needed for characterizing the cables"
MsgBox message

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set calMgr = app.GetCalManager
Set guidedCal = calMgr.GuidedCalibration
Set chan = app.ActiveChannel
chanNum = chan.ChannelNumber

' Initialize guided cal to be performed on the active channel.
' The boolean argument of True indicates to create a new calset
' for storing the new calibration to.
guidedCal.Initialize chanNum, True

' Query the connectors that the VNA system recognizes
conns = guidedCal.ValidConnectorTypes

' Format the list string with linefeed characters between each substring
connList = FormatList(conns)

' Select the connector for Port 1
selectedConn1 = InputBox("Enter your DUT connector for Port 1. Choose from this list:" & _
    Chr(10) & Chr(10) & connList)
If selectedConn1 = "" Then Exit Sub
guidedCal.ConnectorType(1) = selectedConn1

' Select the connector for Port 2
selectedConn2 = InputBox("Enter your DUT connector for Port 2. Again, choose from this list:" & _
    Chr(10) & Chr(10) & connList)
If selectedConn2 = "" Then Exit Sub
guidedCal.ConnectorType(2) = selectedConn2

' Note: If your VNA has more than 2 ports, you would need to uncomment
' one or both of these next two lines, to explicitly specify this is
' just a 2-port cal.
'guidedCal.ConnectorType(3) = "Not used"
'guidedCal.ConnectorType(4) = "Not used"

' Query the list of acceptable cal kits and ECal module characterizations for Port 1.
kits = guidedCal.GetCompatibleCalKits(selectedConn1)

' Format the list string with linefeed characters between each substring
kitList = FormatList(kits)

' Select the Cal Kit or ECal module characterization to use for Port 1.
selectedKit = InputBox("Enter your cal kit or ECal module characterization for Port 1. ", "Choose from this list:" & Chr(10) & Chr(10) & kitList)
If selectedKit = "" Then Exit Sub
guidedCal.CalKitType(1) = selectedKit
' Query the list of acceptable cal kits and ECal module characterizations for Port 2.
kits = guidedCal.GetCompatibleCalKits(selectedConn2)
' Format the list string with linefeed characters between each substring
kitList = FormatList(kits)
' Select the Cal Kit or ECal module characterization to use for Port 2.
selectedKit = InputBox("Enter your cal kit or ECal module characterization for Port 2. ", "Choose from this list:" & Chr(10) & Chr(10) & kitList)
If selectedKit = "" Then Exit Sub
guidedCal.CalKitType(2) = selectedKit
' This determines whether the cal will be a "Guided Power Cal"
or just a traditional S-parameter cal.
message = "On which port number shall power be measured? ", message = message & "For a traditional guided cal without power cal, enter 0"
Dim powerPort
powerPort = CInt( InputBox(message) )
If powerPort > 0 Then
    guidedCal.PerformPowerCalibration(powerPort) = True
    Dim retVal
    retVal = MsgBox("Is the power sensor's connector type or gender different from the DUT connector for that port?", vbYesNo)
    If retVal = vbYes Then
        message = "Enter your power sensor's connector. Choose from this list:"message = message & Chr(10) & Chr(10) & connList
        ' Select the sensor's connector.
        selectedConn1 = InputBox(message)
        If selectedConn1 = "" Then Exit Sub
        guidedCal.PowerSensorConnectorType(powerPort) = selectedConn1
        ' Query the list of acceptable cal kits and ECal module characterizations
        ' that are applicable for the sensor's connector.
kits = guidedCal.GetCompatibleCalKits(selectedConn1)
        ' Format the list string with linefeed characters between each substring
        kitList = FormatList(kits)
        message = "Enter your cal kit or ECal module characterization to use for de-embed of the sensor's connector. ", message = message & "Choose from this list:"message = message & Chr(10) & Chr(10) & kitList
' Select the Cal Kit or ECal module characterization to use for de-embed of the sensor's connector.
    selectedKit = InputBox(message)
    If selectedKit = "" Then Exit Sub
    guidedCal.PowerSensorCalkitType(powerPort) = selectedKit
Else
    guidedCal.PowerSensorConnectorType(powerPort) = "Ignored"
End If
' End of block that considers the sensor's connector
' Ask for the power level to perform the power cal at
' (if this command is omitted, the default is 0 dBm).
Dim powerLevel
powerLevel = InputBox("Enter the power level for the power cal to be performed at")
If powerLevel = "" Then Exit Sub
    guidedCal.PowerCalibrationPowerLevel(powerPort) = CDbl(powerLevel)
Else
    guidedCal.PerformPowerCalibration(1) = False
End If
' End of block that considers if the cal will include power calibration
'-----------------------------------------------------------------------------
' This next block of commented-out code shows optional functions when using ECal.
' These OrientECALModule and ECALPortMapEx properties would need to be set prior to
calling GenerateSteps on the guidedCal object.
' Read the information about the Keysight factory characterization data
' of ECal module #1 on the USB bus
'Set calibrator = chan.Calibrator
'Start ECalModule1 = 1
'module1Info = calibrator.GetECALModuleInfoEx(ECalModule1)
'MsgBox "Description of ECal module #1:" & Chr(10) & Chr(10) & module1Info
' By default, during calibration the VNA automatically determines the orientation of
' the ECal module (senses which port of the module is connected to which port of the
' VNA).
' However, since this setting could have recently been overridden by another user of
' the instrument, use this next line to ensure the auto orientation setting is enabled.
'calibrator.OrientECALModule = True
' Alternatively, if you are measuring at very low power levels where
' the VNA fails to sense the module's orientation, you may need to turn off the auto
' orientation and specify how the module is connected (as in these next two lines of
code,
' "A1,B2" would indicate Port A of the module is connected to Port 1 and
' Port B is connected to Port 2).
'calibrator.OrientECALModule = False
'calibrator.ECALPortMapEx(ECalModule1) = "A1,B2"
' End of optional ECal setup
' Select the thru method of Default. This instructs the VNA to determine which thru
' standard measurement technique to use, based upon the selected connectors and
' calibration kit(s) and what model of VNA this is.
guidedCal.ThruCalMethod = 0 ' 0 = naDefaultCalMethod
' Initiate the calibration and query the number of steps
numCalSteps = guidedCal.GenerateSteps
MsgBox "Number of steps is " + CStr(numCalSteps)
' Measure the standards
MeasureStepsOfCharacterizationOrCal guidedCal, False, numCalSteps
MsgBox "The cal needed for the cable repeatability characterizations is done"
End Sub

'*************************************************************************************
' Perform Cable Characterization
Sub PerformCableChar
pna.ActiveMeasurement.ErrorCorrection = True
Set uncchar = pna.UncertaintyManager.Characterizer
' Initiate repeatability characterizations on Channel 1 for Ports 1 and 2, with 3
' iterative measurements of each standard (normally users should specify more than 3
' iterations per standard but it just makes this example speedier)
umCharSteps = uncchar.InitiateCableCharacterization(1, 1, 3)
message = "Now to characterize the cable currently associated with port 1. "
message = message & "Number of standard steps for characterizing port 1 cable = " & numCharSteps
MsgBox message
' Get guided cal object for Channel 1 to use for performing the characterization
Set guidedcal = uncchar.GuidedCalibration(1)
MeasureStepsOfCharacterizationOrCal guidedcal, False, numCharSteps
MsgBox "Cable characterization for port 1 is done"
umCharSteps = uncchar.InitiateCableCharacterization(1, 2, 3)
message = "Now to characterize the cable currently associated with port 2. "
message = message & "Number of standard steps for characterizing port 2 cable = " & numCharSteps
MsgBox message
Set guidedcal = uncchar.GuidedCalibration(1)
MeasureStepsOfCharacterizationOrCal guidedcal, False, numCharSteps
MsgBox "Cable characterization for port 2 is done"
End Sub

'*************************************************************************************
'Perform Noise Characterization
Sub PerformNoiseChar
pna.ActiveMeasurement.ErrorCorrection = True
Set unchar = pna.UncertaintyManager.Characterizer

' Initiate noise characterization on Channel 1 for Ports 1 and 2, with 3 iterative measurements of each standard (normally users should specify more than 3 iterations per standard but it just makes this example speedier)
numCharSteps = unchar.InitiateNoiseCharacterization(1, 1, 2, 3)
message = "Now to characterize the system noise for ports 1 and 2. "
message = message & "Number of connection steps to be made = " & numCharSteps
MsgBox message

' Get guided cal object for Channel 1 to use for performing the characterization
Set guidedcal = unchar.GuidedCalibration(1)
MeasureStepsOfCharacterizationOrCal guidedcal, True, numCharSteps
MsgBox "Noise characterization is done"

'*******************************************************
'Perform 2-port Uncertainty Calibration (Ports 1 and 2) on Channel 1
Sub TwoPortUncertCal

' Note: normally to use the Application's ActiveChannel property for this, you would have to ensure your active channel is an S-parameter channel before this point, but as the PerformCableChar and PerformNoiseChar subroutines have executed right before this one, that is already ensured.
' Create / Get the VNA application.
Set calMgr = app.GetCalManager
Set guidedCal = calMgr.GuidedCalibration
Set chan = app.ActiveChannel
chanNum = chan.ChannelNumber

' Initialize guided cal to be performed on the active channel.
' The boolean argument of True indicates to create a new calset for storing the new calibration to.
guidedCal.Initialize chanNum, True

' Specify this calibration is to be performed with uncertainties.
guidedCal.UncertaintyEnabled = True

' Query the list of uncertainty cal kits and ECal module factory characterizations applicable for the connector of the cable associated with Port 1.
kits = guidedCal.CompatibleCalKits(1)

' Format the list string with linefeed characters between each substring
kitList = FormatList(kits)

' Select the Cal Kit or ECal module to use for both ports.
' If the selected kit does not have a suitable set of standards that can mate the connector of the cable attached to Port 2, then the attempt to GenerateSteps should yield an applicable error message.
message = "Now to perform an uncertainty calibration for ports 1 and 2. "
message = message & "Enter your cal kit or ECal module to use. Choose from this list:"
message = message & Chr(10) & Chr(10) & kitList
selectedKit = InputBox(message)
If selectedKit = "" Then Exit Sub
guidedCal.CalKitType(1) = selectedKit
guidedCal.CalKitType(2) = selectedKit
' Note: If your VNA has more than 2 ports, then should uncomment
' one or both of these next two lines just in case this channel
' may have had a remote non-Dolcetto cal performed prior that
' involved port 3 and/or 4.
'guidedCal.CalKitType(3) = ""
guidedCal.CalKitType(4) = ""
' This next block of commented-out code shows optional functions when using ECal.
' These OrientECALModule and ECALPortMapEx properties would need to be set prior
' to calling GenerateSteps on the guidedCal object.
'-------------------------------------------------------------------------------
' Read the information about the factory characterization data
' of ECal module #1 on the USB bus
'Set calibrator = chan.Calibrator
'Const ECalModule1 = 1
'module1Info = calibrator.GetECALModuleInfoEx(ECalModule1)
'MsgBox "Description of ECal module #1:" & Chr(10) & Chr(10) & module1Info
' By default, during calibration the VNA automatically determines the orientation
' of the ECal module (senses which port of the module is connected to which port of
' the VNA). However, since this setting could have recently been overridden by
' another user of the instrument, use this next line to ensure the auto orientation
' setting is enabled.
'calibrator.OrientECALModule = True
' Alternatively, if you are measuring at very low power levels where
' the VNA fails to sense the module's orientation, you may need to turn off the auto
' orientation and specify how the module is connected (as in these next two lines
' of code, "A1,B2" would indicate Port A of the module is connected to Port 1 and
' Port B is connected to Port 2).
'calibrator.OrientECALModule = False
'calibrator.ECALPortMapEx(ECalModule1) = "A1,B2"
'-------------------------------------------------------------------------------
' End of optional ECal setup
' Initiate the calibration and query the number of steps
numCalSteps = guidedCal.GenerateSteps
MsgBox "Number of cal steps is " + CStr(numCalSteps)
MeasureStepsOfCharacterizationOrCal guidedCal, False, numCalSteps
MsgBox "Cal is done"
End Sub

'*********************************************
'Make Uncertainty Trace Settings
Sub MakeTraceSettings
' Note: normally to use the Application's ActiveMeasurement property for this,
' you would have to first ensure your active measurement is an S-parameter channel
' measurement before this point, and that the channel has an uncertainty cal to be
' able to turn on uncertainty display, but as the TwoPortUncertCal subroutine has
' executed right before this one, that is already ensured.
Set unc = pna.ActiveMeasurement.Uncertainty
MsgBox "DisplayType = " & GetUncDispEnumValueName(unc.DisplayType)
On Error Resume Next
unc.DisplayType = 4 ' naUncertaintyDisplayShade
If Err.Number <> 0 Then
   MsgBox Err.Description
   Err.Clear
Else
   MsgBox "DisplayType = " & GetUncDispEnumValueName(unc.DisplayType)
End If
On Error GoTo 0
MsgBox "CoverageFactor = " & unc.CoverageFactor
' Uncomment the following line to try setting the CoverageFactor = 3
'unc.CoverageFactor = 3
unc.CableRepeatabilityUncertainty = False
MsgBox "CableRepeatabilityUncertainty = " & unc.CableRepeatabilityUncertainty
unc.CableRepeatabilityUncertainty = True
MsgBox "CableRepeatabilityUncertainty = " & unc.CableRepeatabilityUncertainty
unc.MeasurementNoiseUncertainty = False
MsgBox "MeasurementNoiseUncertainty = " & unc.MeasurementNoiseUncertainty
unc.MeasurementNoiseUncertainty = True
MsgBox "MeasurementNoiseUncertainty = " & unc.MeasurementNoiseUncertainty
unc.ErrorTermUncertainty = False
MsgBox "unc.ErrorTermUncertainty = " & unc.ErrorTermUncertainty
unc.ErrorTermUncertainty = True
MsgBox "unc.ErrorTermUncertainty = " & unc.ErrorTermUncertainty
End Sub
Function GetUncDispEnumValueName(uncDispTypeEnum)
Select Case uncDispTypeEnum
   Case 0
      GetUncDispEnumValueName = "naUncertaintyDisplayNone"
   Case 1

GetUncDispEnumValueName = "naUncertaintyDisplayMaximum"

Case 2
    GetUncDispEnumValueName = "naUncertaintyDisplayMinimum"

Case 3
    GetUncDispEnumValueName = "naUncertaintyDisplayBar"

Case 4
    GetUncDispEnumValueName = "naUncertaintyDisplayShade"

Case 5
    GetUncDispEnumValueName = "naUncertaintyDisplayEllipse"

Case Else
    GetUncDispEnumValueName = "Unexpected display type enum value!"

End Select
End Function
Get Data using VB.NET

In VB.NET, you can use the `getData` and `putData` functions shown in the `IArrayTransfer` interface. However, those functions return Variant data types, and VB.NET does not support Variant data types.

In this example, `getData` returns an "object", which is actually an "object[]" (array of object). Each object in the array is a "float". A new array of floats is created and then each instance is cast within the array to a float.

```vbnet
Dim DataAsObject As Object
Dim DataAsObjectArray(NumPoints - 1) As Object
Dim ResultData (NumPoints - 1) As Double

DataAsObject = na.Measurements.Item(1)

DataAsObjectArray = DataAsObject.getData(AgilentPNA835x.NADataStore.naMeasResult, AgilentPNA835x.NADataFormat.naDataFormat_LogMag)

For i As Integer = 0 To NumPoints - 1
    ResultData(i) = CType(DataAsObjectArray(i), Double)
Next i
```
IFPathConfiguration Setup

The following demonstrates how to:

  Set the Source 1 Pulse Mod to enable
  Set the Source 2 Pulse Mod to disable. In effect, this sets Source 2 to 'CW' (or OFF)
  Set the Pulse Mod drive to Pulse1

See the entire list of IF Path Configuration settings.

```
' Create / Get the VNA application
Dim app
Set app = CreateObject("AgilentPNA835x.Application")
' Preset the instrument
app.Preset
' Get a channel interface on which to operate
Dim chan
Set chan = app.ActiveChannel
' Set the Source 1 Pulse Mod to enable
chan.PathConfiguration.Element("Src1Out1PulseModEnable").Value = "Enable"
' Set the Source 2 Pulse Mod to disable
chan.PathConfiguration.Element("Src2Out1PulseModEnable").Value = "Disable"
' Set the Pulse Mod drive to Pulse1
chan.PathConfiguration.Element("PulseModDrive").Value = "Pulse1"
```
Perform a CalAllChannels Calibration

This example sets up an SMC channel and a standard channel. It then performs a 'Cal All Channels' calibration which calibrates both channels.

Note: The example does NOT modify any path configuration settings.

See Also

CalibrateAllChannels Object
Converter Object
GuidedCalibration Object
Learn about Cal All

See other COM Examples

'Access the CalAllChannels object
Dim app
Set app = CreateObject("AgilentPNA835x.Application")
Dim mgr
Set mgr = app.GetCalManager
'Preset VNA
app.Preset

********** Setup Channel 1, Window 1 **********
' Get a handle to the preset channel 1
set meas1=app.ActiveMeasurement
set chan1=app.ActiveChannel
set win1=app.ActiveNAWindow
'Modify stimulus settings
'Set IF Bandwidth to 1kHz
chan1.IFBandwidth = 1e3
'Set Center and Span Freq's to 1 GHz
chan1.CenterFrequency = 1e9
chan1.FrequencySpan = 1e9
'Set number of points to 11
chan1.NumberOfPoints = 11

'********** Setup SMC in Channel 2 in new window**********
app.CreateCustomMeasurementEx 2, "Scalar Mixer/Converter","SC21",2
Set smcChan = app.ActiveChannel
smcChan.NumberOfPoints = 11
smcChan.IFBandwidth = 1e3

' the rest are Mixer settings
dim mixer
set mixer = app.ActiveMeasurement
mixer.InputRangeMode = 0 'swept Input
mixer.InputStartFrequency = 3.6e9
mixer.InputStopFrequency = 3.9e9
mixer.LORangeMode(1) = 1 'fixed LO
mixer.LoFixedFrequency(1) = 1e9
mixer.LOPower(1) = 10
mixer.LOName(1) = "Port 3"
mixer.OutputSideband = 1 'Low side
mixer.Calculate 2 'Calc Output
mixer.Apply

'*********** Setup CalAll Channel
Dim CalAll
Set CalAll = mgr.CalibrateAllChannels
'Reset Cal All settings
CalAll.Reset
'Select the channels to cal
'VMC channels are not supported by Cal All.
CalAll.Channels = Array(1,2)
'Set IFBW
CalAll.IFBW = 1e3
'Set power level for port 1
CalAll.PowerLevel(1) = 0
'Set CalSet prefix. The channel number is appended to
'the User Cal set for each channel.
'If you don’t set this, only Cal Registers will be generated.
callAll.CalsetPrefix = "MyCalAll"
' Read unique Cal settings for Cal All channels
props = calAll.PropertyNames
msg = "callAll.PropertyNames:" & vbCrLf
For i = 0 to UBound(props)
    msg = msg & " " & CStr(i) & ". " & props(i) & vbCrLf
Next
resp = MsgBox(msg)
'We want "Enable Phase Correction"
' Now find valid settings
propName = "Enable Phase Correction"
uniqVal = calAll.PropertyValues (propName)
'returns 'false,true' make it true.
For i = 1 to UBound(uniqVal)
    MsgBox "calAll.PropertyValues (" & propName & ") = " & uniqVal(i)
Next
calAll.PropertyValue(propName) = "true"
'
'********** 'Perform Guided Cal **********
' Get a handle to the GuidedCal object
Dim guidedcal
set guidedcal = calAll.GuidedCalibration
' Specify the DUT connectors
guidedcal.ConnectorType(1) = "APC 3.5 male"
guidedcal.ConnectorType(2) = "APC 3.5 male"
guidedcal.ConnectorType(3) = "Not used"
guidedcal.ConnectorType(4) = "Not used"
MsgBox "Connectors defined for Ports 1 and 2"
' Select the Cal Kit for each port being calibrated.
guidedCal.CalKitType(1) = "85052D"
guidedCal.CalKitType(2) = "85052D"
MsgBox "Cal kits defined for Ports 1 and 2"
' Initiate the calibration and query the number of steps
Numsteps = guidedcal.GenerateSteps
For i = 1 to Numsteps
    step = "Step " + CStr(i) + " of " + CStr(Numsteps)
    strPrompt = guidedCal.GetStepDescription(i)
    value = MsgBox(strPrompt, vbOKOnly, step)
    guidedCal.AcquireStep i
Next
' Conclude the calibration
guidedCal.GenerateErrorTerms
csets = calAll.GeneratedCalSets
For i = 0 to UBound(uniqVal)
    MsgBox (csets(i))
Next
Perform a Guided Cal with Sliding Load

This example sets the sliding load behavior, then performs a Guided Cal using a sliding load.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as Calibrate.vbs. Learn how to setup and run the macro.

See the GuidedCalibration Object

See Other COM Example Programs

```
Set app = CreateObject("AgilentPNA835x.Application")
Set calMgr = app.GetCalManager
Set guidedCal = calMgr.GuidedCalibration
Set chan = app.ActiveChannel
chanNum = chan.ChannelNumber
guidedCal.Initialize chanNum, True
' Specify that any sliding loads should be measured using the
' remote iterative method rather than launching sliding load dialog.
' 0 = naShowDialog, 1 = naMeasureSlidePosition
guidedCal.SlidingLoadAcquisitionBehavior = 1
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 male"
' 85052B cal kit uses sliding loads
guidedCal.CalKitType(1) = "85052B"
guidedCal.CalKitType(2) = "85052B"
numSteps = guidedCal.GenerateSteps
' Measure the standards
For i = 1 To numSteps
    step = "Step " + CStr(i) + " of " + CStr(numSteps)
    strPrompt = guidedCal.GetStepDescription(i)
    MsgBox strPrompt, vbOKOnly, step
    minIterations = guidedCal.MinimumIterationsForStep(i)
    For j = 1 To minIterations
        If minIterations > 1 Then MsgBox "Adjust/position the standard for measurement " +
        CStr(j) + " of " + CStr(minIterations), vbOKOnly
        guidedCal.AcquireStep i
    Next
    If guidedCal.IterationCountForStep(i) <> minIterations Then
        MsgBox "Unexpected error!", vbOKOnly, step
```

7843
guidedCal.ResetStep i
    End If
Next
'
Conclude the calibration
guidedCal.GenerateErrorTerms
Perform a Guided Power Cal using Multiple Power Sensors

This example uses the following interfaces to perform a power calibration using two power sensors.

GuidedCalibration
GuidedCalibrationPowerSensors
GuidedCalibrationPowerSensor

Learn more about Reading and Writing Calibration data using COM.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as Calibrate.vbs. Learn how to setup and run the macro.

See Other COM Example Programs

```vbs
set app = CreateObject("Agilentpna835x.application")
app.Reset
app.CreateMeasurement 1, "S11", 0, -1 ' create default S11 parameter
app.ActiveChannel.StartFrequency = 1e9
app.ActiveChannel.StopFrequency = 2e9
set calMgr = app.GetCalManager()
set guidedCal = calMgr.GuidedCalibration
guidedCal.Initialize 1, false
guidedCal.ConnectorType(1)="APC 3.5 male"
guidedCal.ConnectorType(2)="APC 3.5 female"
guidedCal.CalKitType(1)= "N4691-60004 ECal"
guidedCal.CalKitType(2)="N4691-60004 ECal"
guidedCal.PerformPowerCalibration(2)= true
set sensors = guidedCal.GuidedCalibrationPowerSensors(2)
sensors.UseMultipleSensors = true
sensors.Item(1).Name = "pmar1"
sensors.Item(1).StartFrequency = 1e9
sensors.Item(1).StopFrequency = 1.5e9
sensors.Add "pmar2"
sensors.Item(2).StartFrequency = 1.5e9
sensors.Item(2).StopFrequency = 2e9
numSteps = guidedCal.GenerateSteps()
for i = 0 to numSteps-1
desc = guidedCal.GetStepDescription(i + 1)
```
msgbox desc
    guidedCal.AcquireStep(i + 1)
next
guidedCal.GenerateErrorTerms()
Perform a VMC Mixer Cal

This example performs a VMC Mixer Characterization ONLY. To run this example program without error:

   Replace the ECal module model and serial number with that of your own, or a mechanical cal kit model.
   Store a 'default.csa' instrument state file on the VNA with the setup information for your mixer. Or add mixer setup information to this program. See an example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as *.vbs.

Learn how to setup and run the macro.

See Other COM Example Programs

```vbs
Dim pna: Set pna = CreateObject("AgilentPNA835x.application")
pna.recall "c:\users\public\network analyzer\documents/default.csa"
Set chan = pna.activechannel
DoBasicVMCCal (chan.channelNumber)

Sub DoBasicVMCCal( channel )
Dim myMixerCharFile: myMixerCharFile = "C:\Program Files(x86)\Keysight\Network Analyzer\Documents\com_characterize.s2p"

' construct a VMC calibration object
Dim calmanager: Set calmanager = pna.GetCalManager
Dim guidedCal: Set guidedCal = calmanager.CreateCustomCalEx( channel )
Dim vmc: Set vmc = guidedCal.CustomCalConfiguration

' Initialize the cal object.
' usecalsetpreference is unused for the mixer characterization wizard
Dim useCalSetPreference: useCalSetPreference = false
vmc.Initialize channel, useCalSetPreference

' Define the DUT connectors and kits at port 1 of the VNA
vmc.ConnectorType (1) = "APC 3.5 female"
vmc.CalKitType(1) = "N4691-60004 ECal 02593"

' Define the DUT connectors for the output of the characterization mixer
' Use (logical) Port 3.
vmc.ConnectorType(3) = "APC 3.5 female"
```

7847
' Specify the mechanical cal kit for measuring the characterization mixer
vmc.CalKitType(3) = "N4691-60004 ECal 02593"
vmc.CharacterizeMixerOnly = true ' this specifies that we will create a characterization files
' this file will be written (.s2p and .s2px file)
vmc.CharFileName = MyMixerCharFile
vmc.AutoOrient = True
' For the mixer char step ONLY,
' Auto orientation is turned OFF by the VNA.
' Otherwise it would fail because of the loss of the mixer.
' Manually set the ECal orientation for that step.
vmc.EcalOrientation1Port(1) = "B1"
' the main calibration loop
' a description for the connection instructions is read
' and then the standard is acquired
dim steps,connectionPrompt
steps = vmc.GenerateSteps
msgbox "Number of Steps = " + cstr(steps)
if (steps > 0) then ' otherwise an error condition occurred
for i = 1 to steps
    connectionPrompt = vmc.GetStepDescription( i )
    msgbox connectionPrompt
    vmc.AcquireStep( i )
next
vmc.GenerateErrorTerms
end if
end sub
Perform an SMC Phase Reference Cal

This example sets Phase Reference Cal properties, then performs a Phase Reference calibration. It is NOT necessary to create an SMC measurement before performing a remote Phase Reference Cal. It is necessary when performed from the user interface.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as PMAR.vbs. Learn how to setup and run the macro.

See Also
PhaseReferenceCalibration Object
Learn about SMC+ Phase

---

Example Program: performs a phase reference calibration between 1Ghz to 10Ghz on Ports 1, 2, and 4.

```vbscript
Set app = CreateObject("agilentpna835x.application")
app.preset
app.activechannel.hold 1
Set PhaseReferenceCal = app.GetCalManager().PhaseReferenceCalibration
' always call reset at the beginning of your cal
PhaseReferenceCal.Reset
PhaseReferenceCal.StartFrequency = 1e9
PhaseReferenceCal.StopFrequency = 10e9
' Read serial numbers of connected phase references
' Into variant array
refs = PhaseReferenceCal.GetConnectedPhaseReferences
' Change the following to your own phase reference name
PhaseReferenceCal.PhaseReference = "MYPILOT44"
PhaseReferenceCal.SourceAttenuator = 10
PhaseReferenceCal.Calset = "Remote Phase Reference"
PhaseReferenceCal.ConnectorType = "APC 3.5 female"
PhaseReferenceCal.CalKitType = "85052D"
' Perform the calibration on Port 4 (Port 1 and Port 2 are always included)
PhaseReferenceCal.IncludePort(4) = true
' Uncomment the following line to use an unknown mixer
' to extend the phase reference cal down to 10 Mhz
' If an unknown mixer is used, then the start frequency is always 10 Mhz
' PhaseReferenceCal.IncludeUnknownMixer = true
```
Set guidedCal = PhaseReferenceCal.GuidedCalibration
Steps = guidedCal.GenerateSteps
for i = 1 to steps
    MsgBox guidedCal.GetStepDescription(i)
    guidedCal.AcquireStep i
next
Msgbox guidedCal.GenerateErrorTerms
Msgbox "done"
OBS_AcquireCalConfidenceCheckECAL

Transfers confidence data from the specified ECal module into the measurement's memory trace. The data is transferred to the specified S-parameter on the same channel as this Calibrator object.

The characterization within the ECal module that the confidence data will be read from is specified by the **ECALCharacterization property** on the ICalibrator2 interface. The default value of the ECALCharacterization property is `naECALFactoryCharacterization`.

**VB Syntax**

```vbnet
cal.AcquireCalConfidenceCheckECAL Sparam[,ecalModule]
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>cal</em></td>
<td>A Calibrator (object)</td>
<td></td>
</tr>
<tr>
<td><em>Sparam</em></td>
<td>(String) S-parameter to transfer confidence data to. This parameter must be present on the same channel as the calibrator object.</td>
<td></td>
</tr>
<tr>
<td><em>ecalModule</em></td>
<td>(enum NAECALModule) – Optional argument. ECal module. Choose from: 0 - naECALModule_A (default, if unspecified) 1 - naECALModule_B</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**

None

**Default**

Not applicable

**Examples**

```vbnet
Cal.AcquireCalConfidenceCheckECAL "S11", naECALModule_A
```

**C++ Syntax**

```cpp
HRESULT AcquireCalConfidenceCheckECAL(_bstr_t strParameter, enum NAECALModule ecalModule);
```
Interface

ICalibrator
## OBSAcquirePowerReadings Method

### VB Syntax

```vbnet
powerCalibrator.AcquirePowerReadings device [,sync]
```

### Variable

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(object)</td>
<td>A <code>SourcePowerCalibrator</code> object</td>
</tr>
</tbody>
</table>
| (enum NAPowerAcquisitionDevice) | The specific device (sensor on the power meter) to be used for the acquisition. Choose from:
- 0 – `naPowerSensor_A`
- 1 – `naPowerSensor_B`

To use the sensor that currently corresponds to the frequency of interest, use the value from the `PowerAcquisitionDevice` property.

| (boolean) | Optional argument. If not specified, value is set to False. Choose from:
- True (1) – The method does not return until this acquisition has completed (the program calling this method is halted while waiting for the method to return).
- False (0) – The method initiates an acquisition then returns immediately (while the acquisition still proceeds). The program calling this method can then perform other operations during the acquisition. |

### Return Type

None

### Default

Not Applicable

### Examples

```
powerCalibrator.AcquirePowerReadings naPowerSensor_A, True
```

### C++ Syntax

```cpp
HRESULT AcquirePowerReadings(tagNAPowerAcquisitionDevice enumAcqDevice, VARIANT_BOOL bSync);
```

### Interface

`ISourcePowerCalibrator`
AddRange Method

Description
Adds a frequency range using the next available range name. For example, with only the default F1 range name present, sending this command will create F2. On the Measurement Setup dialog this is the New setting.

VB Syntax
DIQ.AddRange

Variable
(Type) - Description

DIQ A DIQ Object

Return Type
Not Applicable

Default
Not Applicable

Examples
diq.AddRange 'Adds a new frequency range at the next available.
See example program

C++ Syntax
HRESULT AddRange()

Interface
DIQ1
OBSApplyPowerCorrectionValues Method

Applies the array of power correction values to the channel memory and turns correction ON. Perform after completing a source power cal acquisition sweep.

This command does NOT save the correction values. To save correction values, save an instrument / calibration state (*.cst file) after performing a source power cal.

VB Syntax

```vbnet
powerCalibrator.ApplyPowerCorrectionValues
```

Variable (Type) - Description

<table>
<thead>
<tr>
<th>powerCalibrator (object) - A SourcePowerCalibrator object</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

Return Type

None

Default

Not Applicable

Examples

```vbnet
powerCalibrator.ApplyPowerCorrectionValues
```

C++ Syntax

```cpp
HRESULT ApplyPowerCorrectionValues();
```

Interface

ISourcePowerCalibrator
BandpowerData Property

Description
Returns the Y-axis data from the band power marker.

VB Syntax
value = mkr.BandpowerData

Variable (Type) - Description

value
Variable to store the returned Y-axis value.

mkr
A Marker (object)

Return Type
Double

Default
Not Applicable

Examples
See example program

C++ Syntax
HRESULT get_BandpowerData(double* pVal)

Interface
IMarker6
# CascadeS2PFiles Method

**Description**

Combines the losses and phase shift of two S2P files into a single S2P file. [Learn more.](#)

**VB Syntax**

```
calmgr.CascadeS2PFiles (s2p1, s2p2, s2pResult, format)
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>calMgr</td>
<td>Cal Manager (Object)</td>
</tr>
<tr>
<td>s2p1</td>
<td>Path and filename of one of the S2P files to be combined.</td>
</tr>
<tr>
<td>s2p2</td>
<td>Path and filename of the other S2P file to be combined.</td>
</tr>
<tr>
<td>s2pResult</td>
<td>Path and filename of the combined S2P file.</td>
</tr>
<tr>
<td>format</td>
<td>Format in which the data is to be saved to the combined S2P file. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naLogMagPhase</td>
</tr>
<tr>
<td></td>
<td>1 - naLinMagPhase</td>
</tr>
<tr>
<td></td>
<td>2 - naRealImaginary</td>
</tr>
</tbody>
</table>

**Return Type**

Not Applicable

**Default**

Not Applicable

**Examples**

```
CalManager.CascadeS2PFiles C:\Users\PNA-ADMIN\AppData\Local\Keysight\Network Analyzer\CPM\a.s2p","C:\Users\PNA-ADMIN\AppData\Local\Keysight\Network Analyzer\CPM\b.s2p","C:\Users\PNA-ADMIN\AppData\Local\Keysight\Network Analyzer\CPM\c.s2p",1
```

**C++ Syntax**

```
HRESULT CascadeS2PFiles(BSTR s2p1, BSTR s2p2, BSTR s2pCombo, tagNAPairedDataFormat format);
```

**Interface**

`ICalManager10`
CharacterizeFixture Method

Description
Characterizes a single fixture based on two existing calsets. To characterize two adapter/fixtures (on the input AND output of a fixture), perform this operation twice.

VB Syntax
```
calmgr.CharacterizeFixture (tier1Cset, tier2Cset, portNum, s2pFile, format, [dcPhase])
```

Variable

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>calMgr</td>
<td>(Object)</td>
<td>Cal Manager (Object)</td>
</tr>
<tr>
<td>tier1Cset</td>
<td>(String)</td>
<td>(String) Name of Tier-1 calset. The Tier-1 calset MUST be from a calibration that was performed at the input to the adapter/fixture.</td>
</tr>
<tr>
<td>tier2Cset</td>
<td>(String)</td>
<td>(String) Name of Tier-2 calset. The Tier-2 calset MUST be from a calibration that was performed at the DUT reference plane.</td>
</tr>
<tr>
<td>portNum</td>
<td>(Long Integer)</td>
<td>Port number of the calsets to use when the specified calsets are used to calibrate more than one port.</td>
</tr>
<tr>
<td>s2pFile</td>
<td>(String)</td>
<td>Path and filename to which the S2P file is to be saved.</td>
</tr>
<tr>
<td>format</td>
<td>(Enum as NAPairedDataFormat)</td>
<td>Format in which the data is to be saved to the S2P file. Choose from:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 - naLogMagPhase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - naLinMagPhase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - naRealImaginary</td>
</tr>
<tr>
<td>[dcPhase]</td>
<td>(Integer)</td>
<td>Optional argument. The projected phase of S21 at DC where it crosses the X-axis. Learn more.</td>
</tr>
</tbody>
</table>

Return Type
Not Applicable

Default
Not Applicable

Examples
```
CalManager.CharacterizeFixture ("Tier1", "Tier2", 1,"C:\Users\PNA-ADMIN\AppData\Local\Keysight\Network Analyzer\CPM\FixtureA.s2p",1,0)
```

C++ Syntax
```
HRESULT CharacterizeFixture(BSTR strCset1, BSTR strCset2, long pNum, BSTR fileName, tagNAPairedDataFormat format, Int dcPhase);
```

Interface
ICalManager10
**OBSCfgEnhancedNB Method**

This subroutine determines, then returns, the proper configuration for pulsed measurements on the PNA-X ONLY using the spectral nulling technique. The configuration returned needs to be sent to the VNA and any other related external equipment.

The routine will take a desired Pulse Repetition Frequency (PRF) and measurement IFBW and return a possibly modified PRF and IFBW for proper pulsed operation on the VNA.

**VB Syntax**

```
Pulsed.ConfigEnhancedNB (PRF, BW, PhysicalIF, NCO, Stage1TapArray, Stage2TapArray, Stage3TapArray, FixedPRF)
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulsed</td>
<td>(interface) An interface to the agilentpnapulsed.dll application interface.</td>
</tr>
<tr>
<td>PRF</td>
<td>(Double) The Pulse Repetition Frequency.</td>
</tr>
<tr>
<td>[out]</td>
<td>The pulse repetition frequency that has been optimized for use with the VNA. NOTE: This value may be different from the value requested.</td>
</tr>
<tr>
<td>[in]</td>
<td>The desired pulse repetition frequency.</td>
</tr>
<tr>
<td>BW</td>
<td>(Long) The VNA IF Bandwidth.</td>
</tr>
<tr>
<td>[out]</td>
<td>The VNA IF bandwidth that has been optimized for use with the VNA. NOTE: This value may be different from the value requested. Zero (0) is returned if no solution is found for the specified PRF and BW.</td>
</tr>
<tr>
<td>[in]</td>
<td>The desired VNA IF bandwidth.</td>
</tr>
</tbody>
</table>
PhysicalIF
(Double) Returns physical intermediate frequency.

NCO
(Double) Returns numeric controlled oscillator frequency.

Stage1TapArray
(Long array) Returns the stage 1 filter coefficients

Stage2TapArray
(Long array) Returns the stage 2 filter coefficients

Stage3TapArray
(Long array) Returns the stage 3 filter coefficients

FixedPRF
(Boolean)
1 (True) Signals the .DLL routine to NOT adjust the PRF value; rather adjust ONLY the IF Bandwidth. This is the default setting.
0 (False) Adjust both the PRF and IF Bandwidth values as necessary.

Return Type
Not Applicable

Default
Not Applicable

Example
See an example using this command.

C++ Syntax
HRESULT ConfigEnhancedNB(double *pPRF, long *pBW, double *pIF, double *pNCO, double *pStg1, double *pStg2, double *pStg3)

Interface
AgilentPNAPulsed.Application
### CopyNoiseToAllPorts Method

**Description**  
Copies the characterized noise data associated with the specified port, to all the other ports.

**VB Syntax**  
oPorts.CopyNoiseToAllPorts (value)

**Variable**  
<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oPorts</td>
<td>A Ports Collection</td>
</tr>
<tr>
<td>value</td>
<td>(Long) Port number from which noise data is to be copied.</td>
</tr>
</tbody>
</table>

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`oPorts.CopyNoiseToAllPorts(2)`  
See example program

**C++ Syntax**  
HRESULT CopyNoiseToAllPorts([in] long fromPortNumber);  

**Interface**  
IUncertaintyPorts
OBS Create SParameterEX Method

Write-only

About Measurement Parameters

CreateSParameterEx Method - Superseded

Description

Note: This method is replaced by Create SParameter method. The Load port selection is no longer necessary.

Creates a new S-Parameter measurement in an existing or new window and specifies the load port for 3-port devices.

VB Syntax

app.CreateSParameterEx chan,recvr,source[,loadPort][,window]

Variable (Type) - Description

app

(Application (object)

chan

(long integer) - Channel number of the new measurement.

recv

(long integer) - Port number of the test port receiver.

source

(long integer) - Port number of the source.

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source VNA-x model, then you must use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port.

loadPort

(long integer) - Port number of the load. Required for reflection measurements of 3-port devices on multiport VNA models.

window

(long integer) - Optional argument. Choose between 1 and the maximum number of windows allowed on the VNA. See also Traces, Channels, and Windows on the VNA. If unspecified, the S-Parameter will be created in the Active Window.

Return Type

Not Applicable

Default

Not Applicable

Examples

app.CreateSParameterEx 1,2,1,1 'Creates a new S21 measurement in channel 1 and New window(1)
app.CreateSParameterEx 2,1,1,3,1 'Creates a new S11 measurement on channel 2 with port 3 as the load. Create in the active window

C++ Syntax

HRESULT CreateSParameterEx(long ChannelNum, long RcvPort, long SrcPort, long LoadPort, long windowNumber)

Interface

IApplication
### OBS_CreateCustomMeasurement Method

**VB Syntax**

```vbnet
app.CreateCustomMeasurement chanNum,guid[,window]
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>app</strong></td>
<td>(object)</td>
<td>An Application object</td>
</tr>
<tr>
<td><strong>chanNum</strong></td>
<td>(long)</td>
<td>Channel number used by the new measurement; can exist or be a new channel.</td>
</tr>
<tr>
<td><strong>guid</strong></td>
<td>(string)</td>
<td>the GUID (Globally Unique IDentifier) of the new custom measurement object. The new custom measurement must be installed and registered on the PNA. Should be in “registry format”. See example below.</td>
</tr>
<tr>
<td><strong>window</strong></td>
<td>(long)</td>
<td>Optional argument. Number of the window the new custom measurement will be placed in. Choose 1 to 16. If unspecified, the measurement is placed in the active window.</td>
</tr>
</tbody>
</table>

**Return Type**

Not Applicable

**Default**

Not Applicable

**Examples**

```vbnet
app.CreateCustomMeasurement 1, "\{12345678-56D3-11D5-AD50-00108334AE98\}
'Not an actual custom measurement - for example purpose only
```

**C++ Syntax**

```cpp
HRESULT CreateCustomMeasurement (long ChannelNum, BSTR guid, long windowNumber)
```

**Interface**

IApplication
## OBS_DataToDivisor Method

**Write-only**

**About Receiver Cal**

**DataToDivisor Method**  **Superseded**

### Description

Note: This property is replaced by DoReceiverPowerCal Method.

Stores the measurement's data to the measurement's "divisor" buffer for use by the Normalization data processing algorithm. Normalization is currently supported only on measurements of unratioed power, for purpose of receiver power calibration. If DataToDivisor is called on a ratioed measurement (such as an S-parameter), it will return an error.

### VB Syntax

```
meas.DataToDivisor
```

### Variable (Type) - Description

- `meas` (object) - A Measurement object

### Return Type

Not Applicable

### Default

Not Applicable

### Examples

```
meas.DataToDivisor
```

### C++ Syntax

```
HRESULT DataToDivisor();
```

### Interface

```
IMeasurement
```
## DeleteRange Method

**Description**
Deletes the specified frequency range. On the Measurement Setup dialog this is the Remove setting.

**VB Syntax**
DIQ.DeleteRange (n)

**Variable**
- **(Type) - Description**
- **DIQ** - A DIQ Object
- **n** - (Long) Frequency range number to delete.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vbnet
diq.DeleteRange 2 'removes frequency range 2
See example program
```

**C++ Syntax**
HRESULT DeleteRange(long range);

**Interface**
IDIQ
## DiscreteGetMaxPower Method

**Description**

Returns a single max leveled power value (in dBm) indicating the most restrictive maximum for all discrete maximum powers (the minimum of all maximum leveled powers).

\[\text{maxPower} = \text{powerRange}.\text{DiscreteGetMaxPower}\]

**VB Syntax**

```vbnet
maxPower = powerRange.DiscreteGetMaxPower
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>powerRange</code></td>
<td>A <code>PowerRange</code> (object)</td>
</tr>
<tr>
<td><code>maxPower</code></td>
<td>(Double) Variable to store the most restrictive maximum power level in dBm.</td>
</tr>
</tbody>
</table>

**Return Type**

(Double)

**Default**

Not Applicable

```csharp
maxPower = powerRange.DiscreteGetMaxPower
```

**C++ Syntax**

```c++
HRESULT get_DiscreteGetMaxPower(double* pMax);
```

**Interface**

`IPowreRange`
DiscreteGetMaxPowerArray Method

Description
Returns an array of max leveled power values (in dBm), where each element corresponds to the maximum leveled power possible for CW stimulus at the corresponding frequency set by the DiscreteFrequencies property.

VB Syntax
maxPower = powerRange.DiscreteGetMaxPowerArray

Variable
(powerRange) - Description
powerRange
A PowerRange (object)

maxPower
(Variant array) Variable to store the list of maximum leveled power values in dBm.

Return Type
Variant array

Default
Not Applicable

C++ Syntax
HRESULT get_DiscreteGetMaxPowerArray(VARIANT* pMax);

Interface
IPowreRange
DiscreteGetMinPower Method

Description
Returns a single minimum power value (in dBm) indicating the most restrictive minimum for all discrete minimum powers (the maximum of all minimum powers).

VB Syntax
minPower = powerRange.DiscreteGetMinPower

Variable
(Type) - Description

powerRange
A PowerRange (object)

minPower
(Double) Variable to store the most restrictive minimum power level in dBm.

Return Type
(Double)

Default
Not Applicable

C++ Syntax
HRESULT get_DiscreteGetMinPower(double* pMin);

Interface
IPowreRange
### DiscreteGetMinPowerArray Method

**Description**

Returns an array of minimum power values (in dBm), where each element corresponds to the minimum power possible for CW stimulus at the corresponding frequency set by the `DiscreteFrequencies` property.

**VB Syntax**

```
minPower = powerRange.DiscreteGetMinPowerArray
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>powerRange</code></td>
<td>A <code>PowerRange</code> (object)</td>
</tr>
<tr>
<td><code>minPower</code></td>
<td>(Variant array) Variable to store the list of minimum power values in dBm.</td>
</tr>
</tbody>
</table>

**Return Type**

Variant array

**Default**

Not Applicable

**C++ Syntax**

```
HRESULT get_DiscreteGetMinPowerArray(VARIANT* pMin);
```

**Interface**

`IPowreRange`
OBS_DoECAL1Port Method

Does a 1-Port calibration using an ECAL module. You must first have a 1-port measurement active to perform the calibration.

The characterization within the ECal module that will be used for the calibration is specified by the ECALCharacterization property on the ICalibrator2 interface. The default value of the ECALCharacterization property is naECALFactoryCharacterization.

VB Syntax

cal.DoECAL1Port [port][,module]

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cal</td>
<td>A Calibrator (object)</td>
<td></td>
</tr>
<tr>
<td>port</td>
<td>(long integer) Optional argument - Port number to calibrate. Choose from:</td>
<td>1 - Calibrate port 1 (default if unspecified)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - Calibrate port 2</td>
</tr>
<tr>
<td>module</td>
<td>(enum NAECalModule)</td>
<td>Optional argument - ECAL module. Choose from:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 - naECALModule_A - (default if unspecified)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - naECALModule_B</td>
</tr>
</tbody>
</table>

Return Type

None

Default

Not Applicable

Examples

cal.DoECAL1Port,2,naECALModule_B

C++ Syntax

HRESULT DoECAL1Port(long port, tagNAECALModule ecalModule);

Interface

ICalibrator
OBS_DoECAL2Port Method

Does a 2-Port calibration using an ECAL module. You must first have a 2-port measurement active to perform the calibration.

The characterization within the ECal module that will be used for the calibration is specified by the ECALCharacterization property on the ICalibrator2 interface. The default value of the ECALCharacterization property is naECALFactoryCharacterization.

VB Syntax

```
cal.DoECAL2Port(portA[,portB][,module])
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cal</td>
<td>A Calibrator (object)</td>
</tr>
<tr>
<td>portA</td>
<td>(long integer) Optional argument - Number of the receive port to calibrate.</td>
</tr>
<tr>
<td></td>
<td>Choose from:</td>
</tr>
<tr>
<td></td>
<td>1 - Calibrate port 1 (default, if unspecified)</td>
</tr>
<tr>
<td></td>
<td>2 - Calibrate port 2</td>
</tr>
<tr>
<td></td>
<td>3 - Calibrate port 3 (if the VNA has 3 ports)</td>
</tr>
<tr>
<td>portB</td>
<td>(long integer) Optional argument - Number of the source port to calibrate.</td>
</tr>
<tr>
<td></td>
<td>Choose from:</td>
</tr>
<tr>
<td></td>
<td>1 - Calibrate port 1 (default, if unspecified)</td>
</tr>
<tr>
<td></td>
<td>2 - Calibrate port 2</td>
</tr>
<tr>
<td></td>
<td>3 - Calibrate port 3 (if the VNA has 3 ports)</td>
</tr>
<tr>
<td>module</td>
<td>(enum NAECALModule) – Optional argument. ECal module.</td>
</tr>
<tr>
<td></td>
<td>Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naECALModule_A (default, if unspecified)</td>
</tr>
<tr>
<td></td>
<td>1 - naECALModule_B</td>
</tr>
</tbody>
</table>

Return Type
None

Default
Not Applicable

Examples
```
cal.DoECAL2Port,1,2,naECALModule_B
```
C++ Syntax

HRESULT DoECAL2Port(long rcvport, long srcPort, tagNAECALModule ecalModule);

Interface

ICalibrator
OBS_Get ECAL Module Info Method

Description

Note: This property is replaced by GetECALModuleInfoEx Method

Returns the following information about the connected ECAL module: model number, serial number, connector type, calibration date, min and max frequency.

The characterization within the ECAL module that this information will be read from is specified by the ECALCharacterization property on the ICalibrator2 interface. The default value of the ECALCharacterization property is naECALFactoryCharacterization.

VB Syntax

moduleInfo = cal.GetECALModuleInfo (module)

Variable

moduleInfo (string) - variable to store the module information

cal A Calibrator (object)

module (enum NAECALModule) – ECAL module. Choose from:

0 - naECALModule_A

1 - naECALModule_B

Return Type

String

Default

Not Applicable

Examples

info = cal.GetECALModuleInfo(naECALModule_A)

Example return string:

ModelNumber: 85092–60007, SerialNumber: 01386, ConnectorType: N5FN5F, PortAConnector: Type N (50) female, PortBConnector: Type N (50) female, MinFreq: 30000, MaxFreq: 9100000000, NumberOfPoints: 250, Calibrated: July 4 2002

C++ Syntax

HRESULT GetECALModuleInfo(tagNAECALModule ecalModule, BSTR* info);

Interface

ICalibrator
**OBSget InputVoltage Method**

**VB Syntax**

\[ volts = AuxIO.get_InputVoltage \]

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volts</td>
<td>(double) - variable to store the return value</td>
</tr>
<tr>
<td>AuxIO</td>
<td>(object) - A Hardware Auxiliary Input / Output object</td>
</tr>
</tbody>
</table>

**Return Type**

Double

**Default**

0

**Examples**

```vba
Dim aux as HWAuxIO
Set aux = PNA.getAuxIO
volts = aux.get_InputVoltage 'read voltage on Analog In (pin 14)
```

**C++ Syntax**

```cpp
HRESULT get_InputVoltage ( double* Voltage );
```

**Interface**

HWAuxIO
# OBS_get SourcePowerCalData Method

**Read-only**

**About Source Power Cal**

### getSourcePowerCalData Method - Superseded

- **Description**
  Retrieves (as variant data type) requested source power calibration data, if it exists, from this channel.

  *Note: This method is replaced by GetSourcePowerCalDataEx Method.*

  *Note: This method returns a variant which is less efficient than methods available on the ISourcePowerCalData interface*

- **VB Syntax**
  
  ```vbnet
  data = chan.getSourcePowerCalData sourcePort
  ```

- **Variable**
  
  - **(Type) - Description**
  
    - **data**
      (variant) – Array to store the data.
    
    - **chan**
      (object) – A Channel object
    
    - **sourcePort**
      (long integer) – The source port for which calibration data is being requested.

- **Return Type**
  Variant array – automatically dimensioned to the size of the data.

- **Default**
  Not Applicable

- **Examples**
  ```vbnet
  Dim varData As Variant
  Const port1 As Long = 1
  varData = chan.getSourcePowerCalData port1
  'Print the data
  For i = 0 To chan.NumberOfPoints - 1
     Print varData(i)
  Next i
  ```

- **C++ Syntax**
  ```cpp
  HRESULT getSourcePowerCalData(long sourcePort, VARIANT *pData);
  ```

- **Interface**
  IChannel
**OBS_get SourcePowerCalDataScalar Method**

**Read-only**

**getSourcePowerCalDataScalar Method - Superseded**

**Description**

Retrieves (as scalar values) requested source power calibration data, if it exists, from this channel.

*Note: This method is replaced by GetSourcePowerCalDataScalarEx Method*

**VB Syntax**

chandata.getSourcePowerCalDataScalar sourcePort, numValues, data

**Variable**

- **chandata** *(interface)* — An ISourcePowerCalData interface pointing to a Channel (object)
- **sourcePort** *(long integer)* — The source port for which calibration data is being requested.
- **numValues** *(long integer)* — Number of data values.
  - **[out]** specifies number of data values returned.
  - **[in]** specifies number of values being requested (this must not be larger than the capacity of the data array).
- **data** *(single)* — Array to store the data.

**Return Type**

Single

**Default**

Not Applicable

**Examples**

```vbnet
dim numValues as long
Dim scalarCalValues() As Single
Dim chanData As ISourcePowerCalData
Const port1 As Long = 1
numValues = app.ActiveChannel.NumberOfPoints
ReDim scalarCalValues(numValues)
Set chanData = app.ActiveChannel

chanData.getSourcePowerCalDataScalar port1, numValues, scalarCalValues(0)

'Print the data
For i = 0 to numValues - 1
print scalarCalValues(i)
next i
```

**C++ Syntax**

```cpp
HRESULT getSourcePowerCalDataScalar(long sourcePort, long *pNumValues, float *pVals);
```
OBS_Get Standard Method

Description

This command has been replaced with Get StandardByString

Retrieves data that was acquired for a specific cal standard. This method returns the actual measurement data - not the calculated error terms.

This method returns a variant which is less efficient than getStandardComplex on the ICalData interface.

Learn about reading and writing Calibration data.

VB Syntax

data = cal.getStandardClass,rcv,srcclass

Variable (Type) - Description

data

Variant array to store the data.

cal

A Calibrator (object)

class (enum NACalClass) Standard to be measured. Choose from:

1 - naClassA
2 - naClassB
3 - naClassC
4 - naClassD
5 - naClassE
6 - naReferenceRatioLine
7 - naReferenceRatioThru

SOLT Standards
1 - naSOLT_Open
2 - naSOLT_Short
3 - naSOLT_Load
4 - naSOLT_Thru
5 - naSOLT_Isolation

TRL Standards
1 - naTRL_Reflection
2 - naTRL_Line_Reflection
3 - naTRL_Line_Tracking
4 - naTRL_Thru
5 - naTRL_Isolation

rcv

(long integer) - Receiver Port

src

(long integer) - Source Port

Return Type

(variant) - two-dimensional array (0:1, 0:NumberOfPoints-1)
<table>
<thead>
<tr>
<th>Default</th>
<th>Not Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Examples</strong></td>
<td></td>
</tr>
<tr>
<td>Dim varStd As Variant</td>
<td></td>
</tr>
<tr>
<td>varStd = cal.getStandard(naSOLT_Thru, 2, 1)</td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td></td>
</tr>
<tr>
<td>HRESULT raw_getStandard(tagNACalClass stdclass, long ReceivePort, long SourcePort, VARIANT* pData)</td>
<td></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td></td>
</tr>
<tr>
<td>ICalibrator</td>
<td></td>
</tr>
</tbody>
</table>
GetSnPData Method

Returns noise parameter and S-parameter S2P data for vector noise figure measurements. Noise parameters are NOT valid for NFX or Scalar noise figure measurements.

**Data = nf.GetSnPData**

**variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Variant array to store the data.</td>
</tr>
<tr>
<td>nf</td>
<td>A NoiseFigure (object)</td>
</tr>
<tr>
<td>type</td>
<td>(string) - Choose &quot;NoiseParameter&quot; - Noise parameter data.</td>
</tr>
</tbody>
</table>

**Return Type**

Variant – 3 dimensional array.

First dimension size is number of parameters returned

Second dimension size is number of points in the channel.

Third dimension size is 2; format is specified with SnPFormat Property

Data is returned in this order:

<real S11> <imag S11> <real S21> <imag S21> <real S12> <imag S12> <real S22> <imag S22>

Then noise parameters:

<NFMin dB> <mag GammaOpt> <phase GammaOpt> <Rn/Z0>

**Default**

Not Applicable

**Examples**

```vbnet
snp = nf.GetSnPData("NoiseParameter")
```

**C++ Syntax**

```c
HRESULT GetSnPData( BSTR type, VARIANT * response)
```

**Interface**

INoiseFigure7
# InitiateCableCharacterization Method

**Description**
Initializes a cable repeatability characterization for the specified channel and port.

**VB Syntax**

```vbnet
UncertChar.InitiateCableCharacterization (chan, port, iterations)
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UncertChar</td>
<td>A Characterizer (object)</td>
<td></td>
</tr>
<tr>
<td>chan</td>
<td>(Long)</td>
<td>Channel number to calibrate.</td>
</tr>
<tr>
<td>port</td>
<td>(Long)</td>
<td>Port number to use in the characterization.</td>
</tr>
<tr>
<td>iterations</td>
<td>(Long)</td>
<td>Number of Iterative connections of the standards to be measured for the characterization.</td>
</tr>
</tbody>
</table>

**Note:** chan MUST be the active channel.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```vbnet
oUncertChar.InitiateCableCharacterization(2,2,20)
```

**C++ Syntax**

```csharp
HRESULT InitiateCableCharacterization([in] long channel, [in] long portNum, [in] long numIterationsPerStep, [out,retval] long* pNumSteps);
```

**Interface**

<table>
<thead>
<tr>
<th>Interface</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IUncertaintyCharacterizer</td>
<td></td>
</tr>
</tbody>
</table>

The following existing commands on the GuidedCal object are used to perform the initialized cable repeatability characterization:

- GenerateSteps Method
- Get StepDescription Method
- MinimumIterationsForStep Property
- AcquireStep Method
- IterationCountForStep Property
- ResetStep Method
- Save (CalSet) Method
- Abort Method
# InitiateNoiseCharacterization Method

**Description**
Initializes a noise characterization for the specified channel and port.

**VB Syntax**
```
UncertChar.InitiateNoiseCharacterization (chan, firstPort, secondPort, iterations)
```

**Variable**
<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UncertChar</td>
<td>A Characterizer (object)</td>
<td></td>
</tr>
<tr>
<td>chan</td>
<td>(Long)</td>
<td>Channel number to calibrate.</td>
</tr>
<tr>
<td>firstPort</td>
<td>(Long)</td>
<td>First VNA port number on which the noise characterization is to be performed.</td>
</tr>
<tr>
<td>secondPort</td>
<td>(Long)</td>
<td>Second VNA port number on which the noise characterization is to be performed.</td>
</tr>
<tr>
<td>iterations</td>
<td>(Long)</td>
<td>Number of Iterative connections of the standards to be measured for the characterization.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
UncertChar.InitiateNoiseCharacterization(2,1,2,20)
```

**C++ Syntax**
```
```

**Interface**
IUncertaintyCharacterizer

The following existing commands on the GuidedCal object are used to perform the initialized noise characterization:

- GenerateSteps Method
- Get StepDescription Method
- MinimumIterationsForStep Property
- AcquireStep Method
- IterationCountForStep Property
- ResetStep Method
- Save (CalSet) Method
- Abort Method
Load Method

Recalls the parameters and/or frequency ranges that you previously defined and saved.

**VB Syntax**

```
DIQ.Load (name, type)
```

**Variable**

(Type) - Description

A DIQ Object

**name**

(String) Full path (optional) and filename with or without the *.xml extension. If the full path is not provided, the file is loaded from D:\.

**type**

((Enum NADIQSaveLoadType) Choose the type of settings to be recalled:

- naParameterList - just the parameters.
- naFrequencyRange - just the frequency settings.
- naALL - both parameters and frequency settings.

**Return Type**

Not Applicable

**Default**

naParameterList

**Examples**

```
Diq.Load "myParams",naParameterList
Diq.Load "D:\myParams.xml",naALL
```

**C++ Syntax**

```
HRESULT Load(BSTR name,tag NADIQSaveLoadType type);
```

**Interface**

IDIQ2
**OBS_put SourcePowerCalData Method**

*Write-only*

About Source Power Cal

**putSourcePowerCalData Method**  Superseded

<table>
<thead>
<tr>
<th>Description</th>
<th>Inputs source power calibration data (as variant data type) to this channel for a specific source port.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>This method is Superseded. It is replaced by <a href="#">PutSourcePowerCalDataEx Method</a>.</td>
</tr>
</tbody>
</table>

**VB Syntax**

```vbnet
chan.putSourcePowerCalData sourcePort, data
```

**Variable**

- **chan** (object) – A Channel object
- **sourcePort** (long integer) – The source port for which calibration data is being requested.
- **data** (variant) – Array of source power cal data being input.

**Return Type**

None

**Default**

Not Applicable

**Examples**

```vbnet
chan.putSourcePowerCalData 1, varData
```

**C++ Syntax**

```cpp
HRESULT putSourcePowerCalData(long sourcePort, VARIANT varData);
```

**Interface**

IChannel
OBS_put SourcePowerCalDataScalar Method

Write-only.

About Source Power Cal

putSourcePowerCalDataScalar Method - Superseded

Description

Inputs source power calibration data (as scalar values) to this channel for a specific source port.

VB Syntax

chandata.putSourcePowerCalDataScalar sourcePort, numValues, data

Variable

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chandata</td>
<td>(interface)</td>
<td>An ISourcePowerCalData interface pointing to a Channel (object)</td>
</tr>
<tr>
<td>sourcePort</td>
<td>(long integer)</td>
<td>The source port for which calibration data is being input.</td>
</tr>
<tr>
<td>numValues</td>
<td>(long integer)</td>
<td>Number of data values being input.</td>
</tr>
<tr>
<td>data</td>
<td>(single)</td>
<td>Array of source power cal data being input.</td>
</tr>
</tbody>
</table>

Return Type

None

Default

Not Applicable

Examples

Dim chanData As ISourcePowerCalData
Set chanData = app.ActiveChannel
chanData.putSourcePowerCalDataScalar 1, 201, scalarCalValues(0)

C++ Syntax

HRESULT putSourcePowerCalDataScalar(long sourcePort, long numValues, float *pVals);

Interface

ISourcePowerCalData

Note: This method is replaced by PutSourcePowerCalDataScalarEx Method

Note: If this does not equal the current number of points on the channel, the calibration will not be valid.
**OBS_Put Standard Method - Superseded**

**Description**

Note: This command is replaced by **PutStandardByString**

Writes variant data to the error correction buffer holding measurement data acquired for a specific standard.

Learn about reading and writing Calibration data.

**VB Syntax**

```vbnet
cal.putStandard class,rcv,src,data
```

**Variable**

**(Type) - Description**

- **cal**
  - A Calibrator (object)

- **class**
  - (enum NACalClass) Standard. Choose from:
    1 - naClassA
    2 - naClassB
    3 - naClassC
    4 - naClassD
    5 - naClassE
    6 - naReferenceRatioLine
    7 - naReferenceRatioThru

  SOLT Standards
    1 - naSOLT_Open
    2 - naSOLT_Short
    3 - naSOLT_Load
    4 - naSOLT_Thru
    5 - naSOLT_Isolation

  TRL Standards
    1 - naTRL_Reflection
    2 - naTRL_Line_Reflection
    3 - naTRL_Line_Tracking
    4 - naTRL_Thru
    5 - naTRL_Isolation

- **rcv**
  - (long) - Receiver Port

- **src**
  - (long) - Source Port
<table>
<thead>
<tr>
<th>data</th>
<th>(variant) Two dimensional array (0:1, 0:points-1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>Dim varStd (1,200) As Variant cal.putStandard naSOLT_Thru, 2, 1, varStd</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT raw_putStandard(tagNACalClass stdclass, long ReceivePort, long SourcePort, VARIANT varData) ICalibrator</td>
</tr>
</tbody>
</table>
RangeGetMaxPower Method

Description
Returns the minimum of all max leveled power values from RangeStartFrequency to RangeStopFrequency (inclusive).

VB Syntax
maxPower = powerRange.RangeGetMaxPower

Variable (Type) - Description

    powerRange
    A PowerRange (object)

    maxPower
    (Double) Variable to store the power level in dBm.

Return Type (Double)
Default Not Applicable

C++ Syntax
HRESULT get_RangeGetMaxPower(double* pMax);

Interface IPowreRange
# RangeGetMinPower Method

**Description**

Returns the maximum of all minimum power values from `RangeStartFrequency` to `RangeStopFrequency` (inclusive).

**VB Syntax**

```vbnet
minPower = powerRange.RangeGetMinPower
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>minPower</code></td>
<td>(Double) Variable to store the power level in dBm.</td>
</tr>
<tr>
<td><code>powerRange</code></td>
<td>A <code>PowerRange</code> (object)</td>
</tr>
<tr>
<td><code>returnType</code></td>
<td>(Double)</td>
</tr>
<tr>
<td><code>default</code></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**C++ Syntax**

```cpp
HRESULT get_RangeGetMinPower(double* pMin);
```

**Interface**

`IPowreRange`
## ResetStep Method

**Description**
Resets the specified guided cal connection step as unmeasured. This clears all previous measurements made for that step.

**VB Syntax**
guided.ResetStep (n)

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guided</td>
<td><strong>GuidedCalibration (object)</strong></td>
</tr>
<tr>
<td>n</td>
<td>Guided Cal step number to reset as unmeasured. Use <strong>GenerateSteps Method</strong> to read the number of steps in the calibration.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```vbnet
obj.ResetStep(4)
```

**C++ Syntax**

```c++
HRESULT ResetStep(long step);
```

**Interface**
IGuidedCalibration10
### Save diq Method

**Description**
Stores the list of parameters and frequency range settings to an *.xml file for recall at a later time.

Use **Load Method** to recall files.

**Note:** The Frequency Range settings and the **DIQ Parameters** are saved and recalled from a single *.xml file.

**VB Syntax**

`DIQ.Save (filename)`

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DIQ</code></td>
<td>A <strong>DIQ Object</strong></td>
</tr>
<tr>
<td><code>filename</code></td>
<td>(String) Full path (optional) and filename with or without the *.xml extension. If the full path is not provided, the file is saved to the D:\ drive.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```vbnet
Diq.Save "myDIQfile"
or
Diq.Save "D:\myDIQfile.xml"
```

**C++ Syntax**

```cpp
HRESULT Save(BSTR filename);
```

**Interface**

`IDIQ2`
## SelectCableForAllPorts Method

**Description**
Selects the name of the cable to be associated with all the ports currently enabled on the VNA.

### VB Syntax
```
oPorts.SelectCableForAllPorts (value)
```

### Variable
- `value` (String) Name of the cable.

### Examples
```
oPorts.SelectCableForAllPorts("myCable")
```

### C++ Syntax
```
HRESULT SelectCableForAllPorts([in] BSTR cableName);
```

### Interface
`IUncertaintyPorts`
## OBS_SetCalInfo (power) Method

**Write-only**

**About Source Power Cal**

### SetCalInfo Method (for source power cals) - Superseded

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>This command is replaced by <strong>SetCalInfoEx Method</strong></td>
</tr>
</tbody>
</table>

Specifies the technique to be used for the source power calibration about to be performed, and the channel and source port for which it is to be performed.

### VB Syntax

```vbnet
powerCalibrator.SetCalInfo calMethod, channel, sourcePort, calPower
```

### Variable (Type) - Description

- **powerCalibrator**
  - (object) - A SourcePowerCalibrator object

- **calMethod**
  - (enum NASourcePowerCalMethod) The method of gathering the source power correction data.

- **channel**
  - (long integer) - Number of the VNA channel (not power meter channel) on which the source power cal will be performed. If the channel doesn’t already exist, it will be created.

- **sourcePort**
  - (long integer) - Port number on which the source power cal will be performed.

- **calPower**
  - (double) - Specifies the power level that is expected at the desired reference plane (input or output of DUT) following the source power cal.

### Return Type

None

### Default

Not Applicable

### Examples

```vbnet
powerCalibrator.SetCalInfo naPowerMeter, 1, 1, -10
```

### C++ Syntax

```cpp
HRESULT SetCalInfo(tagNASourcePowerCalMethod calMethod, long channel, long sourcePort, double calPower);
```

### Interface

ISourcePowerCalibrator
# OBS SetCalInfo2 Method

**Write-only**

### About Source Power Cal

**SetCalInfo2 Method (for source power cals) - Superseded**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>This command is replaced by <a href="#">SetCalInfoEx Method</a></td>
</tr>
</tbody>
</table>

Specifies the technique, the channel, and the source port to be used for the source power calibration about to be performed.

### VB Syntax

```vbnet
powerCalibrator.SetCalInfo2 calMethod, channel, sourcePort, [powerOffset,] [display]
```

### Variable (Type) - Description

<table>
<thead>
<tr>
<th>powerCalibrator</th>
<th>(object) - A <code>SourcePowerCalibrator3</code> object</th>
</tr>
</thead>
<tbody>
<tr>
<td>calMethod</td>
<td>(enum NASourcePowerCalMethod) Selects the calibration method to be used for the source power cal acquisition.</td>
</tr>
<tr>
<td>channel</td>
<td>(long integer) - Number of the VNA channel (not power meter channel) on which the source power cal will be performed. If the channel does not already exist, it will be created.</td>
</tr>
<tr>
<td>sourcePort</td>
<td>(long integer) - Port number on which the source power cal will be performed.</td>
</tr>
<tr>
<td>[powerOffset]</td>
<td>(double) - Optional argument. Sets or returns a power level offset from the VNA test port power. This can be a gain or loss value (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier at the input of your DUT. Following the calibration, the VNA power readouts are adjusted by this value. This argument performs the same function as <code>chan.SourcePowerCalPowerOffset</code> Property</td>
</tr>
<tr>
<td>[display]</td>
<td>(boolean) Optional argument. Enables and disables the display of power readings on the VNA screen. After the source power cal data is acquired, this setting is reset to ON. If unspecified, value is set to ON.</td>
</tr>
</tbody>
</table>

| True | Display of power readings is ON |
| False| Display of power readings is OFF |

### Return Type

None

### Default

Not Applicable

### Examples

```vbnet
powerCalibrator.SetCalInfo2 naPowerMeter, 1, 1, -10, True
```

### C++ Syntax

```cpp
HRESULT SetCalInfo2( tagNASourcePowerCalMethod enumCalMethod, long Channel, long SourcePort, double PowerOffset = 0., VARIANT_BOOL bDisplay = VARIANT_TRUE);
```
Interface

ISourcePowerCalibrator3
## UnselectCalset Method

**Description**
Unselects a Cal Set from the channel.

**VB Syntax**
```vbnet
channel.UnselectCalSet
```

**Variable**
(Type) - Description

- `channel`
  
  (object) - A `Channel` object

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**
```vbnet
channel.UnselectCalSet
```

**C++ Syntax**
```cpp
HRESULT UnselectCalSet();
```

**Interface**
`IChannel24`
Cable Object

Description

Provides access to the properties and methods that are used to reset data associated with the cable object.

Accessing the Cable object

Get a handle to the Cable object through the Cables Collection.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim oCables as UncertaintyManager
Set oCables = app.UncertaintyManager.Cables

Dim oCable1 as Cables(1)
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
- Superseded commands

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>See History</td>
</tr>
<tr>
<td>ResetRepeatability</td>
<td>IUncertaintyCable</td>
<td>Resets (clears) the repeatability data associated with the cable object.</td>
</tr>
</tbody>
</table>

Properties

- **Name**
  - IUncertaintyCable
    - Return the cable name.

IUncertaintyCable History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IUncertaintyCable</td>
<td>10.40</td>
</tr>
</tbody>
</table>
Cables Collection

Description

Child of the UncertaintyManager Object. A collection that provides a mechanism for iterating through the Cable objects.

Accessing the Cables collection

Get a handle to an individual cable by specifying an item of the Cables collection.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim cabs As Cables
Set cabs = app.UncertaintyManager.Cables(1)
```

See Also:
- Cable Object
- Collections in the Analyzer
- The VNA Object Model
- Limit Line Testing Example

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>Get a handle by number to a cable in the cables collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of cables in the collection.</td>
</tr>
</tbody>
</table>

IUncertaintyCables History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IUncertaintyCables</td>
<td>10.40</td>
</tr>
</tbody>
</table>
Characterizer Object

Description

Provides access to the properties and methods that are used to characterize Noise and Cables for the Dynamic Uncertainty application.

Accessing the Characterizer object

Get a handle to the Characterizer object.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim oChar as Characterizer
Set oChar = app.UncertaintyManager.Characterizer
```

See Also:

- VNA Automation Interfaces
- The VNA Object Model
- Example Programs
- Superseded commands

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InitiateCableCharacterization</td>
<td>IUncertaintyCharacterizer</td>
<td>Initiates a Cable Repeatability characterization</td>
</tr>
<tr>
<td>InitiateNoiseCharacterization</td>
<td>IUncertaintyCharacterizer</td>
<td>Initiates a Noise characterization</td>
</tr>
</tbody>
</table>

Properties

| GuidedCalibration            | IUncertaintyCharacterizer     | Returns a handle in order to perform a Guided Calibration.      |

IUncertaintyManager History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IUncertaintyManager</td>
<td>10.40</td>
</tr>
</tbody>
</table>
DIQMeasurement Object

Description

Controls the Differential I/Q application settings.

Note: There are TWO objects for making Differential IQ settings.

See the DIQ Object

Accessing the Diff IQ and Diff IQMeas objects

```vba
Dim app as AgilentPNA835x.Application
app.CreateCustomMeasurementEx 2, "Differential I/Q", "IPwrF1"
Dim DIQ
Set DIQ = app.ActiveChannel.CustomChannelConfiguration
Dim DiqMeas
Set DiqMeas = app.ActiveMeasurement.CustomMeasurementConfiguration
```

See Also:

Example program
- Differential I/Q application
- VNA Automation Interfaces
- The VNA Object Model

DIQ Select X-Axis

- ActiveXAxis Property
- XAxisDomain Property
- XAxisSource Property

DIQMeas History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with VNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDIQMeas</td>
<td>10.25</td>
</tr>
</tbody>
</table>
ActiveXAxis Property

Description
Sets the information to display on the X-axis of the selected DIQ measurement. This command does not change the default setting for new traces.

See Also:
XAxisSource Property
XAxisDomain Property

VB Syntax

diqMeas.ActiveXAxis (domain) = source

Variable

(Type) - Description

diqMeas
A DIQMeas (object)

domain
(Enum as NAAxisDomainType) - Domain to display on the X-axis. Choose from:
0 – naFrequency - display the primary frequency range.
   1 – naPower - display the power sweep range
   2 – naPhase - display the phase sweep range
   3 – naDCValue - display the DC sweep range
   4 – naPoints - display the data points in the range

source
(String) Specific source for the selected domain.

Return Type
Enum

Default
0 – naFrequency

Examples

diqMeas.ActiveXAxis(1) = "Port 2" 'Write
variable = diqMeas.ActiveXAxis(1) 'Read

C++ Syntax

HRESULT put_ActiveXAxis(tagNAAxisDomainType domainType, BSTR source);

Interface
IDifferentialIQMeas
AutoRangeState Property

Description
Sets and reads the ON/ OFF state of auto range source attenuation.

On the Source Configuration dialog under Power, this is the Auto range source attenuator setting.

VB Syntax
DIQ.AutoRangeState (port) = value

Variable
(Type) - Description
DIQ
A Differential I/Q (object)
port
(String) Source port name. Use SourcePortNames to read a list of valid source ports.
value
(Boolean) Choose from:
ON or 1 - Auto range the source attenuation.
OFF or 0 - Do not Auto range the source attenuation.

Return Type
Boolean

Default
ON

Examples

diq.AutoRangeState = True 'Write

value = diq.AutoRangeState 'Read

C++ Syntax
HRESULT get_AutoRangeState(BSTR port, BOOL* AutoRangeState);
HRESULT put_AutoRangeState(BSTR port, BOOL AutoRangeState);

Interface
IDifferentialIQ
BalPort1PowerOffset Property

Sets and returns the power offset for Balanced port 1. \texttt{balStim.Mode} must be set to a True Stimulus mode. Applicable only with \texttt{Opt S93460A/B - iTMSA}.

**VB Syntax**

\texttt{balStim.BalPort1PowerOffset = value}

**Variable**

\texttt{balStim} \hspace{1cm} \text{A \textit{BalancedStimulus} (object)}

\texttt{value} \hspace{1cm} \text{(Double) - Power Offset in dB.}

**Return Type**

\texttt{Double}

**Default**

0

**Examples**

\texttt{balStim.BalPort1PowerOffset = 2 } \texttt{'Write}
\texttt{variable = balStim.BalPort1PowerOffset \hspace{1cm} 'Read}

**C++ Syntax**

\texttt{HRESULT get_BalPort1PowerOffset (double *pVal)}
\texttt{HRESULT put_BalPort1PowerOffset (double newVal)}

**Interface**

\texttt{IBalancedStimulus}
CalibrateAllChannelsEx Property

Description
Perform multiple instances of CalibrateAllChannels. You can assign a single client channel to multiple cal all instances. However, care should be taken here. If you create user calsets, all calsets will be created but only the last one will be applied. If you are only using a cal register, only the last cal all will be written to the cal register (only supports one calibration). The order that the cal alls are created should be the order in which they are saved (GenerateErrorTerms). This ensures that the client channel imports the proper ETerms.

VB Syntax
var = mgr.CalibrateAllChannelsEx (calAllNumber)

Variable
mgr
A CalManager (object)

var
Variable to store the returned calibration number.

calAllNumber
(Double) Calibration number.

Return Type
Not Applicable

Default
1

Examples
See the programming example.

C++ Syntax
HRESULT get_CalibrateAllChannelsEx(VARIANT* calAllNumber);

Interface
ICalManager13
CalKitType PhaseRef Property

Description
Sets and returns the Cal Kit to be used during the S-parameter portion of a Phase Reference calibration.

VB Syntax
phaseRef.CalKitType = value

Variable
(object) - Description

object
Any of the following:
PhaseReferenceCalibration (object)

value
(string) - Calibration Kit type. Case-sensitive. Use GetCompatibleCalKits to return a list of valid Cal Kits.

Return Type
String

Default
Not Applicable

Examples
phaseRef.CalKitType = "85052C"

C++ Syntax
HRESULT get_CalKitType(BSTR *calkit)
HRESULT put_CalKitType(BSTR calkit)

Interface
IPhaseReference2
### ChannelClients Property

**Description**
Returns the numbers of the channels using the calset.

**VB Syntax**

```vbnet
value = calSet.ChannelClients
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>calSet</td>
<td>A <code>CalSet</code> (object)</td>
</tr>
</tbody>
</table>

**Return Type**

1-dimensional variant array

**Default**
Not Applicable

**Examples**

```vbnet
Set app = CreateObject("AgilentPNA835x.Application")
Set calset = app.GetCalManager.CalSets.Item(2)
arry = calSet.ChannelClients
For i = 0 to UBound(arry)
    MsgBox (arry(i))
Next
```

**C++ Syntax**

```cpp
HRESULT get_ChannelClients (VARIANT* vals)
```

**Interface**

`ICalSet8`
## ContentDescriptor Property

**Description**
Returns the Cal Types from the calset. Learn more about Cal Type.

**VB Syntax**

```vbnet
value = calSet.ContentDescriptor
```

**Variable**

- **Type**: Description
- **calSet**: A `CalSet` (object)

**Return Type**
1-dimensional variant array

**Default**
Not Applicable

**Examples**

```vbnet
Set app = CreateObject("AgilentPNA835x.Application")
Set calset = app.GetCalManager.CalSets.Item(2)
desc = calSet.ContentDescriptor
For i = 0 to UBound(desc)
    MsgBox (desc(i))
Next
```

**C++ Syntax**

```c++
HRESULT get_ContentDescriptor (VARIANT* vals)
```

**Interface**

`ICalSet8`
CoverageFactor Property

Sets and returns the coverage factor value to apply to the displayed uncertainty for the selected measurement trace. Coverage Factor corresponds to the level of confidence used in computing the specified measurement uncertainties.

VB Syntax

uncert.CoverageFactor = value

Variable

(Type) - Description

uncert
An Uncertainty (object)

value
Long. Choose from:

Coverage Factor  Approximate confidence level

1  67%
2  95%
3  99%
4  >99%

Return Type
Long

Default
2

Examples

uncert.CoverageFactor = True

C++ Syntax

HRESULT get_CoverageFactor(long *pSigma);
HRESULT put_CoverageFactor(long sigma);

Interface
IUncertainty
DeembedCoupler Property

Description
Sets and returns the state of de-embedding (reversing) the port 2 coupler.

VB Syntax
phaseRef.DeembedCoupler = value

Variable
(Type) - Description

object
PhaseReferenceCalibration (object)

value
(Boolean) Port 2 couple de-embed state. Choose from:

True - Configures the calibration to include additional measurements to de-embed the effects of reversing the coupler. (This is the same as clearing the “Omit Coupler” checkbox.)

False - Exclude additional measurements for de-embedding the effects of reversing the coupler.

Return Type
Boolean

Default
True

Examples
phaseRef.DeembedCoupler = True

value = phaseRef.DeembedCoupler  'Read

C++ Syntax
HRESULT get_DeembedCoupler(VARIANT_BOOL *value)
HRESULT put_DeembedCoupler(VARIANT_BOOL value)

Interface
IPhaseReference2
DisplayType Property

Description
Sets and returns the display type for uncertainties for the selected measurement trace.

VB Syntax
```
uncert_DisplayType = value
```

Variable
(Type) - Description

- uncert
  An Uncertainty (object)

- value
  (Enum as NAUncertaintyDisplayType) Display type. Choose from:
    - 0 or naUncertaintyDisplayNone - Display the trace without uncertainties.
    - 1 or naUncertaintyDisplayMaximum - Display the trace as the uncertainty maximum (measured or memory data + upper limit uncertainty values). Not supported with Smith Chart or Polar display format.
    - 2 or naUncertaintyDisplayMinimum - Display the trace as the uncertainty minimum (measured or memory data - lower limit uncertainty values). Not supported with Smith Chart or Polar display format.
    - 3 or naUncertaintyDisplayBar - Display the uncertainties as “error bars” around the trace. Not supported with Smith Chart or Polar display format.
    - 4 or naUncertaintyDisplayShade - Display the uncertainties as a shaded region around the trace. Not supported with Smith Chart or Polar display format.
    - 5 or naUncertaintyDisplayEllipse - Display the uncertainties in ellipse form. Supported only in Smith Chart or Polar display format.

Return Type
Enum as NAUncertaintyDisplayType

Default
naUncertaintyDisplayNone

Examples
```
uncert_DisplayType = naUncertaintyDisplayMaximum
```

C++ Syntax
```
HRESULT get_DisplayType(enum NAUncertaintyDisplayType *pDispType);

HRESULT put_DisplayType(enum NAUncertaintyDisplayType dispType);
```

Interface
IUncertainty
OBS_ECALCharacterization Property

Specifies the characterization data within an ECal module to be used for the calibration.

Learn more about ECal User Characterization.

VB Syntax

cal.ECALCharacterization(mod) = value

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cal</td>
<td>Calibrator (object)</td>
</tr>
<tr>
<td>module</td>
<td>1 - ECal module</td>
</tr>
<tr>
<td>value</td>
<td>(Long) – Characterization data within the ECal module to be used for ECal operations. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 – Factory Characterization</td>
</tr>
<tr>
<td></td>
<td>1 – UserCharacterization1</td>
</tr>
<tr>
<td></td>
<td>2 – UserCharacterization2</td>
</tr>
<tr>
<td></td>
<td>3 – UserCharacterization3</td>
</tr>
<tr>
<td></td>
<td>4 – UserCharacterization4</td>
</tr>
<tr>
<td></td>
<td>5 – UserCharacterization5</td>
</tr>
</tbody>
</table>

Return Type
Long

Default
0 - Factory Characterization

Interface
ICalibrator2
OBS_ECALCharacterizationIndexList Property

Description
Returns a list of characterizations stored in the specified ECal module. Learn more about ECal User characterization.

VB Syntax
clist = cal.ECALCharacterizationIndexList(moduleNum)

Variable
(Type) - Description
clist
Variable to store the returned list of characterization index numbers.

cal
Calibrator (object)
module
ECal module from which to read user characterization numbers.

Return Type
Variant

Default
Not Applicable

Examples
Module 1 contains User Characterizations 1 and 3.
clist = cal.ECALCharacterizationIndexList(1)
Returns the following (0 always indicates the factory characterization):
0,1,3

C++ Syntax
HRESULT get_ECALCharacterizationIndexList( long moduleNumber, VARIANT *characterizations);

Interface
ICalibrator6
OBS_ECALPortMap_Property

Description

Note: This command is replaced by ECALPortMapEx

Specifies which ports of the ECal module are connected to which ports of the VNA for the DoECAL1Port and DoECAL2Port methods when the OrientECALModule property = False.

VB Syntax

cal.ECALPortMap = value

Variable

(Type) - Description

cal A Calibrator (object)

value (string)

(string) - Format this parameter in the following manner:

Aw,Bx,Cy,Dz

where

A, B, C, and D are literal ports on the ECAL module
w,x,y, and z are substituted for VNA port numbers to which the ECAL module port is connected.

Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECal module with

port A connected to VNA port 2
port B connected to VNA port 3
port C not connected
port D connected to VNA port 1

the string would be: A2,B3,D1

DoECAL1Port or DoECAL2Port methods will fail if the port numbers passed to those methods are not in the string of this property and OrientECALModule property = False.

Return Type

String

Default

Not Applicable

Examples

Dim cal As Calibrator
Dim sPortMap As String
Set cal = PNAapp.ActiveChannel.Calibrator
cal.ECALPortMap = "a2,b1" 'Write
sPortMap = cal.ECALPortMap 'Read

C++ Syntax
HRESULT put_ECALPortMap(tagNAECALModule ecalModule, BSTR strPortMap);
HRESULT get_ECALPortMap(tagNAECALModule ecalModule, BSTR *strPortMap);

Interface
ICalibrator3
HiSLIPAddress Property

Description Set and returns HiSLIP instrument number.

VB Syntax

IOConfig.HiSLIPAddress = value

Variable (Type) - Description

IOConfig An IOConfiguration object.

value (Long) HiSLIP instrument number.

Return Type Long

Default 0 (which renders "inst0")

Examples

IOConfig.HiSLIPAddress = 1 'Write
value = IOConfig.HiSLIPAddress 'Read

C++ Syntax

HRESULT get HiSLIPAddress(long Address);
HRESULT put HiSLIPAddress(long * Address);

Interface IIOConfiguration
## HiSLIPPort Property

**Description**  
Set and returns the TCPIP port for HiSLIP communication.

**VB Syntax**  
```vbnet
IOConfig.HiSLIPPort = value
```

**Variable**  
*(Type) - Description*

- **IOConfig**  
  An **IOConfiguration** object.

- **value**  
  (Long) HiSlip port number.

**Return Type**  
Long

**Default**  
4880

**Examples**  
```vbnet
IOConfig.HiSLIPPort = 4880 'Write
tvalue = IOConfig.HiSLIPPort 'Read
```

**C++ Syntax**  
```cpp
HRESULT get HiSLIPPort(long port);
HRESULT put HiSLIPPort(long * port);
```

**Interface**  
**IIOConfiguration**
**IMDECalExtrapolation Property**

**Description**
Sets whether a Swept IMD or IMDx calibration can exceed the stop frequency limit of an ECal module.

**VB Syntax**
```vbnet
pref.IMDECalExtrapolation = bool
```

**Variable**
(Type) - Description

`pref`
A `Preferences` (object)

`bool`
(Boolean) - Choose from:
False - Do NOT allow extrapolation.
True - Allow extrapolation.

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
pref.IMDECalExtrapolation = False 'Write
prefer = pref.IMDECalExtrapolation 'Read
```

**C++ Syntax**
```cpp
HRESULT put_IMDECalExtrapolation( VARIANT_BOOL bExtrap)
HRESULT get_IMDECalExtrapolation( VARIANT_BOOL *bExtrap)
```

**Interface**
`IPreferences14`
InSituCharacterization Property

Description
Sets or returns whether the device (CalPod module) that was specified by ECallID will be characterized as an in situ device. Learn more.

VB Syntax
ecalUser.InSituCharacterization = value

Variable
(Type) - Description

ecalUser An ECalUserCharacterizer (object)

value (Boolean) In situ state. Choose from:
True - Characterize the CalPod module as an in situ device.
False - Do NOT characterize the CalPod module as an in situ device.

Return Type
Boolean

Default
True

Examples
ecalUser.InSituCharacterization = False ' Write

Value = ecalUser.InSituCharacterization

C++ Syntax
HRESULT get_InSituCharacterization(VARIANT_BOOL *value);

HRESULT put_InSituCharacterization(VARIANT_BOOL value);

Interface
IECalUserCharacterizer3
InterfaceTypes Property

Description  Returns the valid interface types for the VNA.

VB Syntax  

value = IOConfig.InterfaceTypes

Variable  

(Type) - Description
value  (Variant) variable to store the returned information.

IOConfig  An IOConfiguration object.

Return Type  Array of strings

Default  Not Applicable

Examples  

values = IO.InterfaceTypes
For i = lbound(values) to ubound(values)
  Wscript.echo values(i)
next

C++ Syntax  

HRESULT InterfaceTypes(VARIANT* recognizedTypes);

Interface  IIOConfiguration
IsBlocked Property

Description
Reads whether the specified channel is currently 'blocked' from sweeping. Learn more about the Mechanical Devices dialog.

VB Syntax
value = chan.IsBlocked

Variable
(Type) - Description
value (boolean) - Choose either:
False - The channel IS blocked.
True - The channel is NOT blocked.

chan Channel (object)

Return Type
Boolean

Default
Not Applicable

Examples
block = chan.IsBlocked 'Read

C++ Syntax
HRESULT get_IsBlocked (VARIANT_BOOL *bBlock);

Interface
IChannel19
**OBS_IsECALModuleFound Property**

**Description**
Tests communication between the VNA and the specified ECal module.

**Note:** This property is replaced by `IsECALModuleFoundEx Property`.

**VB Syntax**
```
modFound = cal.IsECALModuleFound (module)
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>modFound</td>
<td>(boolean)</td>
<td>Variable to store the returned test result. True - The VNA identified the presence of the specified ECal module. False - The VNA did NOT identify the presence of the specified ECal module.</td>
</tr>
</tbody>
</table>

| cal | (object) | A Calibrator object |

<table>
<thead>
<tr>
<th>module</th>
<th>(enum NAECALModule)</th>
<th>ECAL module. Choose from:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>naECALModule_A</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>naECALModule_B</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**

Boolean

**Default**

Not applicable

**Examples**
```
Set cal = pna.ActiveChannel.Calibrator
moduleFound = cal.IsECALModuleFound(naECALModule_A)
```

**C++ Syntax**
```
HRESULT get_IsECALModuleFound(tagNAECALModule moduleNumber, VARIANT_BOOL *bModuleFound);
```

**Interface**

Calibrator
IterationCountForStep Property

Description
Designed to be used for an iterative cal standard such as a sliding load, this command returns the number of iterative measurement acquisitions that has been made for the specified step.

Zero (0) is returned if the step has not yet been measured.

For most cal steps that have already been measured, this command returns 1.

To count acquisition steps, set SlidingLoadAcquisitionBehavior.
value = guided.IterationCountForStep (n)

VB Syntax

Variable

`value` (Long) Variable to store the returned number of iterations that have been measured for the step.

`guided` GuidedCalibration (object)

`n` Guided Cal step number for which the acquisition number will be returned.

Use GenerateSteps Method to read the number of steps in the calibration.

Return Type
Long

Default
Not Applicable

Examples

`value = obj.IterationCountForStep(4)`

C++ Syntax

`HRESULT get_IterationCountForStep (long step, long *iterations);`

Interface
IGuidedCalibration10
LevelingMethod Property

Description: Sets and returns the tone power leveling mode.

VB Syntax: 
```
object.LevelingMethod = value
```

Variable (Type) - Description

- object: One of the following:
  - IMSpectrum Object
  - SweptIMD Object

- value (enum NAIMDLevelMethod) - Choose from:
  - 0 - naNOLevel
  - 1 - naLevelInput
  - 2 - naEqualizeOutput
  - 3 - naLevelOutput

Return Type: Enum

Default: naNOLevel

Examples:
- `imd.LevelingMethod = naEqualizeOutput` 'Write'
- `levMode = ims.LevelingMethod` 'Read'

C++ Syntax:
```
HRESULT get_LevelingMethod(tagNALevelingMethods* pVal)
HRESULT put_LevelingMethod(tagNALevelingMethods newVal)
```

Interface:
- IIMSpectrum
- ISweptIMD
**LOCount Property**

**Description**
Returns the number of LO acquisitions determined by the Image Reject selection and the span.

**VB Syntax**

```vb
value = sa.LOCount
```

**Variable**

- **sa**
  - A `SpectrumAnalyzer` (object)
- **value**
  - (Long) Variable to store the returned number of LOs.

**Return Type**
Long

**Default**
N/A

**Examples**

```vb
value = sa.LOCount
```

See an example program.

**C++ Syntax**

```cpp
HRESULT get_LOCount(long* val);
```

**Interface**
ISpectrumAnalyzer
MatchFrequencyRange Property

Description
Sets and reads the existing frequency ranges over which Match Correction is to be performed.

On the Source Configuration dialog under Match Correction, this is the Match Frequency Range setting.

VB Syntax
DIQ.MatchFrequencyRange (port) = value

Variable
(Type) - Description

DIQ
A Differential I/Q (object)

port
(String) Source port name. Use SourcePortNames to read a list of valid source ports.

value
(String) Frequency ranges, including the "F<n>", where <n> is the range number. Separate each range with a comma.

Return Type
String

Default
Depends on <port>

Examples

<table>
<thead>
<tr>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oDIQ.MatchFrequencyRange(&quot;port 1&quot;) = &quot;F1,F2,F3&quot;</td>
<td></td>
</tr>
<tr>
<td>Value = oDIQ.MatchFrequencyRange(&quot;port 2&quot;)</td>
<td>Read</td>
</tr>
</tbody>
</table>

C++ Syntax
HRESULT get_MatchFrequencyRange(BSTR port, BSTR* MatchFrequencyRange);
HRESULT put_MatchFrequencyRange(BSTR port, BSTR MatchFrequencyRange);

Interface
IDIQ
**MatchRefReceiver Property**

**Description**

Sets and reads the reference receiver to be used to perform Match Correction.

On the *Source Configuration* dialog under Match Correction, this is the Reference Receiver setting.

**VB Syntax**

```
DIQ.MatchRefReceiver (port) = value
```

**Variable**

*(Type) - Description*

- **DIQ**
  
  *A Differential I/Q (object)*

- **port**
  
  *(String) Source port name. Use *SourcePortNames* to read a list of valid source ports.*

- **value**
  
  *(String) Choose any of the reference receivers in the analyzer using logical receiver notation. [Learn more](#). These would be "a_" where _ is the test port number.*

**Return Type**

String

**Default**

Depends on <port>

**Examples**

```
oDIQ.MatchRefReceiver("port 1") = "a1"
```

```
Value = oDIQ.MatchRefReceiver("port 2") 'Read
```

**C++ Syntax**

```
HRESULT get_MatchRefReceiver(BSTR port, BSTR* MatchRefReceiver);
HRESULT put_MatchRefReceiver(BSTR port, BSTR MatchRefReceiver);
```

**Interface**

IDIQ
MatchState Property

Description
Sets and reads the Match Correction ON/OFF state.

On the Source Configuration dialog under Match Correction, this is the Match Correction ON setting.

VB Syntax
DIQ.MatchState (port) = value

Variable
(Type) - Description

DIQ
A Differential I/Q (object)

port
(String) Source port name. Use SourcePortNames to read a list of valid source ports.

value
(Boolean) Choose from:

    True - Perform Match Correction
    False - Do not perform Match Correction.

Return Type
Boolean

Default
False

Examples

    oDIQ.MatchState("port 2") = True

    Value = oDIQ.MatchState("port 2") 'Read

C++ Syntax
HRESULT get_MatchState(BSTR port, BOOL* MatchState);
HRESULT put_MatchState(BSTR port, BOOL MatchState);

Interface
IDIQ
MatchTestReceiver Property

**Description**
Sets and reads the test port receiver to be used to perform Match Correction.

On the Source Configuration dialog under Match Correction, this is the Test Receiver setting.

**VB Syntax**

```
DIQ.MatchTestReceiver (port) = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DIQ</code></td>
<td>A Differential I/Q (object)</td>
</tr>
</tbody>
</table>

```
port (String) Source port name. Use SourcePortNames to read a list of valid source ports.
```

```
value (String) Choose any of the test port receivers in the analyzer using logical receiver notation. Learn more.
These would be "b_" where _ is the test port number.
```

**Return Type**

String

**Default**

Depends on <port>

**Examples**

```
oDIQ.MatchTestReceiver("port 1") = "b1"
```

```
Value = oDIQ.MatchTestReceiver("port 2") 'Read
```

**C++ Syntax**

```
HRESULT get_MatchTestReceiver(BSTR port, BSTR* MatchTestReceiver);
HRESULT put_MatchTestReceiver(BSTR port, BSTR MatchTestReceiver);
```

**Interface**

IDIQ
MeasurementNoiseUncertainty Property

**Description**
Sets and returns whether the noise contribution is currently included in the uncertainty values for the measurement trace.

**VB Syntax**
```
uncert.MeasurementNoiseUncertainty = value
```

**Variable**
(Type) - Description

uncert
(A n Uncertainty (object)

value
(Boolean) Choose from:
False - Noise contribution is NOT included.
True - Noise contribution is included.

**Return Type**
Boolean

**Default**
True

**Examples**
```
uncert.MeasurementNoiseUncertainty = True
```

**C++ Syntax**
```
HRESULT get_MeasurementNoiseUncertainty(VARIANT_BOOL *pState);

HRESULT put_MeasurementNoiseUncertainty(VARIANT_BOOL state);
```

**Interface**
IUncertainty
**MinimumIterationsForStep Property**

**Description**

Designed to be used for an iterative cal standard such as a sliding load, this command returns the minimum number of required iterative measurement acquisitions for the specified step.

For most connection steps this will return 1, but for an iterative cal standard such as a sliding load, it will return a number such as 5.

To count acquisition steps, set `SlidingLoadAcquisitionBehavior`.

```vbnet
value = guided.MinimumIterationsForStep(n)
```

**Variable**

- **value** (Type) - Description
  - (Long) Variable to store the returned number of minimum iterations for the step.
- **guided**
  - `GuidedCalibration` (object)
  - Guided Cal step number for which to return the number of iterative measurement acquisitions that have been made.

*Use GenerateSteps Method to read the number of steps in the calibration.*

**Return Type**

Long

**Default**

Not Applicable

**Examples**

```vbnet
value = obj.MinimumIterationsForStep(4)
```

**C++ Syntax**

```cpp
HRESULT get_MinimumIterationsForStep (long step, long *iterations);
```

**Interface**

`IGuidedCalibration10`
NarrowBand Property

Description
Turns narrowband noise figure compensation ON and OFF.

VB Syntax
noise.NarrowBand = value

Variable
- noise (variable) - A NoiseFigure (object)
- value (boolean) - Narrowband compensation state.
  - False - Narrowband compensation OFF
  - True - Narrowband compensation ON

Return Type
Boolean

Default
False

Examples
noise.NarrowBand = OFF 'Write
Narrowband = noise.NarrowBand 'Read

C++ Syntax
HRESULT get_NarrowBand(VARIANT_BOOL* on);
HRESULT put_NarrowBand(VARIANT_BOOL on);

Interface
INoiseFigure
### NoiseCalTemperature Property

**Description**
Read the cal temperature of the noise source.

**VB Syntax**
```vbnet
temp = enr.NoiseCalTemperature()
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>temp</td>
<td>Variable to store the returned noise source temperature.</td>
</tr>
</tbody>
</table>

| enr       | An ENRFile (object) |

| Return Type | Double |
| Default     | Not Applicable |

<table>
<thead>
<tr>
<th>Examples</th>
<th>See example program</th>
</tr>
</thead>
</table>

| C++ Syntax | HRESULT NoiseCalTemperature(double* temp); |
| Interface  | IENRFile            |
OBS_Normalization Property

Write/Read          About Receiver Cal
Normalization Property  Superseded

Description

Note: This property is replaced by DoReceiverPowerCal Method.
Sets or returns normalization ON or OFF for the measurement. Normalization is currently supported only on measurements of unratioed power for the purpose of performing a receiver power calibration. If this property is set to ON for a ratioed measurement (such as S-parameter), it will return an error. This property will also return an error when set to ON if the divisor buffer doesn’t yet exist.

VB Syntax

meas.Normalization = value

Variable

(meas) - Description
(meas) - A Measurement object

(value) - (boolean)
False – Turns normalization OFF
True – Turns normalization ON

Return Type

Boolean

Default

False

Examples

meas.Normalization = False 'Write
normalized = meas.Normalization 'Read

C++ Syntax

HRESULT put_Normalization(VARIANT_BOOL bState);
HRESULT get_Normalization(VARIANT_BOOL *bState);

Interface

IMeasurement
**Number (ports) Property**

**Description**
Returns the VNA port number.

**VB Syntax**
```
value = port.Number
```

**Variable**
<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>A \texttt{Port} (object)</td>
</tr>
<tr>
<td>value</td>
<td>\texttt{Long Integer} - Variable to store the returned Port number.</td>
</tr>
</tbody>
</table>

**Return Type**
Long

**Default**
Not Applicable

**Examples**
```
value = port.Number \ 'Read the value
```

**C++ Syntax**
```
HRESULT get_Number([out,retval] long *port);
```

**Interface**
iPort
PhaseSweepState Property

Description
Sets and reads the ON / OFF state of phase sweep.

On the Source Configuration dialog under Phase, this is the Sweep Phase (On|Off) setting.

VB Syntax
DIQ.PhaseSweepState (port) = value

Variable
(Type) - Description

DIQ
A Differential I/Q (object)

port (String) Source port name. Use SourcePortNames to read a list of valid source ports.

value (Boolean) Choose from:

  True - Sweep phase.
  False - Do not sweep phase.

Return Type
Boolean

Default
False

Examples
oDIQ.PhaseSweepState("port 2") = True
Value = oDIQ.PhaseSweepState("port 2") 'Read

C++ Syntax
HRESULT get_PhaseSweepState(BSTR port, BOOL* PhaseSweepState);
HRESULT put_PhaseSweepState(BSTR port, BOOL PhaseSweepState);

Interface
IDIQ
PortAttenuator Property

Description
Sets and reads the amount of source attenuation. Sending this command will set AutoRangeState Property False.

On the Source Configuration dialog under Power, this is the Source Attenuation setting.

VB Syntax
DIQ.PortAttenuator (port) = value

Variable
(Type) - Description

DIQ
A Differential I/Q (object)

port
(String) Source port name. Use SourcePortNames to read a list of valid source ports.

value
(Double) Source Attenuation value. Choose from 0 to the maximum amount of source attenuation in the correct step size.

Rounding will occur when the selected value can not be achieved.

See attenuators for all VNA models.

Return Type
Double

Default
0

Examples
oDIQ.PortAttenuator("port 2") = 0

Value = oDIQ.PortAttenuator("port 2") 'Read

C++ Syntax
HRESULT get_PortAttenuator(BSTR port, double* PortAttenuator);
HRESULT put_PortAttenuator(BSTR port, double PortAttenuator);

Interface
IDIQ
PortLevelingMode Property

Description
Sets and reads the leveling mode to be used for the specified source port.

On the Source Configuration dialog under Power, this is the Leveling Mode setting.

VB Syntax
DIQ.PortLevelingMode (port) = value

Variable (Type) - Description

DIQ
A Differential I/Q (object)

port (String) Source port name. Use SourcePortNames to read a list of valid source ports.

value (String) Leveling mode. Choose from:
"Internal" - Standard internal analyzer leveling mode.
"Internal-R<n>,<p>" - Receiver Leveling, where:
<n> is the receiver name.
<p> is the source port to be leveled.

"Open Loop" - Open loop leveling, used during pulse conditions with the internal source modulators. NOT available on all models. No leveling is used in setting the source power. The lowest settable power, without attenuation, is limited to -30dBm. The source power level accuracy is very compromised. Use a source power calibration to make the source power somewhat more accurate.

"Open Loop-<n>,<p>" - Open loop leveling, where:
<n> is the receiver name.
<p> is the source port to be leveled

Return Type
String

Default
"Internal"

Examples

Value = oDIQ.PortLevelingMode("port 2") 'Read

C++ Syntax

HRESULT get_PortLevelingMode(BSTR port, BSTR* PortLevelingMode);
HRESULT put_PortLevelingMode(BSTR port, BSTR PortLevelingMode);

Interface
IDIQ
**PortPhaseParameter Property**

**Description**

Sets and reads the receivers to be used to measure the phase of the sources. The phase measurement will be the difference between these two receivers.

Select the receivers based on your application. You are responsible to make sure that your DUT configuration routes the signals of interest to the correct receivers. Otherwise, the phase will not be properly controlled.

On the **Source Configuration** dialog under Phase, this is the Control Receiver setting.

**VB Syntax**

```vbnet
DIQ.PortPhaseParameter (port) = value
```

**Variable**

- **(Type)** - Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIQ</td>
<td>A Differential I/Q (object)</td>
</tr>
<tr>
<td>port</td>
<td>(String) Source port name. Use <strong>SourcePortNames</strong> to read a list of valid source ports.</td>
</tr>
<tr>
<td>value</td>
<td>(String) Phase parameter using the following syntax: <strong>rCont/rRef</strong></td>
</tr>
</tbody>
</table>

Where:

- **rCont** is the receiver used to measure the controlled source.
- **rRef** is the receiver used to measure the reference source.

Only logical receiver notation (a1,b1, etc) is available. Learn more.

Enclose the entire parameter in quotes.

**Return Type**

String

**Default**

Depends on `<port>`

**Examples**

```vbnet
oDIQ.PortPhaseParameter("port 1") = "a1/a3"
```

Value = oDIQ.PortPhaseParameter("port 2") 'Read

**C++ Syntax**

```c++
HRESULT get_PortPhaseParameter(BSTR port, BSTR* PortPhaseParameter);
HRESULT put_PortPhaseParameter(BSTR port, BSTR PortPhaseParameter);
```

**Interface**

IDIQ
PortPhaseStart Property

Description
Sets and reads the start value for a phase sweep.

On the Source Configuration dialog under Phase, this is the Start Phase setting.

VB Syntax
DIQ.PortPhaseStart (port) = value

Variable
(Type) - Description

DIQ
A Differential I/Q (object)

port (String) Source port name. Use SourcePortNames to read a list of valid source ports.

value (Double) Start phase sweep value in degrees. Choose any positive or negative value.

Return Type
Double

Default
0

Examples
oDIQ.PortPhaseStart("port 2") = -90

Value = oDIQ.PortPhaseStart("port 2") 'Read

C++ Syntax
HRESULT get_PortPhaseStart(BSTR port, double* PortPhaseStart);
HRESULT put_PortPhaseStart(BSTR port, double PortPhaseStart);

Interface
IDIQ
PortPhaseState Property

Sets and reads the ON/OFF state of phase control.

On the Source Configuration dialog under Phase, this is the Phase State setting.

### VB Syntax

```vbnet
DIQ.PortPhaseState (port) = value
```

### Variable

- **Type** - Description
  - **DIQ** - A Differential I/Q (object)
  - **port** - (String) Source port name. Use SourcePortNames to read a list of valid source ports.
  - **value** - (Enum as NAPhaseControlMode) Choose from:
    - **naPhaseControlOff or 0** - Phase is NOT set or controlled.
    - **naPhaseControlParameter or 1** - Phase is measured and iterated to within the specified tolerance. Specify the receivers and iteration properties using the Source:Phase commands.
    - **naPhaseControlOpenLoop or 3** - Phase is set, but receivers are NOT used to measure and iterate the phase of the source. Therefore, the setting of phase is not as accurate or stable. Open Loop mode can be used with phase sweep (for example, from 0 to 360 degrees). However, each sweep may not start at 0 degrees. NO settings on the Phase Control Setup dialog are used in Open Loop. After selecting Open Loop, set each source to ON (not Auto).

- **Return Type** - Enum
- **Default** - naPhaseControlOff or 0
- **Examples**
  ```vbnet
diq.PortPhaseState = naPhaseControlOff 'Write
value = diq.PortPhaseState 'Read
```

- **C++ Syntax**
  ```cpp
  HRESULT get_PortPhaseStateBSTR (BSTR port, tagNAPhaseControlMode* PortPhaseState);
  HRESULT put_PortPhaseState (BSTR port, tagNAPhaseControlMode PortPhaseState);
  ```

- **Interface**
  ```cpp
  IDifferentialIQ
  ```
PortPhaseStop Property

Description
Sets and reads the Stop value for a phase sweep.
On the Source Configuration dialog under Phase, this is the Stop Phase setting.

VB Syntax
DIQ.PortPhaseStop (port) = value

Variable (Type) - Description

DIQ
A Differential I/Q (object)

port (String) Source port name. Use SourcePortNames to read a list of valid source ports.

value (Double) Stop phase sweep value in degrees. Choose any positive or negative value.

Return Type
Double

Default
0

Examples
oDIQ.PortPhaseStop("port 2") = -90
Value = oDIQ.PortPhaseStop("port 2") 'Read

C++ Syntax
HRESULT get_PortPhaseStop(BSTR port, double* PortPhaseStop);
HRESULT put_PortPhaseStop(BSTR port, double PortPhaseStop);

Interface
IDIQ
PortReference Property

Description
Sets and reads the port to be used as a reference when controlling phase for the specified <port>.

On the Source Configuration dialog under Phase, this is the Refer To setting.

VB Syntax
DIQ.PortReference (port) = value

Variable
(Type) - Description

DIQ
A Differential I/Q (object)

port
(String) Source port name. Use SourcePortNames to read a list of valid source ports.

value
(String) Reference port. Use SourcePortNames to return a list of valid port names.

The two internal VNA sources are available ONLY at specific ports. For example on a 4-port PNA-X, the possible port pairings are: 1/3, 1/4, 2/3, or 2/4. Port 1 can NOT be paired with Port 2, and Port 3 can NOT be paired with Port 4. Learn more about these limitations.

Return Type
String

Default
Depends on <port>

Examples

<table>
<thead>
<tr>
<th>oDIQ.PortReference(&quot;port 1&quot;) = &quot;Port 3&quot;</th>
</tr>
</thead>
</table>

Value = oDIQ.PortReference("port 2")  'Read

C++ Syntax

HRESULT get_PortReference(BSTR port, BSTR* PortReference);
HRESULT put_PortReference(BSTR port, BSTR PortReference);

Interface
IDIQ

7945
PortStartPower Property

Description
Sets and reads the start value of a power sweep.

On the Source Configuration dialog under Power, this is the Start Power setting.

VB Syntax
DIQ.PortStartPower (port) = value

Variable
(Type) - Description

DIQ
A Differential I/Q (object)

port
(String) Source port name. Use SourcePortNames to read a list of valid source ports.

value
(Double) Power sweep start value in dBm. Choose start and stop value within a single attenuator range of the analyzer (generally about 30 dB).

Return Type
Double

Default
-5 dBm

Examples
oDIQ.PortStartPower("port 2") = 0

C++ Syntax
HRESULT get_PortStartPower(BSTR port, double* PortStartPower);
HRESULT put_PortStartPower(BSTR port, double PortStartPower);

Interface
IDIQ
PortStopPower Property

Description
Sets and reads the stop value of a power sweep.
On the Source Configuration dialog under Power, this is the Stop Power setting.

VB Syntax
DIQ.PortStopPower (port) = value

Variable
(Type) - Description

DIQ
A Differential I/Q (object)

port
(String) Source port name. Use SourcePortNames to read a list of valid source ports.

value
(Double) Power sweep stop value in dBm. Choose start and stop value within a single
attenuator range of the analyzer (generally about 30 dB).

Return Type
Double

Default
-5 dBm

Examples
oDIQ.PortStopPower("port 2") = 0
Value = oDIQ.PortStopPower("port 2") 'Read

C++ Syntax
HRESULT get_PortStopPower(BSTR port, double* PortStopPower);
HRESULT put_PortStopPower(BSTR port, double PortStopPower);

Interface
IDIQ
PowerSweepState Property

Description
Sets and reads the state of sweeping power.

On the Source Configuration dialog under Power, this is the Sweep Power (On|Off) setting.

VB Syntax
DIQ.PowerSweepState (port) = value

Variable
(Type) - Description

DIQ (object)

port (String) Source port name. Use SourcePortNames to read a list of valid source ports.

value (Boolean) Sweep Power state. Choose from:
- True - Perform power sweep.
- False - Do not perform power sweep.

Return Type
Boolean

Default
False

Examples
oDIQ.PowerSweepState("port 2") = True

Value = oDIQ.PowerSweepState("port 2") 'Read

C++ Syntax
HRESULT get_PowerSweepState(BSTR port, BOOL* PowerSweepState);
HRESULT put_PowerSweepState(BSTR port, BOOL PowerSweepState);

Interface
IDIQ
Properties Property

Description
Returns the properties of the calset. See the Calset Properties page.

VB Syntax
value = calSet.Properties

Variable
(Type) - Description

calSet  A CalSet (object)

Return Type
String

Default
Not Applicable

Examples
Set app = CreateObject("AgilentPNA835x.Application")
Set calset = app.GetCalManager.CalSets.Item(2)
props = CalSet.Properties
Msgbox (props)

C++ Syntax
HRESULT get_Properties (BSTR* vals);

Interface
ICalSet8
**PulseTimingDevice Property**

**Description**
Sets and reads the device being controlled by the pulse generator output.

**VB Syntax**
```
obj.TriggerInType = value
```

**Variable**
(Type) - Description

- **pulse** (Type) - Description
  - A PulseGenerator (object)

- **value** (enum NAPulseGenToDevice) Choose from:
  - 0 - ADC_TRIGGER - Enables offset pulses using ADC delay.
  - 1 - RF_MODULATOR - Indicates that the pulse signal is used to drive the RF modulator. Only one pulse generator output can be used to drive an RF source. If you try to set more than one pulse generator output to RF_MODULATOR, then the other one will be set to UserN (where "N" is the pulse generator number).
  - 2 - ADC_ACTIVITY - (Pulse4 only) Pulse4 can also be set to monitor ADC activity. This selection outputs a signal on Pulse4 when the ADC is active. This is the same as Pulse4OutAsADCAActivity.
  - 3 - USER1 - Use Pulse1 to control a DUT, DC biases, or other signals.
  - 4 - USER2 - Use Pulse2 to control a DUT, DC biases, or other signals.
  - 5 - USER3 - Use Pulse3 to control a DUT, DC biases, or other signals.
  - 6 - USER4 - Use Pulse4 to control a DUT, DC biases, or other signals.

**Return Type**
Enum

**Default**
RF_MODULATOR

**Examples**
```
pulse.PulseTimingDevice = RF_MODULATOR 'Write
```
```
value = pulse.PulseTimingDevice 'Read the value
```

**C++ Syntax**
```
HRESULT get_PulseTimingDevice(enum NAPulseGenToDevice *val);
HRESULT put_PulseTimingDevice(enum NAPulseGenToDevice val);
```

**Interface**
IPulseGenerator6
## RangeCoupleId Property

**Description**
Sets and reads the frequency range to couple settings to.

On the **Frequency Range** dialog under Coupling, this is the range to Couple To setting.

**VB Syntax**
```
DIQ.RangeCoupleId (n) = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DIQ</strong></td>
</tr>
<tr>
<td>A Differential I/Q (object)</td>
</tr>
</tbody>
</table>

| (Long Integer) Frequency range number. |
| **n** |

| (Long Integer) Frequency range number to couple to. This range should be already created using AddRange Method. |
| **value** |

**Return Type**
Long

**Default**
1

**Examples**
```vba
diq.RangeCoupleId 2 = 1 'Write, range 2 is coupled to range 1
value = diq.RangeCoupleId 2 'Read
```

**C++ Syntax**
```
HRESULT get_RangeCoupleId(long range, long* RangeCoupleId);
HRESULT put_RangeCoupleId(long range, long RangeCoupleId);
```

**Interface**
IDIQ
RangeCoupleState Property

Description
Sets and reads the ON / OFF state of frequency range coupling.

On the Frequency Range dialog under Coupling, this is the Couple (On|Off) setting.

VB Syntax
DIQ.RangeCoupleState (n) = value

Variable
(Type) - Description

DIQ
A Differential I/Q (object)

n (Long) Frequency range number.

value (Boolean) Choose from the following:
True - Range is coupled
False - Range is NOT coupled

Return Type
Boolean

Default
False

Examples
diq.RangeCoupleState 2 = True 'Write, range 2 is coupled to range 1
value = diq.RangeCoupleState 2 'Read

C++ Syntax
HRESULT get_RangeCoupleState(long range, VARIANT_BOOL* RangeCoupleState);
HRESULT put_RangeCoupleState(long range, VARIANT_BOOL RangeCoupleState);

Interface
IDIQ
RangeDivisor Property

Description
Sets and reads the value by which the coupled range will be divided to achieve the frequency range specified by <rNum>.

On the Frequency Range dialog under Coupling, this is the Divisor setting.

VB Syntax
DIQ.RangeDivisor (n) = value

Variable
(Type) - Description

DIQ
A Differential I/Q (object)

n (Long Integer) Frequency range number.

value (Long Integer) Divisor value. Choose a positive or negative integer.

Return Type
Long

Default
1

Examples

diq.RangeDivisor 2 = 2 'Write

value = diq.RangeDivisor 2 'Read

C++ Syntax
HRESULT get_RangeDivisor(long range, long* RangeDivisor);
HRESULT put_RangeDivisor(long range, long RangeDivisor);

Interface
IDIQ
RangeIFBW Property

**Description**
Sets and reads the receiver IF bandwidth setting.

On the Frequency Range dialog under Frequency, this is the IFBW setting.

**VB Syntax**

```vbnet
DIQ.RangeIFBW (n) = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIQ</td>
<td>A Differential I/Q (object)</td>
</tr>
</tbody>
</table>

| n      | (Long Integer) Frequency range number. |
| value  | (Double) IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the analyzer model. (See the list.) If an invalid number is specified, the analyzer will round up to the closest valid number. |

**Return Type**
Double

**Default**
1e5

**Examples**

```vbnet
diq.RangeIFBW 1 = 1e3 'Write
```

```vbnet
value = diq.RangeIFBW 'Read
```

**C++ Syntax**

```cpp
HRESULT get_RangelFBW(long range, double* RangeIFBW);
HRESULT put_RangelFBW(long range, double RangeIFBW);
```

**Interface**

IDIQ
### RangeMultiplier Property

#### Description
Sets and reads the value by which the coupled range will be multiplied to achieve the frequency range specified by \( n \).

On the Frequency Range dialog under Coupling, this is the Multiplier setting.

#### VB Syntax
```
DIQ.RangeMultiplier (n) = value
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIQ</td>
<td>A Differential I/Q (object)</td>
</tr>
<tr>
<td>n</td>
<td>(Long Integer) Frequency range number.</td>
</tr>
<tr>
<td>value</td>
<td>(Long Integer) Multiplier value. Choose a positive or negative integer.</td>
</tr>
</tbody>
</table>

| Return Type | Long |
| Default     | 1 |

#### Examples
```
diq.RangeMultiplier 2 = 1 'Write
```
```
value = diq.RangeMultiplier 2 'Read
```

#### C++ Syntax
```
HRESULT get_RangeMultiplier(long range, long* RangeMultiplier);
HRESULT put_RangeMultiplier(long range, long RangeMultiplier);
```

<table>
<thead>
<tr>
<th>Interface</th>
<th>IDIQ</th>
</tr>
</thead>
</table>

---

7955
## SerialNumber Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the serial number of the VNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.SerialNumber</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(String) - Variable to store the returned value.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A <code>Capabilities</code> (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.SerialNumber</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_SerialNumber(BSTR * serial);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>ICapabilities11</code></td>
</tr>
</tbody>
</table>
SlidingLoadAcquisitionBehavior Property

Description
Specifies the behavior for guided cal steps that involve a sliding load in a cal that is about to be performed.

Send this command BEFORE sending the Initialize command.

VB Syntax
guided.\texttt{SlidingLoadAcquisitionBehavior} = value

Variable
\texttt{guided} - Description

\texttt{value} - (Enum as NASlidingLoadAcquisitionBehavior) Choose from:

0 - \texttt{naShowDialog} - The ‘Move Sliding Load’ prompt is presented on the VNA screen. All slide positions are measured from a single invocation of the acquire command.

1 - \texttt{naMeasureSlidePosition} - Each invocation of the acquire command for a sliding load step measures a single slide position and increments the slide position counter. No ‘Move Sliding Load’ prompt is presented on the VNA screen.

Return Type
Enum

Default
0 – \texttt{naShowDialog}

Examples
\begin{verbatim}
obj.SlidingLoadAcquisitionBehavior = naMeasureSlidePosition
\end{verbatim}

C++ Syntax
\begin{verbatim}
HRESULT put_SlidingLoadAcquisitionBehavior(enum NASlidingLoadAcquisitionBehavior behavior);
HRESULT get_SlidingLoadAcquisitionBehavior(enum NASlidingLoadAcquisitionBehavior *behavior);
\end{verbatim}

Interface
IGuidedCalibration10
### SourceRange Property

**Description**
Sets and reads the frequency range number to which the specified source port will be set. At the top of the Source Configuration dialog this is the Frequency Range setting.

**VB Syntax**

```
DIQ.SourceRange (port) = value
```

**Variable**

- **DIQ**
  A Differential I/Q (object)
- **port**
  (String) Source port name. Use SourcePortNames to read a list of valid source ports.
- **value**
  (Long) Frequency range number.

**Return Type**

Long

**Default**

1

**Examples**

```
oDIQ.SourceRange("port 2") = 1
Value = oDIQ.SourceRange("port 2") 'Read
```

**C++ Syntax**

```
HRESULT get_SourceRange(long range, long* SourceRange);
HRESULT put_SourceRange(long range, long SourceRange);
```

**Interface**

IDIQ
Stage3Coefficients Property

Sets and returns Stage3Coefficients.

VB Syntax

```vbnet
spm4.Stage3Coefficients = values
```

Variable

(Type) - Description

`spm4` A `SignalProcessingModuleFour` (object)

`values` (Float array) Filter coefficients

Return Type

Variant

Default

Not Applicable

Examples

```vbnet
spm4.Stage3Coefficients = +0.0E+000,+6.4E+001,+2.56E+002
mode = spm4.Stage3Coefficients
```

C++ Syntax

```cpp
HRESULT get_Stage3Coefficients(Float* pCoefs);
HRESULT put_Stage3Coefficients(Float pCoefs);
```

Interface

`ISignalProcessingModuleFour2`
Stage3MaximumCoefficient Property

Description
Returns the maximum value of the stage 3 coefficients.

VB Syntax
value = spm4.Stage3MaximumCoefficient

Variable
(Type) - Description

value  
(Float) Variable to store the returned Max coefficient.

spm4  
A SignalProcessingModuleFour (object)

Default
Not Applicable

Examples
mode = spm4.Stage3MaximumCoefficient

C++ Syntax
HRESULT get Stage3MaximumCoefficient(float* val);

Interface
ISignalProcessingModuleFour2
Stage3MaximumCoefficientCount Property

Description
Returns the maximum number of coefficients for Stage3 (102400).

VB Syntax
value = spm4.Stage3MaximumCoefficientCount

Variable
(Type) - Description
value (Long) Variable to store the returned Max coefficient count.

spm4
A SignalProcessingModuleFour (object)

Default
Not Applicable

Examples
mode = spm4.Stage3MaximumCoefficientCount 'Read

C++ Syntax
HRESULT get_Stage3MaximumCoefficientCount(long* val);

Interface
ISignalProcessingModuleFour2
Stage3MinimumCoefficient Property

Description  Returns the Minimum value of the stage 3 coefficients.

VB Syntax  

    value = spm4.Stage3MinimumCoefficient

Variable  

(Type) - Description

value  (Float) Variable to store the returned Min coefficient.

spm4  A SignalProcessingModuleFour (object)

Default  Not Applicable

Examples  

    mode = spm4.Stage3MinimumCoefficient

C++ Syntax  

    HRESULT get_Stage3MinimumCoefficient(float* val);

Interface  

    ISignalProcessingModuleFour2
Stage3MinimumCoefficientCount Property

Description
Returns the minimum number of coefficients for Stage3 (2).

VB Syntax
\[ \text{value} = \text{spm4.Stage3MinimumCoefficientCount} \]

Variable
(Type) - Description

value
(Long) Variable to store the returned Min coefficient count.

spm4
A SignalProcessingModuleFour (object)

Default
Not Applicable

Examples
\[ \text{mode} = \text{spm4.Stage3MinimumCoefficientCount} \]

C++ Syntax
HRESULT get_Stage3MinimumCoefficientCount(long* val);

Interface
ISignalProcessingModuleFour2
StandardDefinitionsEnabled Property

Description
Sets and returns the ON/OFF state of allowing the uncertainty associated with the standard definitions in the cal kits to contribute to the uncertainty of a calibration performed using Dynamic Uncertainty. The uncertainty data for the Cal standards must also be present at the time the calibration is performed.

VB Syntax
uncertMan.StandardDefinitionsEnabled = value

Variable
(Type) - Description

uncertMan
An UncertaintyManager Object

desvalue
(Boolean) Enable state. Choose from:
True - Standard definition uncertainty ON.
False - Standard definition uncertainty OFF.

Return Type
Boolean

Default
True

Examples
uncertMan.StandardDefinitionsEnabled = True
See example program

C++ Syntax
HRESULT get.StandardDefinitionsEnabled([out,retval] VARIANT_BOOL* pState);
HRESULT put.StandardDefinitionsEnabled([in] VARIANT_BOOL state);

Interface
IUncertaintyManager
SupportsInSituCharacterization Property

Description
Returns whether the device that was specified by ECallID is a CalPod module, which is capable of being characterized as an in situ device.

VB Syntax
value = ecalUser.SupportsInSituCharacterization

Variable
(Type) - Description
value (Boolean)
True - Device is a CalPod module
False - Device is NOT a CalPod module

ecalUser
An ECalUserCharacterizer (object)

Return Type
Boolean

Examples
Value = ecalUser.SupportsInSituCharacterization

C++ Syntax
HRESULT get_SupportsInSituCharacterization(VARIANT_BOOL *value);

Interface
IECalUserCharacterizer3
TraceDeviationType Property

Description
Calculates the deviation from a least-squares best fit line.

VB Syntax
meas.TraceDeviationType = value

Variable (Type) - Description
meas
A Measurement (object)
value
(enum NATraceDeviationType) - Choose from:
0 - naTraceDeviationOff
1 - naTraceDeviationLinear
2 - naTraceDeviationParabolic
3 - naTraceDeviationCubic

Return Type
Enum

Default
0 - naTraceDeviationOff

Examples
meas.TraceDeviationType = naTraceDeviationLinear 'Write
DeviationType = meas.TraceDeviationType 'Read

C++ Syntax
HRESULT get_TraceDeviationType(tag NATraceDeviationType* devType)
HRESULT put_TraceDeviationType(tag NATraceDeviationType devType)

Interface
IMeasurement20
UnknownMixerInputPower Property

Description
Sets and returns the input power level to the unknown mixer.

VB Syntax
phaseRef.UnknownMixerInputPower = value

Variable
(Type) - Description

object
PhaseReferenceCalibration (object)

value
(Double) Input power level in dBm.

Return Type
Double

Default
-15 dBm

Examples
phaseRef.UnknownMixerInputPower = 0
value = phaseRef.UnknownMixerInputPower 'Read

C++ Syntax
HRESULT get_UnknownMixerInputPower(double *value)
HRESULT put_UnknownMixerInputPower(double value)

Interface
IPhaseReference2
UnknownMixerLOFrequency Property

Description
Sets and returns the LO Frequency to the unknown mixer.

VB Syntax
phaseRef.UnknownMixerLOFrequency = value

Variable
(Type) - Description

object
PhaseReferenceCalibration (object)

value
(Double) LO frequency in Hz. Choose a value between 3 GHz and (Max Frequency minus 1GHz).

For a 26.5 GHz VNA, the range is 3 GHz to 25.5 GHz.

For best results, use the default LO frequency. 3.351GHz. This frequency produces no spurs from the input/LO frequency. And also the Input frequency will have no band breaks.

Return Type
Double

Default
3.351 GHz

Examples
phaseRef.UnknownMixerLOFrequency = 3.351e9
value = phaseRef.UnknownMixerLOFrequency  'Read

C++ Syntax
HRESULT get_UnknownMixerLOFrequency(double *value)
HRESULT put_UnknownMixerLOFrequency(double value)

Interface
IPhaseReference2
# UnknownMixerLOPower Property

**Description**
Sets and returns the LO power level to the unknown mixer.

**VB Syntax**
phaseRef.UnknownMixerLOPower = value

**Variable**
- **Object**
  - `PhaseReferenceCalibration (object)`
- **Value**
  - (Double) LO power level in dBm.

**Return Type**
Double

**Default**
10 dBm

**Examples**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>phaseRef.UnknownMixerLOPower = 10</code></td>
<td></td>
</tr>
<tr>
<td><code>value = phaseRef.UnknownMixerLOPower</code> 'Read'</td>
<td></td>
</tr>
</tbody>
</table>

**C++ Syntax**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT get_UnknownMixerLOPower(double *value)</td>
<td></td>
</tr>
<tr>
<td>HRESULT put_UnknownMixerLOPower(double value)</td>
<td></td>
</tr>
</tbody>
</table>

**Interface**
IPhaseReference2
UseInternalCalFactors Property

Description
Enables/disables use of internal calibration factors for power sensors with built-in calibration factors and reads the current state.

VB Syntax
pwrSensor.UseInternalCalFactors = state

Variable
pwrSensor
(Type) - Description
A PowerSensorAsReceiver (object)

state
(boolean)
False - Disables the use of internal calibration factors.
True - Enables the use of internal calibration factors.

Return Type
Boolean
Default
False

Examples
' This example script demonstrates the set/get of the 'UseInternalCalFactors' property
' on the PMAR PowerSensor COM object for an existing PMAR named 'MyPMAR'.
Option Explicit
dim app
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset
app.ActiveChannel.Hold True
app.ActiveChannel.CWFrequency = 1E9
app.ActiveChannel.SweepType = 3 'naCWTimeSweep
app.ActiveChannel.NumberOfPoints = 3
dim externalDevices
Set externalDevices = app.ExternalDevices
dim externalDevice
Set externalDevice = externalDevices.Item("MyPMAR")
externalDevice.Active = True
'Create a PMAR trace with power sensor connected to port 3
app.ActiveMeasurement.ChangeParameter "MyPMAR", 3
dim PMAR
Set PMAR = externalDevice.ExtendedProperties
' Disable use of the sensor's internal cal factors, take a sweep and report the Mean
PMAR.UseInternalCalFactors = False
externalDevice.IOEnable = True
app.ActiveChannel.Single True
MsgBox "UseInternalCalFactors = " & PMAR.UseInternalCalFactors & ", Mean measured val = " & app.ActiveMeasurement.Mean
' Enable use of the sensor's internal cal factors, take another sweep and report the Mean again
externalDevice.IOEnable = False
PMAR.UseInternalCalFactors = True
externalDevice.IOEnable = True
app.ActiveChannel.Single True
MsgBox "UseInternalCalFactors = " & PMAR.UseInternalCalFactors & ", Mean measured val = " & app.ActiveMeasurement.Mean

C++ Syntax

```cpp
HRESULT get_UseInternalCalFactors(VARIANT_BOOL *pVal)
HRESULT put_UseInternalCalFactors(VARIANT_BOOL Val)
```

Interface

IPowerSensorAsReceiver3
VisaResourceString Property

Description
Returns the valid resource string for the specified interface type.

Use InterfaceTypes Property to read the valid interface types for the VNA.

VB Syntax
value = IOConfig.VisaResourceString (InterfaceType)

Variable (Type) - Description
value
(String) variable to store the returned information.

IOConfig
An IOConfiguration object.

InterfaceType
(String) Name of Interface Type

ReturnType
String

Default
Not Applicable

Examples
value = IOConfig.VisaResourceString "HISLIP"
returns: TCPIP0::<compID>::hislip0::INST

C++ Syntax
HRESULT get VisaResourceString(BSTR interfaceType, long visaResourceID, BSTR* resourceString);

Interface
IIOConfiguration
XAxisSource Property

Description
Returns the X-Axis source of the selected measurement.

See Also:
ActiveXAxis Property
XAxisDomain Property

VB Syntax
\[ src = 
\]

Variable (Type) - Description
\( src \) (String) - Variable to store the returned source for the domain that is displayed on the X-axis. See ActiveXAxis Property for possible returned values.

DIQMeas A DIQMeas (object)

Example
\[
src = \text{diqMeas.XAxisSource} \quad \text{'Read}
\]
\[
\text{'Returns "Port 1"}
\]

C++ Syntax
HRESULT get_XAxisSource(BSTR* source);

Interface
IDIQMeas

7973
Data Access Map
dataMapLarge
# CFData Topic

**File** | **Instrument** | **Response** | **Stimulus** | **Utility** | **Cal** | **Apps** | **Remote ONLY**
---|---|---|---|---|---|---|---

### Get and Put Data:
- Measurement
- Cal
- Power Cal
- Custom
- Power Range
- GPIB Pass-through
- VISA Pass-through
- Capabilities
- Status/Events
- Rear-panel
- FIFO and FastCW
- Speed up Measurements!
- Ground Loop

### Description

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get X-Axis values (variant)</td>
<td>CALCulate:MEASure:X:VALues?</td>
<td>Get X-axis Values</td>
</tr>
<tr>
<td>Get X-Axis values (typed)</td>
<td>None</td>
<td>Get X-Axis Values 2</td>
</tr>
<tr>
<td>Set/get X-axis for trace</td>
<td>CALCulate:MEASure:X:AXIS</td>
<td>None</td>
</tr>
<tr>
<td>Set/get the X-Axis domain</td>
<td>CALCulate:MEASure:X:AXIS:DOMain</td>
<td>None</td>
</tr>
<tr>
<td>Get X-Axis values (Meas object)</td>
<td>None</td>
<td>Get XAxisValues</td>
</tr>
</tbody>
</table>

### Get Measurement Data FROM the Analyzer

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get typed complex data from the specified location.</td>
<td>None</td>
<td>IArrayTrans.getComplex</td>
</tr>
<tr>
<td>Returned in two arrays.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Get typed NAComplex data from the specified location.</td>
<td>None</td>
<td>IArrayTrans.getNAComplex</td>
</tr>
<tr>
<td>Returned in one array.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Get typed data pairs from the specified location.</td>
<td>None</td>
<td>IArrayTrans.getPairedData</td>
</tr>
<tr>
<td>Returned in two arrays.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Get typed scalar data from the specified location.</td>
<td>None</td>
<td>IArrayTrans.getScalar</td>
</tr>
<tr>
<td>Returned in one array.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Get variant data from the specified location in a SPECIFIED FORMAT.</td>
<td>None</td>
<td>meas.GetDataByString</td>
</tr>
<tr>
<td>Returned in one array.</td>
<td>Get receiver data</td>
<td>CALCulate:MEASure:RDATA?</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Specifies ASCII or REAL type for data transfers</td>
<td>Format:Data</td>
<td>None</td>
</tr>
<tr>
<td>Get complex or formatted data from the measurement or memory result buffer</td>
<td>CALCulate:MEASure:DATA</td>
<td>All of the above</td>
</tr>
<tr>
<td>Get the formatted data array of multiple traces of the selected channel.</td>
<td>CALCulate:DATA:MFData</td>
<td>None</td>
</tr>
<tr>
<td>Get the corrected data array of multiple traces of the selected channel.</td>
<td>CALCulate:DATA:MSData</td>
<td>None</td>
</tr>
<tr>
<td>Gets SnP data for the specified ports.</td>
<td>CALCulate:DATA:SNP:PORTs?</td>
<td>GetSnpDataWithSpecifiedPorts</td>
</tr>
</tbody>
</table>

**Put Measurement Data INTO the Analyzer**

| Put complex data into the specified location. | None | IArrayTrans.putComplex |
| Put typed NAComplex data into the specified location. | None | IArrayTrans.putNAComplex |
| Put scalar data into the measurement result location. | None | IArrayTrans.putScalar |
| Put complex Variant data into the specified location. | None | IArrayTrans.putDataComplex |
| Put complex or formatted data into the measurement or memory result buffer | CALCulate:MEASure:DATA | None |

**Get Calibration Data FROM the Analyzer**
| Get complex Error Term data | None | Get ErrorTermComplexByString |
| Get variant Error Term data | SENSe:CORRection:CSET:DATA CSET:ETERm[:DATA]? | GetErrorTermByString |
| Get variant Error Term data by text filter | CSET:ETERm:CATalog? | Get ErrorTermList2 |
| Get complex Standard data | None | GetStandardComplexByString |
| Get variant Standard data | None | GetStandardByString |
| Get variant Standard data by text filter | None | Get StandardList2 |

**Put Calibration Data INTO the Analyzer**

| Put complex Error Term data | None | PutErrorTermComplexByString |
| Put variant Error Term data | SENSe:CORRection:CSET:DATA CSET:ETERm[:DATA] | PutErrorTermByString |
| Put complex Standard data | None | PutStandardComplexByString |
| Put variant Standard data | None | PutStandardByString |

**Power Calibration Data**

| Get variant cal data | SOURce:POWer:CORRection:DATA | Get SourcePowerCalDataEx |
| Get typed cal data | SOURce:POWer:CORRection:DATA | Get SourcePowerCalDataScalarEx |
| Put variant cal data | SOURce:POWer:CORRection:DATA | Put SourcePowerCalDataEx |
| Put typed cal data | SOURce:POWer:CORRection:DATA | Put SourcePowerCalDataScalarEx |

**Get and Put Custom Measurement Data**

| Get and Put Custom data | CALCulate:MEASure:DATA | IArrayTransfer2 Interface |

**Capabilities**

7978
Many queries regarding the capability of a specific PNA

<table>
<thead>
<tr>
<th>PXIe module queries</th>
<th>SYSTem:CAPability:HARDware:MODule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read installed options</td>
<td>*Opt?</td>
</tr>
</tbody>
</table>

### Power Range

<table>
<thead>
<tr>
<th>Set/get list of discrete frequencies corresponding to powers</th>
<th>SYSTem:CAPability:HARDware:POWer:DISCrete:FREQuency:LIST</th>
<th>DiscreteFrequencies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set/get name of the value for the given path element name</td>
<td>SYSTem:CAPability:HARDware:POWer:PATH:CONFig:ELEMent[:STATE]</td>
<td>PathElement</td>
</tr>
<tr>
<td>Get all RF path element names</td>
<td><code>SYSTem:CAPability:_HARDware:POWer:PATH:CONFig:ELEMent:CATalog?</code></td>
<td>PathElements</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Set/get port number for power data</td>
<td><code>SYSTem:CAPability:_HARDware:POWer:PORT</code></td>
<td>PortNumber</td>
</tr>
<tr>
<td>Set/get type of power range data to be returned</td>
<td><code>SYSTem:CAPability:_HARDware:POWer:TYPE</code></td>
<td>PowerRangeType</td>
</tr>
<tr>
<td>Get minimum of all max leveled power values</td>
<td><code>SYSTem:CAPability:_HARDware:POWer:RANGe:MAXimum?</code></td>
<td>RangeGetMaxPower</td>
</tr>
<tr>
<td>Get maximum of all minimum power values</td>
<td><code>SYSTem:CAPability:_HARDware:POWer:RANGe:MINimum?</code></td>
<td>RangeGetMinPower</td>
</tr>
<tr>
<td>Set/get lower bound of the frequency range</td>
<td><code>SYSTem:CAPability:_HARDware:POWer:RANGe:FREQuency:STARt</code></td>
<td>RangeStartFrequency</td>
</tr>
<tr>
<td>Set/get upper bound of the frequency range</td>
<td><code>SYSTem:CAPability:_HARDware:POWer:RANGe:FREQuency:STOP</code></td>
<td>RangeStopFrequency</td>
</tr>
<tr>
<td>Reset all Power Range properties to default values</td>
<td><code>SYSTem:CAPability:_HARDware:POWer:RESet</code></td>
<td>Reset (Power Range)</td>
</tr>
</tbody>
</table>

**Status Commands**

**Status Registers**

| GP-IB/Status | None |
### Events

<table>
<thead>
<tr>
<th>Method</th>
<th>None</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowAllEvents Method</td>
<td>None</td>
<td>app.AllowAllEvents</td>
</tr>
<tr>
<td>AllowEventCategory Method</td>
<td>None</td>
<td>app.AllowEventCategory</td>
</tr>
<tr>
<td>AllowEventMessage Method</td>
<td>None</td>
<td>app.AllowEventMessage</td>
</tr>
<tr>
<td>AllowEventSeverity Method</td>
<td>None</td>
<td>app.AllowEventSeverity</td>
</tr>
<tr>
<td>DisallowAllEvents Method</td>
<td>None</td>
<td>app.DisallowAllEvents</td>
</tr>
<tr>
<td>MessageText Method</td>
<td>None</td>
<td>app.MessageText</td>
</tr>
<tr>
<td>OnCalEvent</td>
<td>None</td>
<td>app.OnCalEvent</td>
</tr>
<tr>
<td>OnChannelEvent</td>
<td>None</td>
<td>app.OnChannelEvent</td>
</tr>
<tr>
<td>OnDisplayEvent</td>
<td>None</td>
<td>app.OnDisplayEvent</td>
</tr>
<tr>
<td>OnHardwareEvent</td>
<td>None</td>
<td>app.OnHardwareEvent</td>
</tr>
<tr>
<td>OnMeasurementEvent</td>
<td>None</td>
<td>app.OnMeasurementEvent</td>
</tr>
<tr>
<td>OnSCPIEvent</td>
<td>None</td>
<td>app.OnSCPIEvent</td>
</tr>
<tr>
<td>OnSystemEvent</td>
<td>None</td>
<td>app.OnSystemEvent</td>
</tr>
<tr>
<td>OnUserEvent</td>
<td>None</td>
<td>app.OnUserEvent</td>
</tr>
<tr>
<td>SetFailOnOverRange</td>
<td>None</td>
<td>app.SetFailOnOverRange</td>
</tr>
</tbody>
</table>

### Rear Panel Connector Controls

<table>
<thead>
<tr>
<th>Connector Controls</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Material Handler I/O Connector</td>
<td>GP-IB/Control</td>
<td>MaterialHandler IO</td>
</tr>
<tr>
<td>Auxiliary IO Connector</td>
<td>GP-IB/Control</td>
<td>Aux IO</td>
</tr>
<tr>
<td>External Test Set Connector</td>
<td>GP-IB/Control</td>
<td>External Test Set</td>
</tr>
</tbody>
</table>
### FIFO Data Buffer (N5264A Only)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIFO ON</td>
<td>OFF</td>
<td>SYSTem:FIFO[:STATe]</td>
</tr>
<tr>
<td>Read number of data points</td>
<td>SYSTem:FIFO:DATA:COUNT?</td>
<td>DataCount</td>
</tr>
<tr>
<td>Read data</td>
<td>SYSTem:FIFO:DATA?</td>
<td>Data</td>
</tr>
<tr>
<td>Read data compact form</td>
<td>None</td>
<td>DataInCompactForm</td>
</tr>
<tr>
<td>Clear data</td>
<td>SYSTem:FIFO:DATA:CLEar</td>
<td>Clear</td>
</tr>
<tr>
<td>Returns a specific number of bytes to read</td>
<td>SYST:FIFO:DATA:BYTe?</td>
<td>DataAsBytes</td>
</tr>
<tr>
<td>Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of bytes.</td>
<td>SYST:FIFO:DATA:BYTe:COUNT?</td>
<td>DataByteCount</td>
</tr>
<tr>
<td>Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of 32-bit floating point (Float32) numbers.</td>
<td></td>
<td>DataAsFloat32</td>
</tr>
<tr>
<td>Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of 16-bit integers.</td>
<td></td>
<td>DataAsInt16</td>
</tr>
<tr>
<td>Reads the FIFO buffer data as a Variant of a specified array size (SafeArray) of 32-bit integers.</td>
<td></td>
<td>DataAsInt32</td>
</tr>
</tbody>
</table>

### Other N5264A Commands

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FastCW</td>
<td>SENSe:SWEep:TYPE:FACW</td>
<td>FastCWPointCount</td>
</tr>
<tr>
<td>Enable Point Averaging</td>
<td>SENSe:AVERage:MODE</td>
<td>AverageMode</td>
</tr>
<tr>
<td>Enable Point Sweep</td>
<td>SENSe:SWEep:GENeration:POINtsweep</td>
<td>PointSweepState</td>
</tr>
<tr>
<td>Ground Loop De-embedding/Embedding commands</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>De-embedding</strong></td>
<td><strong>Embedding</strong></td>
<td></td>
</tr>
<tr>
<td>Sets and returns the Capacitance value</td>
<td>Sets and returns the Capacitance value</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:GLOop:DEEMbed:PARameters:C</td>
<td></td>
</tr>
<tr>
<td>Sets and returns the Inductance value</td>
<td>Sets and returns the Inductance value</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:GLOop:DEEMbed:PARameters:L</td>
<td></td>
</tr>
<tr>
<td>Sets and returns the Resistance value</td>
<td>Sets and returns the Resistance value</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:GLOop:DEEMbed:PARameters:R</td>
<td></td>
</tr>
<tr>
<td>Turns ON or OFF De-embedding</td>
<td>Turns ON or OFF Embedding</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:GLOop:DEEMbed:STATe</td>
<td></td>
</tr>
<tr>
<td>Specifies the circuit model type</td>
<td>Specifies the circuit model type</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:GLOop:DEEMbed:TYPE</td>
<td></td>
</tr>
<tr>
<td>Specifies the filename of the s1p file to load</td>
<td>Specifies the filename of the s1p file to load</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:FSIMulator:GLOop:DEEMbed:USER:FILename</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>-----------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td><strong>Save / Recall</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Save Instrument States (*.csa, *.cst, *.sta, *.cal) and type of file</td>
<td>MMEMory:STORe</td>
<td>app.Save</td>
</tr>
<tr>
<td>Save Data (except snp)</td>
<td>MMEMory:STORe:DATA</td>
<td>app.SaveData</td>
</tr>
<tr>
<td>Recall Files</td>
<td>MMEMory:LOAD</td>
<td>app.recall</td>
</tr>
<tr>
<td>Recall softkey list sort preference</td>
<td>SYSTem:PREFerences:ITEM:MRU</td>
<td>RecallSoftkeysMostRecent</td>
</tr>
<tr>
<td><strong>Reads SNP data for the specified ports</strong></td>
<td>CALCulate:MEASure:DATA:SNP:PORTs?</td>
<td>Get SnpDataWithSpecifiedPorts</td>
</tr>
<tr>
<td>Saves SNP data for the specified ports</td>
<td>CALCulate:MEASure:DATA:SNP:PORTs:SAVE</td>
<td>WriteSnpFileWithSpecifiedPorts</td>
</tr>
<tr>
<td>Reads SnP data from the selected measurement</td>
<td>CALCulate:MEASure:DATA:SNP?</td>
<td>GetSnPData</td>
</tr>
<tr>
<td>Sets format for .SNP files</td>
<td>MMEMory:STORe:TRACe:FORMat:SNP</td>
<td>pref.SnPFormat</td>
</tr>
<tr>
<td>Set/get formatted measurement data</td>
<td>CALCulate:MEASure:DATA:FDATa</td>
<td>getData</td>
</tr>
<tr>
<td></td>
<td></td>
<td>putDataComplex</td>
</tr>
<tr>
<td>Set/get complex measurement data</td>
<td>CALCulate:MEASure:DATA:SDATa</td>
<td>getData</td>
</tr>
<tr>
<td></td>
<td></td>
<td>putDataComplex</td>
</tr>
<tr>
<td>Set/get formatted memory data</td>
<td>CALCulate:MEASure:DATA:FMEMory</td>
<td>getData</td>
</tr>
<tr>
<td></td>
<td></td>
<td>putDataComplex</td>
</tr>
<tr>
<td>Set/get complex memory data</td>
<td>CALCulate:MEASure:DATA:SMEMory</td>
<td>getData</td>
</tr>
<tr>
<td></td>
<td></td>
<td>putDataComplex</td>
</tr>
<tr>
<td><strong>Manage Files</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>List Files</td>
<td>MMEMory:CATalog</td>
<td>None</td>
</tr>
<tr>
<td>Copy Files</td>
<td>MMEMory:COPY</td>
<td>None</td>
</tr>
<tr>
<td>Move Files</td>
<td>MMEMory:MOVE</td>
<td>None</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Delete Files</td>
<td>MMEMory:DELeTe</td>
<td>None</td>
</tr>
<tr>
<td><strong>Manage Folders</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change</td>
<td>MMEMory:CDIRectory</td>
<td>None</td>
</tr>
<tr>
<td>Delete</td>
<td>MMEMory:RDIRectory</td>
<td>None</td>
</tr>
<tr>
<td>Make</td>
<td>MMEMory:MDIRectory</td>
<td>None</td>
</tr>
<tr>
<td>Read directory location for the specified file type</td>
<td>SYSTem:CONFigure:DIRectory?</td>
<td>DirectoryPath</td>
</tr>
<tr>
<td><strong>Print</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Print</td>
<td>HCOPy</td>
<td>app.DoPrint</td>
</tr>
<tr>
<td>Saves image of VNA screen to file. (Print to File)</td>
<td>HCOPy:FILE</td>
<td>app.PrintToFile</td>
</tr>
<tr>
<td>Return the display image in arbitrary binary block</td>
<td>HCOPy:SDUMp:DATA?</td>
<td>None</td>
</tr>
<tr>
<td>Set format of display image</td>
<td>HCOPy:SDUMp:DATA:FORM</td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Read Date and Time</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read the last modified date of a Cal Set</td>
<td>CSET DATE?</td>
<td>LastModified</td>
</tr>
<tr>
<td>Read the last modified time of a Cal Set</td>
<td>CSET:TIME?</td>
<td>LastModified</td>
</tr>
<tr>
<td>Read the last modified date of a file</td>
<td>MMEM:DATE?</td>
<td>None</td>
</tr>
<tr>
<td>Read the last modified time of a file</td>
<td>MMEM:TIME?</td>
<td>None</td>
</tr>
</tbody>
</table>
DIQ

Calculate:DIQ:XAXis Commands

Allows you to set the DIQ channel X-Axis domain and source.

Click to view the command details.

**see Also**

Sense:DIQ commands
Example Programs
Learn about Differential IQ Application
Synchronizing the Analyzer and Controller
SCPI Command Tree

```
CALCulate<ch>:DIQ:XAXis <domain>,<source>
```

Applicable Models: N522xB, N524xB

*(Write-only)* Sets the information to display on the X-axis of the selected DIQ measurement. This command does not change the default setting for new traces.

**Parameters**

- `<ch>`
  - *The Differential IQ channel number. If unspecified, value is set to 1.*

- `<domain>`
  - Character - Domain to display on the X-axis. Choose from:
    - FREQuency - display the primary frequency range
    - POWER - display the power sweep range
    - PHASe - display the phase sweep range
    - DCValue - display the DC sweep range
    - POINTs - display the data points in the range

- `<source>`
  - String - Specific source for the selected domain. Choose from the following:
For this X-Axis Domain:  | Choose one of these X-Axis Sources
---|---
Frequency | Frequency Range (F1, F2, etc.)
Power | Source port (Port 1, Port 2, etc.)
Phase | Source port (Port 1, Port 2, etc.)
DC Value | DC Source (AO1, AO2, etc.)
Points | Points (N/A)

Examples

CALC:DIQ:XAX FREQ,"F2"
calculate:diq:xaxis power,"Port 1"

Query Syntax
Not Applicable

Default
FREQ,"F1"

CALCulate<ch>:DIQ:XAXis:DOMain?

Applicable Models: N522xB, N524xB

(Read-only) Returns the X-Axis domain of the selected measurement.

Parameters

<ch>
The Differential IQ channel number. If unspecified, value is set to 1.

Example

CALC:DIQ:XAX:DOM?

'Possible returned values:
FREQuency - display the primary frequency range.
POWer - display the power sweep range
PHASe - display the phase sweep range
DCValue - display the DC sweep range
POINts - display the data points in the range

Return Type
Character

Default
FREQuency
CALCulate<ch>:DIQ:XAXis:SOURce?

Applicable Models: N522xB, N524xB

(Read-only) Returns the X-Axis source of the selected measurement.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>The Differential IQ channel number. If unspecified, value is set to 1.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:DIQ:XAX:SOURce?</td>
</tr>
<tr>
<td>See CALC:DIQ:XAX for possible returned values.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;F1&quot;</td>
</tr>
</tbody>
</table>
FSimulatorBalun

CALCulate:FSIMulator:BALun:

- BPORt
- OFFSet
- PHASe
- POWer
- STIMulus
- TRUe
- STATe
- SWEep
- PHASe
- STARt
- STOP

CZConversion
- BPORt
- IMAG
- REAL
- Z0
- R

- STATe
- LPOrt
- IMAG
- REAL
- Z0
- R

DEVice

DMCircuit
- BPORt
- PARameters
- C
- G
- L
- R
- TYPE
- USER
Click on a keyword to view the command details.

see Also

- Example Programs
- Learn about Balanced Measurements
- Learn about iTMSA
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

Notes:
Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

CALC:PAR:CAT? alone can NOT be used to return a balanced measurement parameter. If a balanced measurement transform is being performed, then additional querying of the CALC:FSIM system is required to determine the balanced parameter type. See an example.

BPORt versus LPORt commands - For each command in this subsystem that includes a BPORt keyword, there is an LPORt equivalent. The commands are identical except for the way in which the balanced / logical port numbers are specified:

The BPORt commands refer to the Balanced port number. There can only be up to two balanced ports. This method is compatible with the ENA network analyzer.

The LPORt commands refer to the Logical port number. A balanced port can appear as either logical port 1, 2, or 3. These are the references as they appear in the front-panel user interface.

<table>
<thead>
<tr>
<th>Topology</th>
<th>Logical Port</th>
<th>Balanced Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-Bal</td>
<td>1</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Single-Single-Bal</td>
<td>1</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Bal-Bal</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Learn more about logical ports.

CALCulate<cnum>:FSIMulator:BALun:BPORt<pnum>:OFFSet:PHASE <value>

Applicable Models: All
(Read-Write) Sets the phase offset between the two balanced stimulus ports. This command only applies when CALC:FSIM:BAL:STIM:MOD is set to a True Mode - Not Single-Ended. Requires Opt S93460A/B. Learn more about iTMSA Power and Phase offset.
See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
Balanced port number. Choose from 1 to 999.

Phase offset value in degrees.

**Examples**

```
CALC:FSIM:BAL:BPOR:OFFS:PHAS 10
calculate2:fsimulator:balun:bport:offset:phase 300
```

See example iTMSA program

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:BALun:BPORt<pnum>:OFFSet:PHASe?
```

**Return Type**

Numeric

**Default**

0

---

**CALCulate<cnum>:FSIMulator:BALun:BPORt<pnum>:OFFSet:POWer <value>**

Applicable Models: All

(Read-Write) Sets the power offset between the two balanced stimulus ports. This command only applies when CALC:FSIM:BAL:STIM.MOD is set to a True Mode - Not Single-Ended. Requires Opt S93460A/B. Learn more about iTMSA Power and Phase offset.

See Critical Note

**Parameters**

**<cnum>**

Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

**<pnum>**

Balanced port number. Choose from 1 to 999.

**<value>**

Power offset value in dB.

**Examples**

```
CALC:FSIM:BAL:BPOR:OFFS:POW 2
calculate2:fsimulator:balun:bport:offset:power .2
```

See example iTMSA program

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:BALun:BPORt<pnum>:OFFSet:POWer?
```

**Return Type**

Numeric

**Default**

0

---

**CALCulate<cnum>:FSIMulator:BALun:BPORt<pnum>:STIMulus:TRUe:STATe <bool>**
Applicable Models: All
(Read-Write) Sets and reads the True Mode state for a specified balanced port.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum> Balanced port number. Choose from 1 to 999.

<bool> Choose from:
OFF (or 0) Turns True Mode OFF
ON (or 1) Turns True Mode ON

Examples

CALC:FSIM:BAL:BPOR2:STIM:TRU:STAT 1
calculate2:fsimulator:balun:bport2:stimulus:state off

Query Syntax
CALCulate<cnum>:FSIMulator:BALun:BPORt<pnum>:STIMulus:TRUe:STATe?

Return Type Boolean

Default Off

CALCulate<cnum>:FSIMulator:BALu:BPORt<pnum>:SWEep:PHASe:STARt <value>

Applicable Models: All
(Read-Write) Sets the start value for a phase sweep.
Learn more about Phase Sweep.
See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum> Balanced port number. Choose any VNA port (1 to 999). Only one port can have phase sweep.

<value> Phase sweep start value in degrees. Choose a value between -360 and 360.

Examples

**CALCulate<cnum>:FSIMulator:BALun:BPORt<pnum>:SWEep:PHASe:STARt**?

**Return Type** Numeric

**Default** 0

---

**CALCulate<cnum>:FSIMulator:BALun:BPORt<pnum>:SWEep:PHASe:STOP <value>**

Applicable Models: All

(Read-Write) Sets the stop value for a phase sweep.

Learn more about Phase Sweep.

See Critical Note

**Parameters**

- **<cnum>**
  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

- **<pnum>**
  Balanced port number. Choose any VNA port (1 to 999). Only one port can have phase sweep.

- **<value>**
  Phase sweep stop value in degrees. Choose a value between -360 and 360.

**Examples**

```
```

```
```

---

**CALCulate<cnum>:FSIMulator:BALun:CZConversion:BPORt<pnum>:IMAG <value>** Superseded

Applicable Models: All


(Read-Write) Sets the imaginary part of the impedance value for the common port impedance conversion function.

See Critical Note

**Parameters**

---

7995
<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum> Balanced port number. Choose from 1 to 999.

Note: The numbering of logical ports is different from balanced ports. This command works the same as the ENA network analyzer. If there is only one balanced port, it is Balanced Port 1, regardless of the port mapping assignment. Learn more.

)value> Imaginary part of the Impedance value in Units. Choose a number between 0 and 1E18.

Examples

```
CALC:FSIM:BAL:CZC:BPOR:IMAG 0
```

```
calculate2:fsimulator:balun:czconversion:bport:imag 300
```

Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:CZConversion:BPORt<pnum>:IMAG?
```

Return Type Numeric

Default 0
CALCulate<cnum>:FSIMulator:BALun:CZConversion:BPORt<pnum>:Z0[:R] <value> Superseded

Applicable Models: All


(Read-Write) Sets the real part of the impedance value for the common port impedance conversion function. Set either this single value or set the real and imaginary parts separately. The imaginary part is set to 0.0 using this command.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum> The number of balanced ports. For example, if the device topology was SE-BAL-SE-BAL, then this configuration would have 6 logical ports and 3 balanced ports. Learn more about logical and balanced ports.

<value> Impedance value in ohms. Choose a number between 0 to 1E7.

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:BAL:ZC:BPORT:Z0 50</td>
<td></td>
</tr>
<tr>
<td>calculate2:fsimulator:balun:czconversion:bport:z0:r 75</td>
<td></td>
</tr>
</tbody>
</table>

Query Syntax

CALCulate<cnum>:FSIMulator::BALun:CZConversion:BPORt<pnum>:Z0[:R]?

Return Type

Numeric

Default

See Common Mode Port Z Conversion Default

---

CALCulate<cnum>:FSIMulator:BALun:CZConversion:STATE <bool> Superseded

Applicable Models: All


(Read-Write) Sets the common port impedance conversion function ON/OFF. Must also set the fixture simulator function to ON using CALC:FSIM:STAT.

See Critical Note

Parameters

---
### Channel number of the measurement

There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

### State of common port impedance conversion function

Choose from:
- **OFF** (or 0) Conversion OFF
- **ON** (or 1) Conversion ON

#### Examples

```
CALC:FSIM:BAL:CZC:STAT 1
calculate2:fsimulator:balun:czconversion:state off
```

#### Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:CZConversion:STATe?
```

#### Return Type

Boolean

#### Default

Off

### Superseded


#### Parameters

- **<cnum>**
  - Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

- **<pnum>**
  - Logical port number. Choose from 1 to 999.

  **Note:** See Balanced port versus Logical port.

- **<value>**
  - Imaginary part of the Impedance value in Units. Choose a number between 0 and 1E18.

#### Examples

```
CALC:FSIM:BAL:CZC:LPOR:IMAG 0
calculate2:fsimulator:balun:czconversion:lport:imag 300
```

#### Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:CZConversion:LPORt<pnum>:IMAG?
```

#### Return Type

Numeric

#### Default

0
**CALCulate<cnm>:FSIMulator:Balun:CZConversion:LPORt<pnm>:REAL <vlue>**  Superseded

Applicable Models: All


*Read-Write* Sets the real part of the impedance value for the common port impedance conversion function.

See Critical Note

### Parameters

**<cnm>**
Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnm>` is set to 1.

**<pnm>**
Logical port number. Choose from 1 to 999.

*Note: See Balanced port versus Logical port.*

**<vlue>**
Real part of the Impedance value in Units. Choose a number between 0 and 1E18.

### Examples

```
```

```
calculate2:fsimulator:balun:czconversion:lport:real 50
```

### Query Syntax

```
CALCulate<cnm>:FSIMulator:Balun:CZConversion:LPORt<pnm>:REAL?
```

### Return Type
Numeric

### Default
See Common Mode Port Z Conversion Default

---

**CALCulate<cnm>:FSIMulator:Balun:CZConversion:LPORt<pnm>:Z0[:R] <vlue>**  Superseded

Applicable Models: All


*Read-Write* Sets the real part of the impedance value for the common port impedance conversion function. Set either this single value or set the real and imaginary parts separately. The imaginary part is set to 0.0 using this command.

See Critical Note

### Parameters

**<cnm>**
Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnm>` is set to 1.

**<pnm>**
Logical port number. Choose from 1 to 999.

*Note: See Balanced port versus Logical port.*

---
**<value>**

Impedance value in ohms. Choose a number between 0 to 1E7.

**Examples**

```
cALC:FSIM:BAI\L:CAZC:LPOP:R 20 50
calculate2:fsimulator:balun:czconversion:1port:z0:r 75
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:BAI\Lun:CZConversion:LPOP<pnum>:Z0[:R]?

**ReturnType**

Numeric

**Default**

See Common Mode Port Z Conversion Default

---

**CALCulate<cnum>:FSIMulator:BAI\Lun:DEVice <char>**

Applicable Models: All

(Read-Write) Selects the device type for the balanced measurement. To map the device type logical ports to the VNA physical ports, use the CALCulate:DTOPology, CALC:FSIM:BAI\Lun:TOP:XXXXX command.

See Critical Note

### Parameters

- **<cnum>**
  
  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

- **<char>**
  
  - BALanced - 1 port balanced device (2 ports)
  - BBALanced - Balanced - Balanced device (4 ports).
  - BALSended - Balanced - Single-ended device (3 ports).
  - SBALanced - Single-ended - Balanced device (3 ports).
  - CUST - Define custom device type for systems with greater than 4 ports.

**Examples**

```
cALC:FSI\M:BAI\L:DEV SSB
calculate2:fsimulator:balun:device bbal
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:BAI\Lun:DEVice?

**ReturnType**

Character

**Default**

SBALanced

---

**CALCulate<cnum>:FSIMulator:BAI\Lun:D\MCircuit:BPOPrt<pnum>:PARameters:C <value>**

Superseded

Applicable Models: All

(Read-Write) Sets the Capacitance value of the differential matching circuit.

- **Parameters**
  - `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
  - `<pnum>`: Balanced port number. Choose from 1 to 999.
  - `<value>`: Capacitance value in farads. Choose a number between -1E18 to 1E18

- **Examples**
  ```
  ```

- **Query Syntax**
  ```
  CALCulate<cnum>:FSIMulator:BALun:DMCircuit:BPORt<pnum>:PARameters:C?
  ```

- **Return Type** Numeric

- **Default** 0

---


(Read-Write) Sets the Conductance value of the differential matching circuit.

- **Parameters**
  - `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
  - `<pnum>`: Balanced port number. Choose from 1 to 999.

- **Note**: The numbering of logical ports is different from balanced ports. This command works the same as the ENA network analyzer. If there is only one balanced port, it is Balanced Port 1, regardless of the port mapping assignment. Learn more.

- **Examples**
  ```
  CALC:FSIM:BAL:DMC:BPOR:PARameters:G <value>
  ```

- **Applicable Models**: All

- **Superseded**

- **Query Syntax**
  ```
  CALCulate<cnum>:FSIMulator:BALun:DMCircuit:BPORt<pnum>:PARameters:G <value>
  ```

- **Return Type** Numeric

- **Default** None

---
### Conductance value in siemens

Choose a number between $-1E18$ to $1E18$.

#### Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>calculate2:fsimulator:balun:dmcircuit:bport:parameters:g</code></td>
<td>1E-3</td>
</tr>
</tbody>
</table>

#### Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:DMCircuit:BPORt<pnum>:PARameters:G?
```

#### Return Type

Numeric

#### Default

0

---

### Inductance value in henries

Choose a number between $-1E18$ to $1E18$.

#### Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
</table>

#### Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:DMCircuit:BPORt<pnum>:PARameters:L?
```

#### Return Type

Numeric

#### Default

0

---

### Resistance value in ohms

Choose a number between $-1E18$ to $1E18$.

#### Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
</table>

#### Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:DMCircuit:BPORt<pnum>:PARameters:R?
```

#### Return Type

Numeric

#### Default

0

---

Applicable Models: All


(Read-Write) Sets the Inductance value of the differential matching circuit.

See Critical Note

---

### Parameters

#### <cnum>

Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

#### <pnum>

Balanced port number. Choose from 1 to 999.

Note: The numbering of logical ports is different from balanced ports. This command works the same as the ENA network analyzer. If there is only one balanced port, it is Balanced Port 1, regardless of the port mapping assignment. Learn more.

#### <value>

Inductance value in henries. Choose a number between $-1E18$ to $1E18$.
### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
<tr>
<td>&lt;pnum&gt;</td>
<td>Balanced port number. Choose from 1 to 999. Note: The numbering of logical ports is different from balanced ports. This command works the same as the ENA network analyzer. If there is only one balanced port, it is Balanced Port 1, regardless of the port mapping assignment. Learn more.</td>
</tr>
<tr>
<td>&lt;value&gt;</td>
<td>Resistance value in ohms. Choose a number between -1E18 to 1E18.</td>
</tr>
</tbody>
</table>

### Examples

```
CALC:FSIM:BAL:DMC:BPORT:PARAMETERS:R 100
```

### Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:DMCircuit:BPORT<pnum>:PARAMeters:R?
```

### Return Type

Numeric

### Default

0

---

### Applicable Models: All


(Read-Write) Sets the Resistance value of the differential matching circuit.

See Critical Note

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
<tr>
<td>&lt;pnum&gt;</td>
<td>Balanced port number. Choose from 1 to 999.</td>
</tr>
</tbody>
</table>

---

### Applicable Models: All


(Read-Write) Sets the differential matching circuit type. To select a user-defined circuit, specify IN ADVANCE the 2-port touchstone filename with `CALC:FSIM:BAL:DMC:BPORT:USER:FILename`. If you do not specify the appropriate file and you select USER, an error occurs and NONE is automatically selected.

See Critical Note

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
<tr>
<td>&lt;pnum&gt;</td>
<td>Balanced port number. Choose from 1 to 999.</td>
</tr>
</tbody>
</table>
Note: The numbering of logical ports is different from balanced ports. This command works the same as the ENA network analyzer. If there is only one balanced port, it is Balanced Port 1, regardless of the port mapping assignment. Learn more.

Circuit type. Choose from:

NONE - Specifies no-circuit.

PLPC - Specifies the circuit that consists of shunt L and shunt C.

USER - Specifies the user-defined circuit.

Examples

```
CALC:FSIM:BAL:DMC:BPOR2 PLPC
```

```
calculate2:fsimulator:balun:dmcircuit:bport1:type none
```

Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:DMCircuit:BPORt<pnum>:TYPE?
```

Return Type

Character

Default

PLPC

CALCulate<cnum>:FSIMulator:BALun:DMCircuit:BPORt<pnum>:USER:FILename <string> Superseded

Applicable Models: All


(Read-Write) Specifies the 2-port touchstone file in which the information on the user-defined differential matching circuit is saved. Following this command, send CALC:FSIM:BAL:DMC:BPOR2 USER. If the specified file does not exist, an error occurs when you set the type of differential matching circuit to USER.

See Critical Note

Parameters

- `<cnum>`
  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

- `<pnum>`
  Balanced port number. Choose from 1 to 999.

- `<string>`
  File name and extension (.s2P) of the differential matching circuit. Files are stored in the default folder "D:\". To recall from a different folder, specify the full path name.
CALCulate<cnum>:FSIMulator:BALun:DMCircuit:STATe <bool> Superseded

Applicable Models: All


(Read-Write) Sets the differential matching circuit embedding function ON/OFF. Must also set the fixture simulator function to ON using CALC:FSIM:STAT.

See Critical Note

Parameters

<cnum>
Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<bool>
State of differential matching circuit embedding function. Choose from

OFF (or 0) Matching circuit OFF
ON (or 1) Matching circuit ON

Examples
CALC:FSIM:BAL:DMC:STAT 1
calculate2:fsimulator:balun:dmcircuit:state off

Query Syntax
CALCulate<cnum>:FSIMulator:BALun:DMCircuit:STATe?

Return Type
Boolean

Default
Off

CALCulate<cnum>:FSIMulator:BALun:DMCircuit:LPORt<pnum>:PARameters:C <value> Superseded

Applicable Models: All


(Read-Write) Sets the Capacitance value of the differential matching circuit.
Parameters

<cnun> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnun> is set to 1.

<pnum> Logical port number. Choose from 1 to 999.

Note: See Balanced port versus Logical port.

<value> Capacitance value in farads. Choose a number between -1E18 to 1E18

Examples


Query Syntax

CALCulate<cnun>:FSIMulator:BALun:DMCircuit:LPORt<pnum>:PARameters:C?

Return Type

Numeric

Default

0

CALCulate<cnun>:FSIMulator:BALun:DMCircuit:LPORt<pnum>:PARameters:G <value> Superseded

Applicable Models: All


(Read-Write) Sets the Conductance value of the differential matching circuit.

See Critical Note

Parameters

<cnun> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnun> is set to 1.

<pnum> Logical port number. Choose from logical 1 to 999.

Note: See Balanced port versus Logical port.

<value> Conductance value in siemens. Choose a number between -1E18 to 1E18.

Examples

calculate2:fsimulator:balun:dmcircuit:lport:parameters:g 1E-3

Query Syntax

CALCulate<cnun>:FSIMulator:BALun:DMCircuit:LPORt<pnum>:PARameters:G?
CALCulate<cnum>:FSIMulator:BALun:DMCircuit:LPORt<pnum>:PARameters:L <value>

Applicable Models: All


(Read-Write) Sets the Inductance value of the differential matching circuit.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum> Logical port number. Choose from 1 to 999.

Note: See Balanced port versus Logical port.

[value] Inductance value in henries. Choose a number between -1E18 to 1E18.

Examples


Query Syntax

CALCulate<cnum>:FSIMulator:BALun:DMCircuit:LPORt<pnum>:PARameters:L?

Return Type Numeric

Default 0

CALCulate<cnum>:FSIMulator:BALun:DMCircuit:LPORt<pnum>:PARameters:R <value>

Applicable Models: All


(Read-Write) Sets the Resistance value of the differential matching circuit.

See Critical Note

Parameters
### Parameters

- **<cnum>**
  
  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

- **<pnum>**
  
  Logical port number. Choose from 1 to 999.

  **Note:** See Balanced port versus Logical port.

- **<value>**
  
  Resistance value in ohms. Choose a number between -1E18 to 1E18.

### Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:BAL:DMC:LPOR:PARameters:R 100</td>
<td>Set resistance to 100 ohms</td>
</tr>
<tr>
<td>calculate2:fsimulator:balun:dmcircuit:lport:parameters:r 4E3</td>
<td>Set resistance to 4E3 ohms</td>
</tr>
</tbody>
</table>

### Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:DMCircuit:LPORt<pnum>:PARameters:R?
```

### Return Type

Numeric

### Default

0

---

**CALCulate<cnum>:FSIMulator:BALun:DMCircuit:LPORt<pnum>[:TYPE] <char>** Superseded

*Applicable Models: All*


*(Read-Write)* Sets the differential matching circuit type. To select a user-defined circuit, specify IN ADVANCE the 2-port touchstone filename with `CALC:FSIM:BAL:DMC:LPOR:USER:FILename`. If you do not specify the appropriate file and you select USER, an error occurs and NONE is automatically selected.

See Critical Note

#### Parameters

- **<cnum>**
  
  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

- **<pnum>**
  
  Logical port number. Choose from 1 to 999.

  **Note:** See Balanced port versus Logical port.

- **<char>**
  
  Circuit type. Choose from:

  - **NONE** - Specifies no-circuit.
  - **PLPC** - Specifies the circuit that consists of shunt L and shunt C.
  - **USER** - Specifies the user-defined circuit.

#### Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:BAL:DMC:LPOR2 PLPC</td>
<td>Set PLPC circuit</td>
</tr>
<tr>
<td>calculate2:fsimulator:balun:dmcircuit:lport1:type none</td>
<td>Set type to none</td>
</tr>
</tbody>
</table>
### Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:DMCircuit:LPORt<pnum>:TYPE?
```

### Return Type

Character

### Default

PLPC

---

### Applicable Models: All

**Superseded**


*(Read-Write)* Specifies the 2-port touchstone file in which the information on the user-defined differential matching circuit is saved. Following this command, send `CALC:FSIM:BAL:DMC:BPOR2 USER`. If the specified file does not exist, an error occurs when you set the type of differential matching circuit to USER.

See Critical Note

### Parameters

- **<cnum>**
  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

- **<pnum>**
  Logical port number. Choose from 1 to 999.

  **Note:** See Balanced port versus Logical port.

- **<string>**
  File name and extension (.s2P) of the differential matching circuit. The default location where files are stored is "D:\". To recall from a different folder, specify the full path name.

### Examples

```
calculate2:fsimulator:balun:dmcircuit:lport:user:filename
"D:\myFile.s2P"
```

### Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:DMCircuit:LPORt<pnum>:USER:FILename <string> Superseded
```

### Return Type

String

### Default

Not Applicable

---

### Applicable Models: All

**Superseded**


```
CALCulate<cnum>:FSIMulator:BALun:DZConversion:BPORt<pnum>:IMAG <value> Superseded
```

Applicable Models: All

---

8009

(Read-Write) Sets the imaginary part of the impedance value for the differential port impedance conversion function.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum> Balanced port number. Choose from 1 to 999.

Note: The numbering of logical ports is different from balanced ports. This command works the same as the ENA network analyzer. If there is only one balanced port, it is Balanced Port 1, regardless of the port mapping assignment. Learn more.

<value> Imaginary part of the Impedance value in Units. Choose a number between 0 and 1E18.

Examples

```
CALC:FSIM:BL:DZC:BPOR:IMAG 0
calculate2:fsimulator:balun:dczconversion:bport:imag 300
```

Query Syntax

```
CALCulate<cnum>:FSIMulator:BLUn:DZConversion:BPORt<pnum>:IMAG?
```

Return Type Numeric

Default 0

---

`CALCulate<cnum>:FSIMulator:BLUn:DZConversion:BPORt<pnum>:REAL <value>` Superseded

Applicable Models: All


(Read-Write) Sets the real part of the impedance value for the differential port impedance conversion function.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum> Balanced port number. Choose from 1 to 999.

Note: The numbering of logical ports is different from balanced ports. This command works the same as the ENA network analyzer. If there is only one balanced port, it is Balanced Port 1, regardless of the port mapping assignment. Learn more.
<value> Real part of the Impedance value in Units. Choose a number between 0 and 1E18

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:BAL:dzc:bpor:real 50</td>
</tr>
<tr>
<td>calculate2:fsimulator:balun:dzconversion:bport:real 75</td>
</tr>
</tbody>
</table>

Query Syntax

CALCulate<cnum>:FSIMulator:BAUn:DZConversion:BPORt<pnum>:REAL?

Return Type

Numeric

Default

See Differential Port Z Conversion Default

---

CALCulate<cnum>:FSIMulator:BAUn:DZConversion:BPORt<pnum>:Z0[:R] <value> Superseded

Applicable Models: All

This command set of CALCulate:FSIMulator:BAUn:DZConversion:XXXX is changed to the set of
CALCulate:FSIMulator:DRAFT:ZConversion:DIFFerential:YYYY and

(Read-Write) Sets the impedance value for the differential port impedance conversion function. Set either this single value or set the real and imaginary parts separately. The imaginary part is set to 0.0 using this command.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum> Balanced port number. Choose from 1 to 999.

Note: The numbering of logical ports is different from balanced ports. This command works the same as the ENA network analyzer. If there is only one balanced port, it is Balanced Port 1, regardless of the port mapping assignment. Learn more.

<value> Impedance value in ohms. Choose a number between 0 to 1E7

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:BAL:dzc:bpor:20 50</td>
</tr>
<tr>
<td>calculate2:fsimulator:balun:dzconversion:bport:z0:r 75</td>
</tr>
</tbody>
</table>

Query Syntax

CALCulate<cnum>:FSIMulator:BAUn:DZConversion:BPORt<pnum>:Z0[:R]?

Return Type

Numeric

Default

See Differential Port Z Conversion Default

---

CALCulate<cnum>:FSIMulator:BAUn:DZConversion:STATe <bool> Superseded

8011

(Read-Write) Sets the differential port impedance conversion function ON/OFF. Must also set the fixture simulator function to ON using `CALC:FSIM:STAT`.

See Critical Note

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <code>&lt;cnum&gt;</code> is set to 1.</td>
</tr>
<tr>
<td><code>&lt;bool&gt;</code></td>
<td>State of the differential port impedance conversion function. Choose from OFF (0) Differential port impedance conversion OFF, ON (1) Differential port impedance conversion ON</td>
</tr>
</tbody>
</table>

Examples

```
CALC:FSIM:BAL:DZC:STAT 1
calculate2:fsimulator:balun:dzconversion:state off
```

Query Syntax

`CALCulate<cnum>:FSIMulator:BALun:DZConversion:STATe?`

Return Type

Boolean

Default

Off

---


(Read-Write) Sets the imaginary part of the impedance value for the differential port impedance conversion function.

See Critical Note

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <code>&lt;cnum&gt;</code> is set to 1.</td>
</tr>
<tr>
<td><code>&lt;pnum&gt;</code></td>
<td>Logical port number. Choose from 1 to 999. <strong>Note:</strong> See Balanced port versus Logical port.</td>
</tr>
<tr>
<td><code>&lt;value&gt;</code></td>
<td>Imaginary part of the Impedance value in Units. Choose a number between 0 and 1E18.</td>
</tr>
</tbody>
</table>
CALC:FSIM:BAL:DZC:LPOR:IMAG 0
calculate2:fsimulator:balun:dczconversion:lport:imag 300

Query Syntax
CALCulate<cnum>:FSIMulator:BALun:DZConversion:LPORt<pnum>:IMAG?

Return Type
Numeric

Default
0

CALCulate<cnum>:FSIMulator:BALun:DZConversion:LPORt<pnum>:REAL <value> Superseded

Applicable Models: All
This command set of CALCulate:FSIMulator:BALun:DZConversion:XXXX is changed to the set of
CALCulate:FSIMulator:DRAFt:ZCONversion:DIFFerential:YYYY and

(Read-Write) Sets the real part of the impedance value for the differential port impedance conversion function.
See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum> Logical port number. Choose from 1 to 999.

/value> Real part of the Impedance value in Units. Choose a number between 0 and 1E18

Examples

CALC:FSIM:BAL:DZC:LPOR:REAL 50
calculate2:fsimulator:balun:dczconversion:lport:real 75

Query Syntax
CALCulate<cnum>:FSIMulator:BALun:DZConversion:LPORt<pnum>:REAL?

Return Type
Numeric

Default See Differential Port Z Conversion Default

CALCulate<cnum>:FSIMulator:BALun:DZConversion:LPORt<pnum>:Z0[:R] <value> Superseded

Applicable Models: All
This command set of CALCulate:FSIMulator:BALun:DZConversion:XXXX is changed to the set of
CALCulate:FSIMulator:DRAFt:ZCONversion:DIFFerential:YYYY and
(Read-Write) Sets the impedance value for the differential port impedance conversion function. Set either this single value or set the real and imaginary parts separately. The imaginary part is set to 0.0 using this command.

See Critical Note

Parameters

<cnum>

Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum>

Logical port number. Choose from 1 to 999.

Note: See Balanced port versus Logical port.

/value>

Impedance value in ohms. Choose a number between 0 to 1E7

Examples

CALC:FSIM:BAL:DZC:LPOR:Z0 50
calculate2:fsimulator:balun:dzconversion:lport:z0:r 75

Query Syntax

CALCulate<cnum>:FSIMulator:BALun:DZConversion:LPORt<pnum>:Z0[:R]?

Return Type

Numeric

Default

See Differential Port Z Conversion Default

---

CALCulate<cnum>:FSIMulator:BALun:FIXTure:OFFSet:PHASe <bool>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA
(Read-Write) Sets and reads the state of "Phase Offset - Offset as Fixture" with True Mode balanced measurements. Learn more about iTMSA phase and power offset.

See Critical Note

Parameters

<cnum>

Channel number of the measurement. There must be a selected true mode balanced measurement on that channel. If unspecified, <cnum> is set to 1.

<bool>

State of phase Offset as Fixture.

OFF (or 0) Offset is applied but is NOT included as a fixture in the output calculations.

ON (or 1) Offset is applied and included as a fixture in the output calculations.

Examples

CALC:FSIM:BAL:FIXT:OFFS:PHAS 0
calculate2:fsimulator:balun:fixture:offset:phase on

See example iTMSA program

Query Syntax

CALCulate<cnum>:FSIMulator:BALun:FIXTureOFFSet:PHASe?
**CALCulate\(<cnum>\):FSIMulator:BA Lun:FIXTure:OFFSet:POWer <bool>**

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and reads the state of "Power Offset - Offset as Fixture" with True Mode balanced measurements.

Learn more about iTMSA phase and power offset.

See Critical Note

**Parameters**

\(<cnum>\) Channel number of the measurement. There must be a selected true mode balanced measurement on that channel. If unspecified, \(<cnum>\) is set to 1.

\(<bool>\) State of power Offset as Fixture.

**OFF** (or 0) Offset is applied but is NOT included as a fixture in the output calculations.

**ON** (or 1) Offset is applied and included as a fixture in the output calculations.

**Examples**

```
CALC:FSIM:BAL:FIXT:OFFS:POW 0
```

```
calculate2:fsimulator:balun:fixture:offset:power on
```

**Query Syntax**

```
CALCulate\(<cnum>\):FSIMulator:BA Lun:FIXTureOFFSet:POWer?
```

**Return Type** Boolean

**Default** Off

---

**CALCulate\(<cnum>\):FSIMulator:BA Lun:FIXTure:PHASe <bool>**

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets and reads the state of "Phase Sweep - Offset as Fixture" (labeling on GUI).

Learn more about iTMSA Phase Sweep.

See Critical Note

**Parameters**

\(<cnum>\) Channel number of the measurement. There must be a selected true mode balanced measurement on that channel. If unspecified, \(<cnum>\) is set to 1.
State of phase sweep offset as a fixture:

OFF (or 0)  Phase Sweep offset disabled.
ON (or 1)  Phase Sweep offset enabled.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate&lt;cnum&gt;:FSIMulator:BALun:FIXTure:PHASe?</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

Default

Off

CALCulate<cnum>:FSIMulator:BALun:LPORt<pnum>:OFFSet:PHASe <value>

Applicable Models: All

(Read-Write) Sets the phase offset between the two balanced stimulus ports. This command only applies when CALC:FSIM:BAL:STIM:MOD is set to a True Mode - Not Single-Ended. Requires Opt S93460A/B. Learn more about iTMSA Power and Phase offset.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
<pnum> Logical port number. Choose from 1 to 999.
<value> Phase offset value in degrees.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:BAL:LPOR:OFFS:PHAS 10</td>
<td>Numeric</td>
</tr>
<tr>
<td>calculate2:fsimulator:balun:lport:offset:phase 300</td>
<td></td>
</tr>
</tbody>
</table>

See example iTMSA program

CALCulate<cnum>:FSIMulator:BALun:LPORt<pnum>:OFFSet:POWer <value>

Applicable Models: All
(Read-Write) Sets the power offset between the two balanced stimulus ports. This command only applies when CALC:FSIM:BAL:STIM:MOD is set to a True Mode - Not Single-Ended. Requires Opt S93460A/B. Learn more about iTMSA Power and Phase offset.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum> Logical port number. Choose from 1 to 999.

<value> Power offset value in dB.

Examples

```
CALC:FSIM:BAL:LPOR:OFFS:POW 2
calculate2:fsimulator:balun:lport:offset:power .2
```

See example iTMSA program

Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:LPORt<pnum>:SWEep:PHASe:STARt <value>
```

Applicable Models: All

(Read-Write) Sets the start value for a phase sweep.

Learn more about Phase Sweep.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<pnum> Logical port number. Choose any VNA port (1 to 999). Only one port can have phase sweep.

<value> Phase sweep start value in degrees. Choose a value between -360 and 360.

Examples

```
```

Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:LPORt<pnum>:SWEep:PHASe:STARt?
```
Return Type: Numeric

Default: 0

**CALCulate<cnum>:FSIMulator: BALun:LPORt<pnum>:SWEep:PHASe:STOP <value>**

Applicable Models: All
(Read-Write) Sets the stop value for a phase sweep.
Learn more about Phase Sweep.
See Critical Note

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<pnum>`: Logical port number. Choose any VNA port (1 to 999). Only one port can have phase sweep.
- `<value>`: Phase sweep stop value in degrees. Choose a value between -360 and 360.

Examples

```plaintext
```

Query Syntax: `CALCulate<cnum>:FSIMulator: BALun:LPORt<pnum>:SWEep:PHASe:STOP?`

Return Type: Numeric

Default: 0

**CALCulate<cnum>:FSIMulator: BALun:PARameter<n>:BALSended[:DEFine] <char>**

Applicable Models: All
(Read-Write) For a Balanced-Single-ended device type, selects the measurement parameter for the specified trace.
Set device type using `CALC:FSIM: BAL:DEV`
See Critical Note

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
Trace number on the specified channel <cnum>

Balanced - Single-ended Measurement parameter. Choose from:

<table>
<thead>
<tr>
<th>Sdd11</th>
<th>Sdc11</th>
<th>Sds12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scd11</td>
<td>Scc11</td>
<td>Scs12</td>
</tr>
<tr>
<td>Ssd21</td>
<td>Ssc21</td>
<td>Sss22</td>
</tr>
<tr>
<td>Imb</td>
<td>CMRR1</td>
<td>CMRR2</td>
</tr>
<tr>
<td></td>
<td>(Ssd21/Ssc21)</td>
<td>(Sds12/Scs12)</td>
</tr>
</tbody>
</table>

Examples

```
CALC:FSIM:BAL:PAR:BALS SDC11
```

Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:PARameter<n>:BALSended[:DEFine]?
```

Return Type

Character

Default

Sdd11

---

**CALCulate<cnum>:FSIMulator:BALun:PARameter<n>:BALanced[:DEFine] <char>**

Applicable Models: All

(Read-Write) For a Balanced device type, selects the measurement parameter for the specified trace. Set device type using **CALC:FSIM:BAL:DEV**

See Critical Note

Parameters

- `<cnum>`
  
  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

- `<n>`
  
  Trace number on the specified channel `<cnum>`

- `<char>`
  
  Balanced Measurement parameter. Choose from:
  
  SDD11 - Specifies Sdd11.
  SCD11 - Specifies Scd11.
  SDC11 - Specifies Sdc11.
  SCC11 - Specifies Scc11.

Examples

```
CALC:FSIM:BAL:PAR:BAL SDD11
```
**CALCulate<cnum>:FSIMulator:BALun:PARameter<n>:BBALanced[:DEFine] <char>**

- **Applicable Models:** All
- **(Read-Write)** For a Balanced - Balanced device type, selects the measurement parameter for the specified trace. Set device type using CALC:FSIM:BAL:DEV
- **See Critical Note**

### Parameters

- **<cnum>**
  - Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

- **<n>**
  - Trace number on the specified channel <cnum>

- **<char>**
  - Balanced- Balanced Measurement parameter. Choose from:

<table>
<thead>
<tr>
<th>Sdd11</th>
<th>Sdd12</th>
<th>Sdc11</th>
<th>Sdc12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sdd21</td>
<td>Sdd22</td>
<td>Sdc21</td>
<td>Sdc22</td>
</tr>
<tr>
<td>Scd11</td>
<td>Scd12</td>
<td>Scc11</td>
<td>Scc12</td>
</tr>
<tr>
<td>Scd21</td>
<td>Scd22</td>
<td>Scc21</td>
<td>Scc22</td>
</tr>
<tr>
<td>Imb1</td>
<td>Imb2</td>
<td>CMRR</td>
<td>-(Sdd21/Scc21)</td>
</tr>
</tbody>
</table>

### Examples

```
CALC:FSIM:BAL:PAR:BBAL SDD12
```

**CALCulate<cnum>:FSIMulator:balun: parameter2:balanced:define sc11**

**CALCulate<cnum>:FSIMulator:BALun:PARameter<n>:BSSended[:DEFine] <char>**

- **Applicable Models:** All

**Query Syntax**

CALCulate<cnum>:FSIMulator:balun: parameter2:balanced:define scc11

**Return Type**

String

**Default**

Sdd11

**CALCulate<cnum>:FSIMulator:BALun:PARameter<n>:BBALanced[:DEFine] <char>**

**Query Syntax**

CALCulate<cnum>:FSIMulator:balun: parameter2:balanced:define cmrr

**Return Type**

Character

**Default**

Sdd11

**CALCulate<cnum>:FSIMulator:BALun:PARameter<n>:BSSended[:DEFine] <char>**

**Query Syntax**

CALCulate<cnum>:FSIMulator:balun: parameter2:balanced:define cmrr

**Return Type**

Character

**Default**

Sdd11
(Read-Write) For a Balanced - Single-ended - Single-ended type, selects the measurement parameter for the specified trace. Set device type using CALC:FSIM:BAL:DEV

See Critical Note

Parameters

<cnum>
Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<n>
Trace number on the specified channel <cnum>

<char>
Balanced - Single-ended - Single-ended Measurement parameter. Choose from:

<table>
<thead>
<tr>
<th>Sdd11</th>
<th>Sdc11</th>
<th>Sds12</th>
<th>Scs12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scd11</td>
<td>Scc11</td>
<td>Sds13</td>
<td>Scs13</td>
</tr>
<tr>
<td>Ssd21</td>
<td>Ssd31</td>
<td>Sss22</td>
<td>Sss23</td>
</tr>
<tr>
<td>Ssc21</td>
<td>Ssc31</td>
<td>Sss32</td>
<td>Sss33</td>
</tr>
<tr>
<td>Imb1</td>
<td>Imb2</td>
<td>CMRR1</td>
<td>CMRR2</td>
</tr>
</tbody>
</table>

CMRR1 = (Sds12/Scs12)
CMRR2 = (Sds13/Scs13)

Examples
CALC:FSIM:BAL:PAR:BSS CMRR2
CULCulate1:fsimulator:balun: parameter2:bssended:define SSC31

Query Syntax
CALCulate<cnum>:FSIMulator:BALun:PARameter<n>:BSSended[:DEFine]?

Return Type
Character

Default
Sdd11

CALCulate<cnum>:FSIMulator:BALun:PARameter:CATalog?

Applicable Models: All

(Read-only) This query returns the list of measurement parameters available for the currently selected device type and topology.

Balanced parameters are defined by specifying a topology: selecting which ports are balanced ports and which ports are single ended. Once a topology has been specified, that topology will yield a set of available measurement parameters. Set the topology using CALC:FSIM:BAL:DEVice.
**CALCulate<n>:FSIMulator:BALun:PARameter:CUSTom[:DEFine] <string>**

Applicable Models: Multi-port systems with > 4 ports

*(Read-Write)* Defines a balanced measurement parameter corresponding to a custom topology for systems where the port count is expandable beyond 4 ports. The device type parameter must be set to CUST using the `CALC:FSIM:BAL:DEV` command to use this command.

**See Also:**
- `CALC:FSIM:BAL:PAR:CAT?` - returns the list of parameters available for the currently selected device type and topology.
- `CALC:DTOPology` - maps device type logical ports to VNA physical ports.

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number.
- `<string>`: Balanced measurement parameter name. The parameter selections depend on the currently selected topology.

**Examples**

```
CALC:PAR:COUNT 1
CALC:FSIM:BAL:DEV CUST
CALC:FSIM:BAL:PAR:STATE ON
CALC:DTOPology "SSBSS",1,2,4,5,3,6
CALC:FSIM:BAL:PAR:CUST:DEF "SDD22"
```

**Query Syntax**

`CALCulate<n>:FSIMulator:BALun:PARameter:CUSTom[:DEFine]?`

**Return Type**

String

**Default**

Not Applicable

---

**CALCulate<n>:FSIMulator:BALun:PARameter<n>:SBALanced[:DEFine] <char>**
Applicable Models: All
(Read-Write) For a Single-ended - Balanced device type, selects the measurement parameter for the specified trace. Set device type using CALC:FSIM:BAL:DEV
See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<n> Trace number on the specified channel <cnum>

<char> Single-ended - Balanced Measurement parameter. Choose from:

<table>
<thead>
<tr>
<th>Sss11</th>
<th>Ssd12</th>
<th>Ssc12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sds21</td>
<td>Sdd22</td>
<td>Sdc22</td>
</tr>
<tr>
<td>Scs21</td>
<td>Scd22</td>
<td>Scc22</td>
</tr>
<tr>
<td>Imb</td>
<td>CMRR1</td>
<td>CMRR2</td>
</tr>
<tr>
<td></td>
<td>(Sds21/Scs21)</td>
<td>(Ssd12/Ssc12)</td>
</tr>
</tbody>
</table>

Examples

CALC:FSIM:BAL:PAR:SBAL SSD12
calculate1:fsimulator:balun: parameter2:sbalanced:define imb

Query Syntax

CALCulate<cnum>:FSIMulator:BALun:PARameter<n>:SBBalanced[:DEFine]?

Return Type Character

Default Sss11

---

CALCulate<cnum>:FSIMulator:BALun:PARameter<n>:SSBalanced[:DEFine] <char>

Applicable Models: All
(Read-Write) For a Single-ended - Single-ended - Balanced device type, selects the measurement parameter for the specified trace. Set device type using CALC:FSIM:BAL:DEV
See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
Trace number on the specified channel <cnum>

Single-ended - Single-ended - Balanced Measurement parameter. Choose from:

<table>
<thead>
<tr>
<th>Sss11</th>
<th>Sss12</th>
<th>Ssd13</th>
<th>Ssc13</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sss21</td>
<td>Sss22</td>
<td>Ssd23</td>
<td>Ssc23</td>
</tr>
<tr>
<td>Sds31</td>
<td>Sds32</td>
<td>Sdd33</td>
<td>Sdc33</td>
</tr>
<tr>
<td>Scs31</td>
<td>Scs32</td>
<td>Scd33</td>
<td>Scc33</td>
</tr>
<tr>
<td>Imb1</td>
<td>Imb2</td>
<td>CMRR1</td>
<td>CMRR2</td>
</tr>
</tbody>
</table>

Examples

```
CALC:FSIM:BAL:PAR:SSB SSD23
```

 calculus1:fsimulator:balun: parameter2:ssbalanced:define imb1

Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:PARameter<n>:SSBalanced[:DEFine]?
```

Return Type

Character

Default

Sss11

---

**CALCulate<cnum>:FSIMulator:BALun:PARameter:STATe <bool>**

Applicable Models: All

(Read-Write) Turns balanced transform ON and OFF.

See Critical Note

Parameters

- **<cnum>**
  - Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

- **<bool>**
  - State of balanced transform. Choose from
    - OFF (or 0) Balanced Transform OFF
    - ON (or 1) Balanced Transform ON

Examples

```
CALC:FSIM:BAL:PAR:STAT 1
```

 calculus1:fsimulator:balun:parameter:state off

Query Syntax

```
CALCulate<cnum>:FSIMulator:BALun:PARameter:STATe?
```
Return Type  Boolean

**Default**  OFF

### CALCulate\(<cnum>\):FSIMulator:BALun:PHASe:SWEep:PORT \(<\text{intValue}>\)

**Applicable Models:** All

(Read-Write) For a Balanced device type, specifies which balanced port the phase will be swept if `CALC:FSIM:BAL:PHAS:SWE:STAT` is set to ON.

See Critical Note

**Parameters**

\(<\text{cnum}>\)  
Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \(<\text{cnum}>\) is set to 1.

\(<\text{intValue}>\)  
Balanced port. Enter 1 or 2. Up to 2 True Mode balanced ports are possible.

**Examples**

```
CALC:FSIM:BAL:PHAS:SWE:PORT 2
```

```
calculate1:fsimulator:balun:phase:sweep:port 2
```

**Query Syntax**

```
CALCulate\(<\text{cnum}>\):FSIMulator:BALun:PHASe:SWEep:PORT?
```

**Return Type**  Numeric

**Default**  Not Applicable

### CALCulate\(<cnum>\):FSIMulator:BALun:PHASe:SWEep:STATe \(\text{<bool}>\)

**Applicable Models:** N522xB, N523xB, N524xB

(Read-Write) Sets and reads the state of phase sweep.

Learn more about iTMSA Phase Sweep.

See Critical Note

**Parameters**

\(<\text{cnum}>\)  
Channel number of the measurement. There must be a selected true mode balanced measurement on that channel. If unspecified, \(<\text{cnum}>\) is set to 1.

\(\text{<bool}>\)  
State of phase sweep:

- OFF (or 0)  Phase Sweep disabled.
- ON (or 1)  Phase Sweep enabled.
CALCulate<cnum>:FSIMulator:BALun:PHASe:SWEep:STATe?

Return Type: Boolean

Default: Off

CALCulate<cnum>:FSIMulator:BALun:STIMulus:MODE <value>

Applicable Models: M98xxA, P50xxA, E5080B

(Read-Write) Sets the stimulus mode of the VNA source. True Mode settings requires Opt S9x460A/B.

Learn more about iTMSA.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

/value> Stimulus mode. When a True-Mode is selected, the Balanced port powers are automatically uncoupled. Choose from:

- SE - Single-Ended stimulus
- TM - True-Mode stimulus
- FTM - Forward only True-Mode stimulus
- RTM - Reverse only True-Mode stimulus
- STM - Source only True-Mode stimulus

Examples

CALC:FSIM:BAL:PHAS:SWE:STAT 0
calculate2:fsimulator:balun:phase:sweep:state on

CALC:FSIM:BAL:STIM:MODE SE
calculate2:fsimulator:balun:stimulus:mode rtm

See example program

Query Syntax

CALCulate<cnum>:FSIMulator:BALun:STIMulus:MODE?

Return Type: Character

Default: SE
CALCulate<cnum>:FSIMulator:BA Lun:TOPology:BALSended[:PPORts] <bPos>,<bNeg>,<se>

Applicable Models: All
(Read-Write) For a Balanced - Single-ended device type, maps the VNA ports to the DUT ports.
Set the Balanced - Single-ended device type using CALC:FSIM:BA L:DEV
See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
<bPos> VNA port number that connects to each of the following DUT ports:
<bNeg>
<se>

Examples
CALC:FSIM:BA L:TOP:BA LS 1,2,3
calculate1:fsimulator:balun:topology:balsended:pports 4,3,2

Query Syntax
CALCulate<cnum>:FSIMulator:BA Lun:TOPology:BALSended[:PPORts]?

Return Type Numeric - Returns three numbers separated by commas.

Default Not Applicable

---

CALCulate<cnum>:FSIMulator:BA Lun:TOPology:BA Lanced[:PPORts] <p1Pos>,<p1Neg>

Applicable Models: All
(Read-Write) For a Balanced device type, maps the VNA ports to the DUT ports.
Set the Balanced - Balanced device type using CALC:FSIM:BA L:DEV
See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
<p1Pos> VNA port number that connects to each of the following DUT ports:
<p1Neg>
Examples

CALC:FSIM:BAL:TOP:BAL 1,2

Query Syntax

CALCulate<cnum>:FSIMulator:BALun:TOPology:BBALanced[:PPORts]

Return Type

Numeric - Returns two numbers separated by commas.

Default

Not Applicable

**CALCulate<cnum>:FSIMulator:Balun:Topo:BBALanced[:PPORts]**

### Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<p1Pos>` VNA port number that connects to each of the following DUT ports:
- `<p1Neg>`
- `<p2Pos>`
- `<p2Neg>`

Applicable Models: All

(Read-Write) For a Balanced - Balanced device type, maps the VNA ports to the DUT ports.

Set the Balanced - Balanced device type using **CALC:FSIM:DEV**

See Critical Note

Examples

CALC:FSIM:BAL:TOP:BBAL 1,2,3,4

Query Syntax

CALCulate<cnum>:FSIMulator:BALun:TOPology:BBALanced[:PPORts]

Return Type

Numeric - Returns four numbers separated by commas.
CALCulate\(<cnum>\)::FSIMulator:BALun:TOPology:BSSended[\:<PPORts>\]<bPos>,<bNeg>,<se1>,<se2>

Applicable Models: All
(Read-Write) Balanced-Single-ended - Single-ended device type, maps the VNA ports to the DUT ports.
Set the Balanced - Balanced device type using CALC:FSIM:BAL:DEV
See Critical Note

Parameters

\(<cnum>\) Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \(<cnum>\) is set to 1.

\(<bPos>\) VNA port number that connects to each of the DUT ports:

\(<bNeg>\)

\(<se1>\)

\(<se2>\)

Examples

\(\text{CALC:FSIM:TOP:BSS 1,2,3,4}\)

\(\text{calculate1:fsimulator:balun: topology:bssended:pports 4,3,2,1}\)

Query Syntax

CALCulate\(<cnum>\)::FSIMulator:BALun:TOPology:BSSended[\:<PPORts>\]?

Return Type

Numeric - Returns four numbers separated by commas.

Default

Not Applicable

CALCulate\(<cnum>\)::FSIMulator:BALun:TOPology:SBALanced[\:<PPORts>\]<se>,<bPos>,<bNeg>

Applicable Models: All
(Read-Write) For a Single-ended - Balanced device type, maps the VNA ports to the DUT ports.
Set the Single-ended - Balanced device type using CALC:FSIM:BAL:DEV
See Critical Note

Parameters
<cnum>  
Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<se>  
VNA port number that connects to each of the following DUT ports:

<table>
<thead>
<tr>
<th>Single-end Prc 1</th>
<th>Bal Port 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;se&gt; DUT</td>
<td>&lt;bPos&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;bNeg&gt;</td>
</tr>
</tbody>
</table>

Examples
CALC:FSIM:BAL:TOP:SBAL 1,2,3
calculate1:fsimulator:balun:topology:ssbalanced:pports 4,3,2

Query Syntax
CALCulate<cnum>:FSIMulator:BALun:TOPology:SSBalanced[:PPORts]?

ReturnType  
Numeric - Returns three numbers separated by commas.

Default  
Not Applicable

---

CALCulate<cnum>:FSIMulator:Balun:TOPOlogy:SSBalanced[:PPORts] <se1>,<se2>,<bPos>,<bNeg>

Applicable Models: All

(Read-Write) For a Single-ended - Single-ended - Balanced device type, maps the VNA ports to the DUT ports.


See Critical

Parameters

<cnum>  
Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<se1>  
VNA port number that connects to each of the following DUT ports:

<table>
<thead>
<tr>
<th>Single-end Prc 1</th>
<th>Bal Port 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;se1&gt; DUT</td>
<td>&lt;bPos&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;bNeg&gt;</td>
</tr>
</tbody>
</table>

Examples
CALC:FSIM:BAL:TOP:SSB 1,2,3,4
calculate1:fsimulator:balun:topology:ssbalanced:pports 4,3,2,1

8030
<table>
<thead>
<tr>
<th><strong>Query Syntax</strong></th>
<th>CALCulate&lt;cnm&gt;:FSIMulator:BALun:TOPology:SSBalanced[:PPORts]?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Return Type</strong></td>
<td>Numeric - Returns four numbers separated by commas.</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
FSimulatorEmbed

Specifies settings for embedding and de-embedding balanced (4-port) fixturing circuits.

```
CALCulate:FSIMulator:EMBed:
  NETWork:
    | FILename
    | PMAP
    | TYPE
  STATe
  TOPology:
    | A:PORTs
    | B:PORTs
    | C:PORTs
    | D:PORTs
  TYPE
```

Click a blue keyword to view the command details.

See Also

- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

**CALCulate<cnum>:FSIMulator:EMBed:NETWork<n>:FILename <string>** Superseded

Applicable Models: All


(Read-Write) Specifies the 4-port touchstone file (*.s4p) in which the network to embed or de-embed resides. Following this command, send CALC:FSIM:EMB:NETW:TYPE. If the specified file does not exist, an error occurs when type command is sent. Learn about 4-port network embedding.
**CALCulate<cnum>:FSIMulator:EMBed:NETWork<n>:PMAP <inA>,<inB>,<outA>,<outB>** Superseded

Applicable Models: All


*(Read-Write)* Set and return the port mapping for a 4-port SNP file to be embedded.
Learn about 4-port network embedding.

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<n>`: Network position. Choose from 1 or 2.
- `<inA> <inB> <outA> <outB>`: Port Mapping. Use four port numbers in any order.

**Example**

```
CALC:FSIM:EMB:NETW1:PMAP 1,3,2,4
```

**Query Syntax**

`CALCulate<cnum>:FSIMulator:EMBed:NETWork<n>:PMAP?`

**Return Type**

Comma-separated numeric

**Default**

1,2,3,4

---

**CALCulate<cnum>:FSIMulator:EMBed:NETWork<n>:TYPE <char>** Superseded

Applicable Models: All


*(Read-Write)* Specify the type of processing to take place on the specified 4-port network. First specify the network filename with `CALC:FSIM:EMB:NETW:FIL`. Learn about 4-port network embedding.

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<n>`: Network position. Choose from 1 or 2.
- `<string>`: File name and extension (.s4P) of the circuit. Files are stored in the "D:" drive. To recall from a different folder, specify the full path name.

**Examples**

```
10 calculate2:fsimulator:embed:network2:filename "D:\myFile.s4P"
20 calculate2:fsimulator:embed:network2:type embed
```
Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<n> Network position. Choose from 1 or 2.

<char> Processing type. Choose from:
- NONE - The same as disabling.
- EMBed - Add Network circuit.
- DEEMbed - Remove Network circuit.

Example

<table>
<thead>
<tr>
<th>Number</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>CALC:FSIM:EMB:NETW2:FIL 'myFile.s4p'</td>
</tr>
<tr>
<td>20</td>
<td>CALC:FSIM:EMB:NETW2:TYPE EMBed</td>
</tr>
</tbody>
</table>

Query Syntax

CALCulate<cnum>:FSIMulator:EMBed:NETWork<n>:TYPE?

Return Type

Character

Default

NONE

CALCulate<cnum>:FSIMulator:EMBed:STATe <bool> Superseded

Applicable Models: All

This command is changed to CALCulate:FSIMulator:DRAFt:SECTION:CIRCuit:ENABLE <ON | OFF> and CALCulate:FSIMulator:SECTION:CIRCuit:ENABLE? Learn about Using Fixture Simulator.

(Read-Write) Turns ON or OFF 4-port Network Embedding/De-embedding for all ports on the specified channel.

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<bool> Choose from:
- ON or 1 - Turns 4-port Network Embedding/De-embedding ON
- OFF or 0 - Turns 4-port Network Embedding/De-embedding OFF

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:EMB:STAT 1</td>
</tr>
<tr>
<td>calculate2:fsimulator:embed:state 0</td>
</tr>
</tbody>
</table>

Query Syntax

CALCulate<cnum>:FSIMulator:EMBed:STATe?

Return Type

Boolean

Default

OFF

CALCulate<cnum>:FSIMulator:EMBed:TOPology:A:PORTs <p1>,<p2> Superseded

Applicable Models: All

```
(Read-Write) Specifies the VNA port connections when topology A is selected. Specify topology using CALC:FSIM:EMBed:TYPE.
```

![Topology A Diagram]

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<p1>`: VNA Port number assigned to a in above graphic.
- `<p2>`: VNA Port number assigned to b in above graphic.

**Examples**

```
CALC:FSIM:EMB:TOP:A:PORT 2,1
calculate2:fsimulator:embed:topology:a:ports 1,2
```

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:EMBed:TOPology:A:PORTs?
```

**Return Type**

Numeric

**Default**

1,2

---

**Superseded**

```
CALCulate<cnum>:FSIMulator:EMBed:TOPology:B:PORTs <p1>,<p2>,<p3>
```

**Applicable Models:** All


```
(Read-Write) Specifies the VNA port connections when topology B is selected. Specify topology using CALC:FSIM:EMBed:TYPE.
```

![Topology B Diagram]
<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<p1> VNA Port number assigned to a in above graphic.

<p2> VNA Port number assigned to b in above graphic.

<p3> VNA Port number assigned to c in above graphic.

Examples

| CALC:FSIM:EMB:TOP:B:PORT 2,1,4 |
| calculate2:fsimulator:embed:topology:b:ports 1,2,3 |

Query Syntax
CALCulate<cnum>:FSIMulator:EMBed:TOPology:B:PORTs?

Return Type
Numeric

Default
1,2,3

---

CALCulate<cnum>:FSIMulator:EMBed:TOPology:C:PORTs <p1>,<p2>,<p3>,<p4> Superseded

Applicable Models: All


(Read-Write) Specifies the VNA port connections when topology C is selected. Specify topology using CALC:FSIM:EMBed:TYPE.

Topology C

Parameters

| <cnum> |
| Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1. |

| <p1> |
| VNA Port number assigned to a in above graphic. |

| <p2> |
| VNA Port number assigned to b in above graphic. |

| <p3> |
| VNA Port number assigned to c in above graphic. |

| <p4> |
| VNA Port number assigned to d in above graphic. |

Examples

| CALC:FSIM:EMB:TOP:C:PORT 2,1,4,3 |
| calculate2:fsimulator:embed:topology:c:ports 1,2,3,4 |

Query Syntax
CALCulate<cnum>:FSIMulator:EMBed:TOPology:C:PORTs?
### CALCulate<cnun>:FSIMulator:EMBed:TOPology:D:PORTs <p1>,<p2>,<p3>,<p4>...<pN> Superseded

**Applicable Models:** All


*(Read-Write)* Specifies the VNA port connections for VNAs having greater than 4 ports. Specify topology using `CALC:FSIM:EMBed:TYPE`. Learn more.

**Parameters**

- `<cnum>`: Channel number of the measurement.
- `<p1>,<p2>,<p3>,<p4>...<pN>`: VNA port number assignment.

**Examples**

- `CALC:FSIM:EMB:TOP:D:PORT 2,1,4,3,6,5,8,7`
- `calculate2:fsimulator:embed:topology:d:ports 1,2,3,4,5,6,7,8`

**Query Syntax**

`CALCulate<cnun>:FSIMulator:EMBed:TOPology:D:PORTs?`

**Return Type**

- Numeric

**Default**

- `1,2,3,4`

---

### CALCulate<cnun>:FSIMulator:EMBed:TYPE <char> Superseded

**Applicable Models:** All


*(Read-Write)* Specifies the VNA / DUT topology. Learn more about these and other VNA/DUT configurations.

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>`: VNA / DUT topology. Choose from:
  - A - 2 PNA/DUT Ports
  - B - 3 PNA/DUT Ports
  - C - 4 PNA/DUT Ports
  - D - >4 VNA/DUT Ports

Click links to see topologies and port assignment commands.
<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:EMB:TYPE A</td>
</tr>
<tr>
<td>calculate2:fsimulator:embed:type c</td>
</tr>
<tr>
<td>Query Syntax</td>
</tr>
<tr>
<td>CALCulate&lt;cnum&gt;:FSIMulator:EMBed:TYPE?</td>
</tr>
<tr>
<td>Return Type</td>
</tr>
<tr>
<td>Character</td>
</tr>
<tr>
<td>Default</td>
</tr>
<tr>
<td>A</td>
</tr>
</tbody>
</table>
FSimulatorGLoop
Specifies settings and fixturing for ground loop de-embedding / embedding.

CALCulate:FSIMulator:GLOop:
DEEMBed:
| PARameters:
| C
| L
| R
| STATe
| TYPE
| USER:FILename
EMBed:
| PARameters:
| C
| L
| R
| STATe
| TYPE
| USER:FILename
PORTs:

See Also

SCPI Command Tree

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

CALCulate<cnum>:FSIMulator:GLOop:DEEMbed:PARameters:C <value>
**Applicable Models:** All

(Read-Write) Sets and returns the Capacitance, 'C' value for the ground loop de-embedding of the specified channel.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
<tr>
<td>&lt;value&gt;</td>
<td>Capacitance value in farads. Choose a value between 1E-18 to 1E18</td>
</tr>
</tbody>
</table>

**Examples**

```
calc:fsim:glodeem:par:c 0.00002
```

```
calculate2:fsimulator:gloop:deembed:parameters:c 0.00003
```

**Query Syntax**

```
cALCulate<cnum>:FSIMulator:GLOop:DEEMbed:PARameters:C?
```

**Return Type**

Numeric

**Default**

0

**CALCulate<cnum>:FSIMulator:GLOop:DEEMbed:PARameters:L <value>**

**Applicable Models:** All

(Read-Write) Sets and returns the Inductance, 'L' value for the ground loop de-embedding of the specified channel.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
<tr>
<td>&lt;value&gt;</td>
<td>Inductance value in henries. Choose a value between -1E18 to 1E18</td>
</tr>
</tbody>
</table>

**Examples**

```
cALC:FSIM:GLO:DEEM:PAR:L 0.00002
```

```
calculate2:fsimulator:gloop:deembed:parameters:l 0.00003
```

**Query Syntax**

```
cALCulate<cnum>:FSIMulator:GLOop:DEEMbed:PARameters:L?
```

**Return Type**

Numeric
### CALCulate<cnum>:FSIMulator:GLOop:DEEMbed:PARameters:R <value>

**Applicable Models:** All

**(Read-Write)** Sets and returns the Resistance, 'R' value for the ground loop de-embedding of the specified channel.

**Note:** This command affects ALL measurements on the specified channel.

#### Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<value>`: Resistance value in ohms. Choose a value between -1E18 to 1E18

#### Examples

```
CALC:FSIMGLO:DEEM:PAR R 0.00002
```

```
calculate2:fsimulator:gloop:deembed:parameters:r 0.00003
```

#### Query Syntax

```
CALCulate<cnum>:FSIMulator:GLOop:DEEMbed:PARameters:R?
```

#### Return Type

Numeric

**Default**

0

### CALCulate<cnum>:FSIMulator:GLOop:DEEMbed:STATE <bool>

**Applicable Models:** All

**(Read-Write)** Turns ON or OFF De-embedding for the specified channel.

**Note:** This command affects ALL measurements on the specified channel.

#### Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<bool>`: Choose from:
  - `ON` or `1` - Turns De-embedding ON
OFF or 0 - Turns De-embedding OFF

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:GLO:DEEM:STAT 1</td>
<td>Boolean</td>
</tr>
<tr>
<td>calculate2:fsimulator:gloop:deembed:state 0</td>
<td></td>
</tr>
</tbody>
</table>

Query Syntax

CALCulate<cnum>:FSIMulator:GLOop:DEEMbed:STATe?

Return Type

Boolean

Default

OFF

**CALCulate<cnum>:FSIMulator:GLOop:DEEMbed:TYPE <char>**

Applicable Models: All

*(Read-Write)* Specifies the circuit model type for ground loop de-embedding.

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>` Choose from:
  - RL - Selects Shunt L circuit model.
  - RC - Selects Shunt C circuit mode.

Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:GLO:DEEM:TYPE RL</td>
<td>Character</td>
</tr>
<tr>
<td>calculate2:fsimulator:gloop:deembed:type RC</td>
<td></td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:FSIMulator:GLOop:DEEMbed:USER:FILENAME <filename>**

Applicable Models: All

*(Read-Write)* Specifies the filename of the s1p file to load for ground loop de-embedding.
Note: This command affects ALL measurements on the specified channel.

Parameters

<cnum>
Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<filename>
File name and extension (.s1P) of the de-embedding circuit.

Files are stored in the default folder "D:"

To recall from a different folder, specify the full path name.

Examples

```
CALC:FSIM:GLO:DEEM:USER:FIL 'myFile.s1P'
calculate2:fsimulator:loop:deembed:user:filename "D:\myFile.s1P"
```

Query Syntax

CALCulate<cnum>:FSIMulator:GLOop:DEEMbed:USER:FILENAME?

Return Type

String

Default

Not Applicable

CALCulate<cnum>:FSIMulator:GLOop:EMBed:PARameters:C <value>

Applicable Models: All

(Read-Write) Sets and returns the Capacitance, 'C' value for the ground loop embedding of the specified channel.

Note: This command affects ALL measurements on the specified channel.

Parameters

<cnum>
Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<value>
Capacitance value in farads. Choose a value between \(-1E18\) to \(1E18\)

Examples

```
CALC:FSIM:GLO:EMB:PAR:C 0.00002
calculate2:fsimulator:loop:embed:parameters:c 0.00003
```

Query Syntax

CALCulate<cnum>:FSIMulator:GLOop:EMBed:PARameters:C?

Return Type

Numeric

Default

0
CALCulate<cnum>:FSIMulator:GLOop:EMBed:PARameters:L <value>

Applicable Models: All

*(Read-Write)* Sets and returns the Inductance, 'L' value for the ground loop embedding of the specified channel.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<value>` Inductance value in henries. Choose a value between -1E18 to 1E18

**Examples**

```
CALC:FSIM:GLO:EMB:PAR:L 0.00002
calculate2:fsimulator:gloop:embed:parameters:l 0.00003
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:GLOop:EMBed:PARameters:L?

**Return Type**

Numeric

**Default**

0

---

CALCulate<cnum>:FSIMulator:GLOop:EMBed:PARameters:R <value>

Applicable Models: All

*(Read-Write)* Sets and returns the Resistance, 'R' value for the ground loop embedding of the specified channel.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<value>` Resistance value in ohms. Choose a value between -1E18 to 1E18

**Examples**

```
CALC:FSIM:GLO:EMB:PAR:R 0.00002
calculate2:fsimulator:gloop:embed:parameters:r 0.00003
```

**Query Syntax**

CALCulate<cnum>:FSIMulator:GLOop:EMBed:PARameters:R?

**Return Type**

Numeric

**Default**

0
**CALCulate<cnum>:FSIMulator:GLOop:EMBed:STATe <bool>**

Applicable Models: All

*(Read-Write)*  Turns ON or OFF Embedding for the specified channel.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- **<cnum>**  
  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

- **<bool>**  
  Choose from:
  
  - ON or 1 - Turns Embedding ON
  - OFF or 0 - Turns Embedding OFF

**Examples**

```
CALC:FSIM:GLO:EMB:STAT 1
calculate2:fsimulator:gloop:embed:state 0
```

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:GLOop:EMBed:STATe?
```

**Return Type**

Boolean

**Default**

OFF

**CALCulate<cnum>:FSIMulator:GLOop:EMBed:TYPE <char>**

Applicable Models: All

*(Read-Write)*  Specifies the circuit model type for ground loop embedding. Learn more.

**Parameters**

- **<cnum>**  
  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

- **<char>**  
  Choose from:
  
  - RL - Selects Shunt L circuit model.
  - RC - Selects Shunt C circuit mode.
**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:FSIM:GLO:EMB:TYPE RL</code></td>
<td>Example command</td>
</tr>
<tr>
<td><code>calculate2:fsimulator:gloop:embed:type RC</code></td>
<td>Example command</td>
</tr>
</tbody>
</table>

**Query Syntax**

`CALCulate<cnum>:FSIMulator:GLOop:EMBed:TYPE?`  

**Return Type**

Character  

**Default**

RL

---

**CALCulate<cnum>:FSIMulator:GLOop:EMBed:USER:FILename <filename>**

---

**Applicable Models:** All

**Read-Write**  
Specifies the filename of the .s1p file to load for ground loop embedding.

**Note:** This command affects ALL measurements on the specified channel.

---

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<filename>`: File name and extension (.s1P) of the embedding circuit.

Files are stored in the default folder "D:\"

To recall from a different folder, specify the full path name.

---

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:FSIM:GLO:EMB:USER:FIL 'myFile.s1P'</code></td>
<td>Example command</td>
</tr>
<tr>
<td><code>calculate2:fsimulator:gloop:embed:user:filename &quot;D:\myFile.s1P&quot;</code></td>
<td>Example command</td>
</tr>
</tbody>
</table>

**Query Syntax**

`CALCulate<cnum>:FSIMulator:GLOop:EMBed:USER:FILename?`  

**Return Type**

String  

**Default**

Not Applicable

---

**CALCulate<cnum>:FSIMulator:GLOop:PORTs <p1>,<p2>,<p3>,<p4>**

---

**Applicable Models:** All

**Read-Write**  
Specifies the VNA port connections for ground loop embedding/de-embedding. Ground loop ports are typically automatically determined based on the requested measurements, fixturing ports, and calibration ports. However, if it is desired to set them specifically, use this command.
### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Channel number of the measurement.</td>
</tr>
<tr>
<td><code>&lt;p1&gt;,&lt;p2&gt;,&lt;p3&gt;,&lt;p4&gt;</code></td>
<td>VNA port number assignment.</td>
</tr>
</tbody>
</table>

#### Examples

- `CALC:FSIM:GLO:PORT 1,2`
- `calculate2:fsimulator:gloop:ports 3,4`

#### Query Syntax

```plaintext
CALCulate<cnum>:FSIMulator:GLOop:PORTs?
```

#### Return Type

- **Numeric**
- **Default**: Not Applicable
Specifies settings for embedding and de-embedding Single-Ended (2-port) fixturing circuits.

```
CALCulate:FSIMulator:SENDed:
  DEEMbed
    PORT
      SNP:REVerse
      [TYPE]
      USER:FILename
    STATe
  OORDer
  PMCircuit
    PORT
      PARameters
        C
        G
        L
        R
        [TYPE]
        USER:FILename
      STATe
  POWer:PORT:COMPensation
    ZCONversion
      PORT
        IMAG
        REAL
        Z0[:R]
```

Applicable Models: All


(Read-Write) Set and read whether or not to reverse ports on a 2-port fixture or adapter to be de-embedded. Learn more.

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<port>`: VNA port number to which SNP file is to be de-embedded.
- `<bool>`: Choose from:
  - ON or 1 - Reverse ports
  - OFF or 0 - Do NOT Reverse ports

Examples

```
CALC:FSIM:SEND:DEEM:PORT1:SNP:REV 1
```

```
calculate2:fsimulator:sended:deembed:port2:snp:reverse 0
```

Query Syntax

```
CALCulate<cnum>:FSIMulator:SENDed:DEEMbed:PORT<n>:SNP:REVerse?
```

Return Type: Boolean

Default: OFF

CALCulate<cnum>:FSIMulator:SENDed:DEEMbed:PORT<n>[:TYPE] <char> Superseded

Applicable Models: All

(Read-Write) Select whether or not to load a 2-port De-embedding circuit model for the specified port number. Circuit model USER is valid when an associated .s2p file is specified with CALC:FSIM:SEND:DEEM:PORT1:USER:FILename.

Note: This command affects ALL measurements on the specified channel.

Parameters

- **<cnum>**: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<port>**: Port number to receive circuit model.
- **<char>**: Choose from:
  - **NONE**: Port does not have a circuit model.
  - **USER**: Circuit model for the port will be loaded from VNA drive.

Examples

- `CALC:FSIM:SEND:DEEM:PORT1 USER`
- `calculate2:fsimulator:sendded:deembedded:port2:type none`

Query Syntax

`CALCulate<cnum>:FSIMulator:SENDed:DEEMbed:PORT<n>[:TYPE]?

Return Type

- **Character**

Default

- **None**

---

**CALCulate<cnum>:FSIMulator:SENDed:DEEMbed:PORT<n>:USER:FILename <string> Superseded**

Applicable Models: All


(Read-Write) Specifies the filename of the circuit model to be used for de-embedding. Circuit model is applied when both CALC:FSIM:SEND:DEEM:PORT1 USER is selected and the filename is specified with this command.

Note: This command affects ALL measurements on the specified channel.

Parameters

- **<cnum>**: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<port>**: Port number to receive circuit model.
- **<string>**: File name and extension (.s2P) of the de-embedding circuit.
  - Files are stored in the default folder "D:\".
  - To recall from a different folder, specify the full path name.
### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>calculate2:fsimulator:deembed:port2:user:file &quot;D:\myFile.s2P&quot;</code></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### CALCulate<nun>:FSIMulator:SENDed:DEEMbed:STATe <bool> Superseded

Applicable Models: All

This command is changed to `CALCulate:FSIMulator:DRAFt:SECTion:CIRCuit:ENABle <ON | OFF>`, `CALCulate:FSIMulator:SECTion:CIRCuit:ENABle?` Learn about Using Fixture Simulator.

(Read-Write) Turns de-embedding ON or OFF for all ports on the specified channel.

Note: This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<bool>`: Choose from:
  - ON or 1 - Turns de-embedding ON
  - OFF or 0 - Turns de-embedding OFF

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:FSIM:SEND:DEEM:STAT 1</code></td>
<td>String</td>
<td>Not Applicable</td>
</tr>
<tr>
<td><code>calculate2:fsimulator:sended:deembed:state 0</code></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### CALCulate<nun>:FSIMulator:SENDed:OORDer <a,b,c,d>

Applicable Models: All

(Read-Write) Sets and returns the order in which Single-ended Fixture Operations occur. Learn more about these operations.

Note: The operation for ground loop embedding and ground loop de-embedding will always occur as the 3rd step. It cannot be moved. By default, this is after the 2-Port DeEmbedding operation.

Note: This command affects ALL measurements on the specified channel.

**Parameters**
<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<a,b,c,d> Order of operations, where:
0 - Port Extension operation
1 - 2-Port DeEmbedding operation
2 - Port Matching operation
3 - Arbitrary Impedance operation

Examples
CALC:FSIM:SEND:OORD 1,2,3,0
calculate2:fsimulator:sended:oorer 1,2,3,0

Query Syntax  CALCulate<cnum>:FSIMulator:SENDed:OORDer?

Return Type Comma-separated values

Default 0,1,2,3
CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:PORT<n>:PARameters:G <value> Superseded

Applicable Models: All

The command set of CALCulate:FSIMulator:SENDed:PMCircuit:PORT:XXXX is changed to CALCulate:FSIMulator:DRAFt:CIRCuit:YYYY and CALCulate<cnum>:FSIMulator::CIRCuit:YYYY?

Learn about Using Fixture Simulator.

(Read-Write) Sets and returns the value for the 'G' (Conductance) circuit element for a port matching circuit model to simulate on port 'n'. The port matching circuit model is selected using CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:PORT<n>[:TYPE]<char>. You can specify G, G1, or G2 based on the selected port matching circuit model. Setting a value not used by the selected circuit will have no affect. Learn more.

There are three steps to set up a port matching circuit model simulation on a specified port:

Select the port matching circuit model to simulate using CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:PORT<n>[:TYPE]<char>.

Set the values for R (Resistance), G (Conductance), C (Capacitance), and L (Inductance) corresponding to the selected port matching circuit model.

Turn the feature on using CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:STATe to simulate the circuit and compute the measurement as if the circuit were attached to the port.

Note: This command affects ALL measurements on the specified channel.

Parameters

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<n>` Port number to receive value.
- `<value>` Conductance value in siemens. Choose a value between -1E18 to 1E18

Examples

CALC:FSIM:SEND:PMC:PORT1:PAR:G 0.00002
calculate2:fsimulator:sended:pmcircuit:port2:parameters:g 0.00003

Query Syntax

CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:PORT<n>:PARameters:G?

Return Type

Numeric

Default

0

CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:PORT<n>:PARameters:L <value> Superseded

Applicable Models: All

(Read-Write) Sets and returns the value for the 'L' (Inductance) circuit element for a port matching circuit model to simulate on port 'n'. The port matching circuit model is selected using CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:PORT<n>[TYPE]<char>. You can specify L, L1, or L2 based on the selected port matching circuit model. Setting a value not used by the selected circuit will have no affect. Learn more.

There are three steps to set up a port matching circuit model simulation on a specified port:

Select the port matching circuit model to simulate using


Set the values for R (Resistance), G (Conductance), C (Capacitance), and L (Inductance) corresponding to the selected port matching circuit model.

Turn the feature on using CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:STATe to simulate the circuit and compute the measurement as if the circuit were attached to the port.

Note: This command affects ALL measurements on the specified channel.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
<tr>
<td>&lt;n&gt;</td>
<td>Port number to receive value.</td>
</tr>
<tr>
<td>&lt;value&gt;</td>
<td>Inductance value in henries. Choose a value between -1E18 to 1E18</td>
</tr>
</tbody>
</table>

Examples

CALC:FSIM:SEND:PMC:PORT1:PAR:L 0.00002
CALC:FSIM:SEND:PMC:PORT1:PAR:L 0.00003

Query Syntax

CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:PORT<n>:PARameters:L?

Return Type

Numeric

Default

0

CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:PORT<n>:PARameters:R <value> Superseded

Applicable Models: All


(Read-Write) Sets and returns the value for the 'R' (Resistance) circuit element for a port matching circuit model to simulate on port 'n'. The port matching circuit model is selected using CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:PORT<n>[TYPE]<char>. You can specify R, R1, or R2 based on the selected port matching circuit model. Setting a value not used by the selected circuit will have no affect. Learn more.

There are three steps to set up a port matching circuit model simulation on a specified port:

Select the port matching circuit model to simulate using

Set the values for R (Resistance), G (Conductance), C (Capacitance), and L (Inductance) corresponding to the selected port matching circuit model.

Turn the feature on using `CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:STATe` to simulate the circuit and compute the measurement as if the circuit were attached to the port.

**Note:** This command affects ALL measurements on the specified channel.

### Parameters

- `<cnum>`
  - Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

- `<n>`
  - Port number to receive value.

- `<value>`
  - Resistance value in ohms. Choose a value between \(-1E18\) to \(1E18\)

### Examples

- `CALC:FSIM:SEND:PMC:PORT1:PAR:R 0.00002`
- `calculate2:fsimulator:sended:pmcircuit:port2:parameters:r 0.00003`

### Query Syntax

- `CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:PORT<n>:PARameters:R?`

### Return Type

- Numeric

### Default

- 0

---

**CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:PORT<n>:TYPE <char> Superseded**

**Applicable Models:** All


**(Read-Write)** Select whether or not to load a 2 port matching circuit model for the specified port number. Circuit model USER is valid when an associated .s2p file is specified with `CALC:FSIM:SEND:PMC:PORT1:USER:FIL`.

**Note:** This command affects ALL measurements on the specified channel.

### Parameters

- `<cnum>`
  - Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

- `<n>`
  - Port number to receive circuit model.

- `<char>`
  - Circuit model. Choose from
    - NONE  No circuit model
    - SLPC  Series L - Parallel C
    - PCSL  Parallel C - Series L
    - PLSC  Parallel L - Series C
    - SCPL  Series C - Parallel L
    - PLPC  Parallel L - Parallel C
    - SCPC  Series C - Parallel C
    - PCSC  Parallel C - Series C
    - SLPL  Series L - Parallel L
    - PLPC  Parallel L - Series L
**USER**  Load S2P file

<table>
<thead>
<tr>
<th>Examples</th>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
</table>

**CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:PORT<n>:USER:FILename <string> Superseded**

Applicable Models: All


(Read-Write) Specifies the filename of the circuit model to be used for port matching. Circuit model is applied when both CALC:FSIM:SEND:PMC:PORT1 USER is selected and the filename is specified with this command.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<port>` Port number to receive circuit model.
- `<string>` File name and extension (.s2P) of the de-embedding circuit. Files are stored in the default folder "D:"

**Examples**

<table>
<thead>
<tr>
<th>Examples</th>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>calculate2:fsimulator:pmcircuit:port2:user:file &quot;D:\myFile.s2P&quot;</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:STATe <bool> Superseded**

Applicable Models: All

**Note:** This command is changed to CALCulate:FSIMulator:DRAFt:SECTION:CIRCuit:ENABle <ON | OFF> and CALCulate:FSIMulator:SECTION:CIRCuit:ENABle? Learn about Using Fixture Simulator.

(Read-Write) Turns Port Matching ON or OFF for all ports on the specified channel.
### Parameters

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <code>&lt;cnum&gt;</code> is set to 1.</td>
</tr>
<tr>
<td><code>&lt;bool&gt;</code></td>
<td>Choose from:</td>
</tr>
<tr>
<td></td>
<td>ON or 1 - Turns Port Matching ON</td>
</tr>
<tr>
<td></td>
<td>OFF or 0 - Turns Port Matching OFF</td>
</tr>
</tbody>
</table>

**Examples**

```
CALC:FSIM:SEND:PM:STAT 1
calculate2:fsimulator:send:pmcircuit:state 0
```

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:SENDed:PMCircuit:STATe?
```

**Return Type**

Boolean

**Default**

OFF

---

### CALCulate<cnum>:FSIMulator:SENDed:POWer:PORT<n>:COMPensation <bool>

**Applicable Models:** All

**Note:** This command is changed to `CALCulate:FSIMulator:POWer:PORT[p]:COMPensate[:STATe]` Learn about Using Fixture Simulator.

**(Read-Write)** Adjusts the source power at the specified port by the combined amount of loss through ALL enabled fixturing operations. Use this function to set the power level at the DUT input. Learn more.

**Note:** This command affects ALL measurements on the specified channel.

#### Parameters

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <code>&lt;cnum&gt;</code> is set to 1.</td>
</tr>
<tr>
<td><code>&lt;port&gt;</code></td>
<td>Port number to receive power compensation.</td>
</tr>
<tr>
<td><code>&lt;bool&gt;</code></td>
<td>Choose from:</td>
</tr>
<tr>
<td></td>
<td>ON or 1 - Compensate source power</td>
</tr>
<tr>
<td></td>
<td>OFF or 0 - Do NOT compensate source power</td>
</tr>
</tbody>
</table>

**Examples**

```
CALC:FSIM:SEND:POW:PORT1:COMP 1
calculate2:fsimulator:power:port2:compensation off
```

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:SENDed:POWer:PORT<n>:COMPensation?
```

**Return Type**

Boolean

**Default**

OFF
**CALCulate<cnum>:FSIMulator:SENDed:ZCONversion:PORT<n>:IMAG <value> Superseded**

Applicable Models: All


**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<n>` Port number to receive value.
- `<value>` Imaginary impedance value. Choose a value between -1E18 and 1E18

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FSIM:SEND:ZCON:PORT1:IMAG 75</td>
<td></td>
</tr>
<tr>
<td>calculate2:fsimulator:sended:zconversion:port2:imag 150</td>
<td></td>
</tr>
</tbody>
</table>

**Query Syntax**

`CALCulate<cnum>:FSIMulator:SENDed:ZCONversion:PORT<n>:IMAG?`

**Return Type**

Numeric

**Default**

0

---

**CALCulate<cnum>:FSIMulator:SENDed:ZCONversion:PORT<n>:REAL <value> Superseded**

Applicable Models: All


*(Read-Write)* Sets and returns the Real portion of the impedance value for the specified single-ended port. Use `CALC:FSIM:SEND:ZCON:PORT:IMAG` to set the imaginary value. Or use `CALC:FSIM:SEND:ZCON:PORT:Z0` to set both values together.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
<n> Port number to receive value.

<value> Real Impedance value in ohms. Choose a value between 0 to 1E7

Examples
CALC:FSIM:SEND:ZCON:PORT1:REAL 51
Calculate2:fsimulator:sended:zconversion:port2:real 75

Query Syntax
CALCulate<cnum>:FSIMulator:SENDed:ZCONversion:PORT<n>:REAL?

Return Type
Numeric

Default
50

CALCulate<cnum>:FSIMulator:SENDed:ZCONversion:PORT<n>:Z0[:R] <value> Superseded

Applicable Models: All

(Read-Write) Sets and returns the Real portion of the impedance value for the specified single-ended port. The imaginary portion is automatically set to 0.0.

Note: This command affects ALL measurements on the specified channel.

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<n> Port number to receive value.

<value> Port Impedance value in ohms. Choose a value between 0 to 1E7

Examples
CALC:FSIM:SEND:ZCON:PORT1:Z0 51
Calculate2:fsimulator:sended:zconversion:port2:z0:r 75

Query Syntax
CALCulate<cnum>:FSIMulator:SENDed:ZCONversion:PORT<n>:Z0[:R]?

Return Type
Numeric

Default
50

CALCulate<cnum>:FSIMulator:SENDed:ZCONversion:STATe <bool> Superseded

Applicable Models: All

(Read-Write) Turns Port Impedance ON or OFF for all ports on the specified channel.

Note: This command affects ALL measurements on the specified channel.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <code>&lt;cnum&gt;</code> is set to 1.</td>
</tr>
<tr>
<td><code>&lt;bool&gt;</code></td>
<td>Choose from:</td>
</tr>
<tr>
<td></td>
<td>ON or 1 - Turns Port Impedance ON</td>
</tr>
<tr>
<td></td>
<td>OFF or 0 - Turns Port Impedance OFF</td>
</tr>
</tbody>
</table>

**Examples**

```
CALC:FSIM:SEND:ZCON:STAT 1
calculate2:fsimulator:sended:zconversion:state 0
```

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:SENDed:ZCONversion:STATe?
```

**Return Type**

Boolean

**Default**

OFF
MarkerPNOP

Initiates a Power Normal Operating Point marker search and reads the results.

These commands are **Superseded** by the CALCulate:MEASure:MARKer:PNOP commands.

**CALCulate:MARKer:PNOP**

BACKoff

| GAIN?
| PIN?
| POUT?

COMPression?

| MAXimum?
| GAIN?
| MAXimum?
| PIN?
| MAXimum?

POFFset

POUT?

| MAXimum?

Click on a keyword to view the command details.

**See Also**

PNOP Example
Learn about PNOP Markers
Other SCPI Marker commands
Synchronizing the Analyzer and Controller
SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

**CALCulate<cnum>:MARKer:PNOP:BACKoff <num>**

*Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA*  
*(Read-Write)*

Turns on and sets markers 1, 2, 3, and 4 to calculate various PNOP parameters. Either this command, or the POFFset command, will initiate the PNOP search markers.

To turn off the PNOP markers, either turn them off individually or turn them All Off.
To search a User Range with the PNOP search, first activate marker 1 and set the desired User Range. Then send CALC:MARK:PNOP:BACK. The user range used with the PNOP search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

### Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Backoff value. Choose any number between -500 and 500

### Examples

```
CALC:MARK:PNOP:BACK?
calculate2:marker:pnop:backoff 10
```

### Query Syntax

CALCulate<cnum>:MARKer:PNOP:BACKoff?

### Return Type

Numeric

### Default

Not applicable

---

### CALCulate<cnum>:MARKeR:PNOP:BACKoff:GAIN?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Reads the power backoff gain value from a PNOP marker search.

PBO Gain = PBO Out - PBO In


See Critical Note

### Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

### Examples

```
CALC:MARK:PNOP:BACK:GAIN?
```

### Default

Not applicable

---

### CALCulate<cnum>:MARKeR:PNOP:BACKoff:PIN?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Reads the power backoff input value from a PNOP marker search.

PBO In = Marker 2 X-axis


See Critical Note
CALCulate\(<cnum>\):MARKer:PNOP:BACKoff:POUT?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA
(Read-only) Reads the power backoff output value from a PNOP marker search.

PBO Out = Marker 2 Y-axis


See Critical Note

Parameters

\(<cnum>\) Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \(<cnum>\) is set to 1.

Examples

\[
\text{CALC:MARK:PNOP:BACK:POUT?}
\]

Default Not applicable

CALCulate\(<cnum>\):MARKer:PNOP:COMpression?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA
(Read-only) Reads the PNOP compression value from a PNOP marker search.

Pnop Comp = Pnop Gain - Linear Gain (not shown on marker readout).


See Critical Note

Parameters

\(<cnum>\) Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \(<cnum>\) is set to 1.

Examples

\[
\text{CALC:MARK:PNOP:COMP?}
\]

Default Not applicable

CALCulate\(<cnum>\):MARKer:PNOP:COMPresision:MAXimum?
Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Reads the max compression value from a PNOP marker search.
Comp Max = Gain Max - Linear Gain (not shown on marker readout).
See Critical Note

Parameters

```
<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
```

Examples

```
CALC : MARK : PNOP : COMP : MAX?
```

Default Not applicable

---

`CALCulate<cnum>:MARKer:PNOP:GAIN?`

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Reads the PNOP gain value from a PNOP marker search.
Pnop Gain = Pnop Out - Pnop In.
See Critical Note

Parameters

```
<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
```

Examples

```
CALC : MARK : PNOP : GAIN?
```

Default Not applicable

---

`CALCulate<cnum>:MARKer:PNOP:GAIN:MAXimum?`

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Reads the max gain from a PNOP marker search.
Gain Max = PMax Out - PMax In
See Critical Note

Parameters
<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

```
CALC : MARK : PNOP : GAIN : MAX?
```

Default Not applicable

CALCulate<cnum>:MARKer:PNOP:PIN?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA
(Read-only) Reads the PNOP input value from a PNOP marker search.

Pnop In = Marker 4 X-axis value


See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

```
CALC : MARK : PNOP : PIN?
```

Default Not applicable

CALCulate<cnum>:MARKer:PNOP:PIN:MAXimum?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Reads the max input power from a PNOP marker search.

PMax In = Marker 3 X-axis value


See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

```
CALC : MARK : PNOP : PIN : MAX?
```

Default Not applicable

CALCulate<cnum>:MARKer:PNOP:POFFset <num>

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA
(Read-Write) Turns on and sets markers 1, 2, 3, and 4 to calculate various PNOP parameters.

Either this command, or the Backoff command, will initiate the PNOP search markers.

To turn off the PNOP markers, either turn them off individually or turn them All Off.

To search a User Range with the PNOP search, first activate marker 1 and set the desired User Range. Then send the CALC:MARK:PNOP:POFF command. The user range used with the PNOP search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<num> Power Offset value in dB. Choose any number between -500 and 500

Examples

CALC:MARK:PNOP:POFF 3
calculate2:marker:pnop:poffset 10

Query Syntax  CALCulate<cnum>:MARKer:PNOP:POFFset?

Return Type  Numeric

Default  ??

CALCulate<cnum>:MARKer:PNOP:POUT?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Reads the output power value of the offset marker from a PNOP marker search.

Pnop Out = Marker 4 Y-axis value


See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

CALC:MARK:PNOP:POUT?

Default  Not applicable

CALCulate<cnum>:MARKer:PNOP:POUT:MAXimum?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Reads the max output power from a PNOP marker search.
PMax Out = Marker 3 Y-axis value


See Critical Note

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
</tbody>
</table>

Examples

```
CALC:MARK:PNOP:POUT:MAX?
```

Default

Not applicable
MarkerPSAT

Initiates a Power Saturation marker search and reads the results.

These commands are Superseded by the CALCulate:MEASure:MARKer:PSAT commands.

```plaintext
CALCulate:MARKer:PSATuration
    BACKoff
    COMPression
        | MAXimum?
        | SATuration?
            GAIN?
        | LINear?
        | MAXimum?
            PIN?
        | MAXimum?
            POUT?
        | MAXimum?
```

Click on a keyword to view the command details.

See Also
- PSAT Example
- Learn about PSAT Markers
- Other SCPI Marker commands
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:MNUM or Calc:Par:Select. Learn more.

```
CALCulate<cnum>:MARKer:PSATuration:BACKoff <num>
```

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA
(Read-Write) Turns on and sets markers 1, 2, and 3 to calculate various Power Saturation parameters. The `<num>` parameter sets and reads the back-off value for a Power Saturation marker search. To turn off the Power Saturation markers, either turn them off individually or turn them All Off.

To search a User Range with the PSAT search, first activate marker 1 and set the desired User Range. Then send the CALC:MARK:PSAT:BACK command. The user range used with the PSAT search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

See Critical Note

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>`: Backoff value. Choose any number between -500 and 500

Examples

- `CALC:MARK:PSAT:BACK 3`
- `calculate2:marker:psaturation:backoff 10`

Query Syntax

`CALCulate<cnum>:MARKer:PSATuration:BACKoff?`

Return Type

Numeric

Default

0

**CALCulate<cnum>:MARKer:PSATuration:COMPression:MAXimum?**

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Reads the compression maximum value from a PSAT marker search.

Comp Max = Gain Max - Gain Linear

Use `CALC:MARK:PSAT:BACK` to initiate a PSAT search.

See Critical Note

Parameters

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

Examples

- `CALC : MARK : PSAT : COMP : MAX ?`

Default

Not applicable

**CALCulate<cnum>:MARKer:PSATuration:COMPression:SATuration?**

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Reads the compression saturation value from a PSAT marker search.

Comp Sat = Gain Sat - Gain Linear
Use `CALC:MARK:PSAT:BACK` to initiate a PSAT search.

See Critical Note

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <code>&lt;cnum&gt;</code> is set to 1.</td>
</tr>
</tbody>
</table>

### Examples

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:MARK:PSAT:COMP:SAT?</code></td>
<td>Gain Sat = Psat Out - Psat In</td>
</tr>
</tbody>
</table>

### Default

Not applicable

---

**CALCulate<cnum>:MARKer:PSATuration:GAIN?**

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-only)* Reads the saturation gain value from a PSAT marker search.

Gain Sat = Psat Out - Psat In

Use `CALC:MARK:PSAT:BACK` to initiate a PSAT search.

See Critical Note

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <code>&lt;cnum&gt;</code> is set to 1.</td>
</tr>
</tbody>
</table>

### Examples

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:MARK:PSAT:GAIN?</code></td>
<td>Gain Sat = Psat Out - Psat In</td>
</tr>
</tbody>
</table>

### Default

Not applicable

---

**CALCulate<cnum>:MARKer:PSATuration:GAIN:LINear?**

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

*(Read-only)* Reads the linear gain value from a PSAT marker search.

Gain Linear = Marker 1 - Y-axis value MINUS X-axis value.

Use `CALC:MARK:PSAT:BACK` to initiate a PSAT search.

See Critical Note

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <code>&lt;cnum&gt;</code> is set to 1.</td>
</tr>
</tbody>
</table>

### Examples

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
</table>

### Default

Not applicable
CALCulate<cnum>:MARKer:PSATuration:GAIN:MAXimum?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA
(Read-only) Reads the maximum gain value from a PSAT marker search.
Gain Max = PMax Out - PMax In
Use CALC:MARK:PSAT:BACK to initiate a PSAT search.
See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

CALC:MARK:PSAT:GAIN:MAX?

Default Not applicable

CALCulate<cnum>:MARKer:PSATuration:PIN?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA
(Read-only) Reads the power saturation input value from a PSAT marker search.
Psat In = Marker 2 X-axis value
Use CALC:MARK:PSAT:BACK to initiate a PSAT search.
See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

CALC:MARK:PSAT:PIN?

Default Not applicable

CALCulate<cnum>:MARKer:PSATuration:PIN:MAXimum?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA
(Read-only) Reads the maximum input power from a PSAT marker search.
PMax In = Marker 3 X-axis value
Use CALC:MARK:PSAT:BACK to initiate a PSAT search.
See Critical Note

Parameters
<cnum>

Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

```
CALC:MARK:PSAT:PIN:MAX?
```

Default

Not applicable

CALCulate<cnum>:MARKer:PSATuration:POUT?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Reads the back-off output power from a PSAT marker search.

PSat Out = Marker 2 Y-axis value

Use `CALC:MARK:PSAT:BACK` to initiate a PSAT search.

See Critical Note

Parameters

<cnum>

Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

```
CALC:MARK:PSAT:POUT?
```

Default

Not applicable

CALCulate<cnum>:MARKer:PSATuration:POUT:MAXimum?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Reads the back-off output power from a PSAT marker search.

PMaxOut = Marker 3 Y-axis value

Use `CALC:MARK:PSAT:BACK` to initiate a PSAT search.

See Critical Note

Parameters

<cnum>

Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

```
CALC:MARK:PSAT:POUT:MAX?
```

Default

Not applicable
MeasurePhaseNoise
CALCulate:MEASure:PN Commands

Defines the settings for phase noise measurements. Option S93031xB is required and applies ONLY to instruments with serial prefix 6021 and above.

<table>
<thead>
<tr>
<th>CALCulate:MEASure:PN</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVARiance:</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>CARRier</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>DATA</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>INTegral</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>SNOise</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
SPURious

| ANALysis
| [:STATe]
| DATA
| OMISsion
| [:STATe]
| OSSPur
| DATA
| DELete
| [:STATe]
| SENSibility
| SORT
| THReshold
| LEVel
| MINimum
| TABle
| DATA
| DELete

Click a keyword to view the command details.

See Also

- Calibrating with SCPI
- Synchronizing the Analyzer and Controller
CALCulate<ch>:MEASure<mnum>:PN:AVARiance:DEViation? <avg_time>,<fcutoff>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-only) Returns the calculated Allan deviation using the measured average time and the measured cut-off frequency for the selected measurement.

Parameters

<ch> Channel number of the measurement (optional).
<mnum> Measurement number for each measurement.
<avg_time> Measured average time (sec).
<fcutoff> Measured cut-off frequency (Hz).

Examples

```
CALC:MEAS2:PN:AVAR:DEV? 0.001,10 kHz
```

Return Type Numeric

Default Not applicable

CALCulate<ch>:MEASure<mnum>:PN:AVARIance:JITTer? <avg_time>,<fcutoff>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-only) Returns the calculated Jitter using the measured average time and the measured cut-off frequency for the selected measurement.

Parameters

<ch> Channel number of the measurement (optional).
<mnum> Measurement number for each measurement.
<avg_time> Measured average time (sec).
<fcutoff> Measured cut-off frequency (Hz).

Examples

```
CALC:MEAS2:PN:AVAR:JITT? 0.001,10 kHz
```

Return Type Numeric

Default Not applicable
**CALCulate<ch>:MEASure<mnum>:PN:AVARiance:VARiance? <avg_time>,<fcutoff>**

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

*(Read-only)* Returns the calculated Allan variance using the measured average time and the measured cut-off frequency for the selected measurement.

**Parameters**

- `<ch>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
- `<avg_time>`: Measured average time (sec).
- `<fcutoff>`: Measured cut-off frequency (Hz).

**Examples**

```
CALC:MEAS2:PN:AVAR:VAR? 0.001,10 kHz
```

**Return Type**

Numeric

**Default**

Not applicable

---

**CALCulate<ch>:MEASure<mnum>:PN:CARRier:FREQuency?**

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

*(Read-only)* Returns the measured carrier frequency.

**Parameters**

- `<ch>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.

**Examples**

```
CALC:MEAS2:PN:CARR:FREQ?
calculate2:measure2:pn:carrier:frequency?
```

**Return Type**

Numeric

**Default**

Not applicable
CALCulate<ch>:MEASure<mnum>:PN:CARRier:LEVel?

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-only) Returns the measured carrier power level.

Parameters

<ch> Channel number of the measurement (optional).

<mnum> Measurement number for each measurement.

Examples

```
CALC:MEAS2:PN:CARR:LEV?
calculate2:measure2:pn:carrier:level?
```

Return Type

Numeric

Default

Not applicable

CALCulate<ch>:MEASure<mnum>:PN:DATA:PDATa

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the dBc data array for the selected measurement.

Parameters

<ch> Channel number of the measurement (optional).

<mnum> Measurement number for each measurement.

Examples

```
CALC:MEAS2:PN:DATA:PDAT -81.285,-97.663,-100.02,-100.15,-103.37,-117.93
calculate2:measure2:pn:ndata:pdata -81.285,-97.663,-100.02,-100.15,-103.37,-117.93
```

Query Syntax

```
CALCulate<ch>:MEASure<mnum>:PN:DATA:PDATa?
```

Return Type

Numeric

Default

Not applicable
CALCulate<ch>:MEASure<mnum>:PN:DATA:PMEMory

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the dBc memory data for the selected measurement.

Parameters

<ch> Channel number of the measurement (optional).

<mnum> Measurement number for each measurement.

Examples

CALC:MEAS2:PN:DATA:PDAT -81.285, -97.663, -100.02, -100.15, -103.37, -117.93
calculate2:measure2:pn: data: pdata -81.285, -97.663, -100.02, -100.15, -103.37, -117.93

Query Syntax

CALCulate<ch>:MEASure<mnum>:PN:DATA:PDATa?

Return Type

Numeric

Default

Not applicable

CALCulate<ch>:MEASure<mnum>:PN:DATA:SPData?

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-only) Returns the spurious data (0 or 1) for the selected measurement.

Parameters

<ch> Channel number of the measurement (optional).

<mnum> Measurement number for each measurement.

Examples

CALC:MEAS2:PN:DATA:SPData?
calculate2:measure2:pn: data: spdata?

Return Type

Numeric

Default

Not applicable

CALCulate<ch>:MEASure<mnum>:PN:DATA:SPMemory?
Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-only) Returns the spurious memory data (0 or 1) for the selected measurement.

Parameters

<ch> Channel number of the measurement (optional).

<mnum> Measurement number for each measurement.

Examples

CALC:MEAS2:PN:DATA:SPMemory?
calculate2:measure2:pn:data:spmemory?

Return Type Numeric

Default Not applicable

CALCulate<ch>:MEASure<mnum>:PN:INTegral:RANGe[1-4]:DATA? <enum>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-only) Returns the specified data for the selected range number. RANGe[1-4] allows selection of the desired integration range. The following commands are used to configure these integration ranges:

CALCulate<ch>:MEASure<mnum>:PN:INTegral:RANGe[1-4]:TYPE

CALCulate<ch>:MEASure<mnum>:PN:INTegral:RANGe[1-4]:STARt

CALCulate<ch>:MEASure<mnum>:PN:INTegral:RANGe[1-4]:STOP

CALCulate<ch>:MEASure<mnum>:PN:INTegral:RANGe[1-4]:WEIGhting

Parameters

<ch> Channel number of the measurement (optional).

<mnum> Measurement number for each measurement.

<enum> (Enumeration) Choose from:

IPN - Integrated Noise
RFM - Residual FM
RAM - Residual AM
RPM - Residual PM
RMSJ - RMS Jitter
RMSR - RMS Radian
RMSD - RMS Degree

Examples
calculate2:measure2:pn:rangel:data? RFM

Return Type Numeric
Default IPN

CALCulate<ch>:MEASure<mnum>:PN:INTegral:RANGe[1-4]:STARt <value>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the start frequency of the selected integration range. RANGe[1-4] allows selection of the desired integration range. The CALCulate:MEASure:PN:INTegral:RANGe:TYPE command must be set to CUSTom.

Parameters

<ch> Channel number of the measurement (optional).
<mnum> Measurement number for each measurement.
<value> Start frequency of integration range. If the range is unspecified, the start frequency is assigned to RANGe1.

Examples
CALC:MEAS2:PN:INT:RANG1 10 MHz
calculate2:measure2:pn:integral:rangel:start 10 MHz

Query Syntax CALCulate<ch>:MEASure<mnum>:PN:INTegral:RANGe[1-4]:STARt?

Return Type Numeric
Default Minimum frequency of the analyzer

CALCulate<ch>:MEASure<mnum>:PN:INTegral:RANGe[1-4]:STOP <value>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB
(Read-Write) Sets and returns the stop frequency of the selected integration range. RANGe[1-4] allows selection of the desired integration range. The CALCulate:MEASure:PN:INTegral:RANGe:TYPE command must be set to CUSTom.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<value>` Stop frequency of integration range. If the range is unspecified, the stop frequency is assigned to RANGe1.

Examples

```
CALC:MEAS2:PN:INT:RANG1 10 GHz
calculate2:measure2:pn:integral:range1:start 10 GHz
```

Query Syntax

```
CALCulate<ch>:MEAS< Именно<mnum>>:PN:INTegral:RANGe[1-4]:STOP?
```

Return Type

Numeric

Default

Maximum frequency of the analyzer

---

CALCulate<ch>:MEAS< Именно<mnum>:PN:INTegral:RANGe[1-4]:TYPE <enum>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the integration range type of the selected integration range. RANGe[1-4] allows selection of the desired integration range.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<enum>` (Enumeration) Choose from:
  - OFF - Turn integration range off. The range will not be measured.
  - FULL - Set integration range to full span.
  - CUSTom - Set user-defined start and stop frequency range.

Examples

```
CALC:MEAS2:PN:INT:RANG1:TYPE FULL
calculate2:measure2:pn:integral:range1:type full
```
**CALCulate<ch>:MEASure<mnum>:PN:INTegral:RANGe[1-4]:WEIGhting <string>*

**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the weighting filter of the selected integration range. RANGe[1-4] allows selection of the desired weighting filter. This command selects a weighting filter whose values are applied to the calculation of residual effects. The file should be saved in the following directory: C:\Program Data\Keysight\Network Analyzer\WeightingFilter.

**Parameters**

- `<ch>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
- `<string>`: Weighting filter file name which contains the weighting data.

**Examples**

```
CALC:MEAS2:PN:INT:RANG1:WEIG "Filter_A"
calculate2:measure2:pn:range1:weighting "Filter_A"
```

**Query Syntax**

CALCulate<ch>:MEASure<mnum>:PN:INTegral:RANGe[1-4]:WEIGhting?

**Return Type**

String

**Default**

Not applicable

---

**CALCulate<ch>:MEASure<mnum>:PN:SNOise:DECades[:STATe] <bool>**

**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Enables and disables the spot noise calculation on every decade offset frequency.

**Parameters**

- `<ch>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
Choose from:

0 - OFF - Disable spot noise calculation on every decade offset frequency.
1 - ON - Enable spot noise calculation on every decade offset frequency.

**Examples**

```
CALCULATE:MEAS2:PN:SN0ise:DECades ON
calculate2:measure2:pn:snoise:decades:state ON
```

**Query Syntax**

```
CALCULATE<ch>:MEASURE<mnum>[:PN]:SN0ise:DECades[:STATe]?
```

**Return Type**

Boolean

**Default**

ON

---

**CALCulate<ch>:MEASURE<mnum>[:PN]:SN0ise:DECades:X?**

- **Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

(Read-only) Returns the spot noise x-axis array of all decade offset frequencies.

**Parameters**

- `<ch>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.

**Examples**

```
CALCULATE:MEAS2:PN:SN0ise:DECades:X?
calculate2:measure2:pn:snoise:decades:x?
```

**Return Type**

Numeric

**Default**

Not applicable

---

**CALCulate<ch>:MEASURE<mnum>[:PN]:SN0ise:DECades:Y?**

- **Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

(Read-only) Returns the spot noise y-axis array of all decade offset frequencies.

**Parameters**

- `<ch>`: Channel number of the measurement (optional).
Measurement number for each measurement.

Examples

```
CALC:MEAS2:PN:SNO:DEC:y?
calculate2:measure2:pn:snoise:decades:y?
```

Return Type

Numeric

Default

Not applicable

---

**CALCulate<ch>:MEASure<mnum>:PN:SNOise[:STATE] <bool>**

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Enables and disables spot noise calculation for the selected measurement number.

Parameters

- `<ch>` Channel number of the measurement (optional).
- `<mnum>` Measurement number for each measurement.
- `<bool>` Choose from:
  - 0 - OFF - Disable spot noise.
  - 1 - ON - Enable spot noise.

Examples

```
CALC:MEAS2:PN:SNOise ON
calculate2:measure2:pn:snoise:state ON
```

Query Syntax

`CALCulate<ch>:MEASure<mnum>:PN:SNOise[:STATE]?`

Return Type

Boolean

Default

OFF

---


Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Enables and disables spot noise calculation for the specified user-defined offset frequency.

Parameters

- `<ch>` Channel number of the measurement (optional).
<mnum> Measurement number for each measurement.

<bool> Choose from:
0 - OFF - Disable spot noise calculation for the specified user-defined offset frequency.
1 - ON - Enable spot noise calculation for the specified user-defined offset frequency.

Examples
CALC:MEAS2:PN:SNOise:USER2 ON
calculate2:measure2:pn:snoise:user2:state ON

Query Syntax
CALCulate<ch>:MEASure<mnum>:PN:SNOiseUSER[1-6]:STATe?

Return Type
Boolean

Default
ON

CALCulate<ch>:MEASure<mnum>:PN:SNOise:USER[1-6]:X

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB
(Read-Write) Sets and returns the offset frequency on which the spot noise is calculated. Up to six offset frequencies can be defined.

Parameters

<ch> Channel number of the measurement (optional).

<mnum> Measurement number for each measurement.

Examples
CALC:MEAS2:PN:SNO:USER2:X 1.234 kHz
calculate2:measure2:pn:snoise:user2:x 1.234 kHz

Query Syntax
CALCulate<ch>:MEASure<mnum>:PN:SNOiseUSER[1-6]:X?

Return Type
Numeric

Default
Not applicable

CALCulate<ch>:MEASure<mnum>:PN:SNOise:USER[1-6]:Y?

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB
(Read-only) Returns the spot noise y-axis value of the specified offset frequency. The offset frequency is defined using the 

Parameters

  <ch> Channel number of the measurement (optional).

  <mnum> Measurement number for each measurement.

Examples

calculate2:measure2:pn:snoise:user2:y?

Return Type Numeric

Default Not applicable

CALCulate<ch>:MEASure<mnum>:PN:SPURious:ANALysis[:STATe] <bool>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Enables and disables spurious analysis.

Parameters

  <ch> Channel number of the measurement (optional).

  <mnum> Measurement number for each measurement.

  <bool> Choose from:

           0 - OFF - Disable spurious analysis.
           1 - ON - Enable spurious analysis.

Examples

CALC:MEAS2:PN:SPUR:ANAL ON
calculate2:measure2:pn:spurious:analysis:state ON

Query Syntax CALCulate<ch>:MEASure<mnum>:PN:SPURious:ANALysis[:STATe]?

Return Type Boolean

Default OFF

CALCulate<ch>:MEASure<mnum>:PN:SPURious:DATA?
CALCulate<ch>:MEASure<mnum>:PN:SPURious:OMISsion[:STATe] <bool>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Enables and disables spur omission.

Parameters

<ch> Channel number of the measurement (optional).
<mnum> Measurement number for each measurement.
<bool> Choose from:

0 - OFF - Disable spurious omission.
1 - ON - Enable spurious omission.

Examples

CALC:MEAS2:PN:SPUR:OMIS ON
calculate2:measure2:pn:spurious:omission:state ON

Query Syntax

CALCulate<ch>:MEASure<mnum>:PN:SPURious:OMISsion[:STATe]?

Return Type

Boolean

Default

OFF

CALCulate<ch>:MEASure<mnum>:PN:SPURious:OSSPur:DATA <data>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-only) Returns a list of detected spurs. Data per spur includes frequency, power, and jitter.

Parameters

<ch> Channel number of the measurement (optional).
<mnum> Measurement number for each measurement.

Examples

CALC:MEAS2:PN:SPUR:DATA?
calculate2:measure2:pn:spurious:data?
Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the User Spur Table data which defines spurs to omit.

Parameters

- `<ch>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
- `<data>`: (Array) List of spurious frequencies of spurs to omit.

Examples

```
CALC:MEAS2:PN:SPUR:OSSP:DATA 1.0e3,10.0e3
```
```
calculate2:measure2:pn:spurious:osspur:data 1.0e3,10.0e3
```

Query Syntax

```
CALCulate<ch>:MEASure<mnum>:PN:SPURious:OSSPur:DATA?
```

Return Type: Array

Default: Not applicable

---

**CALCulate<ch>:MEASure<mnum>:PN:SPURious:OSSPur:DELete**

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Write-only) Deletes the User Spur Table data which defines spurs to omit.

Parameters

- `<ch>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.

Examples

```
CALC:MEAS2:PN:SPUR:OSSP:DEL
```
```
calculate2:measure2:pn:spurious:osspur:delete
```

Default: Not applicable

---

**CALCulate<ch>:MEASure<mnum>:PN:SPURious:OSSPur[:STATe] <bool>**

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Enables and disables user specified spur omission.

Parameters
### Channel number of the measurement (optional).

### Measurement number for each measurement.

### Choose from:

- **0 - OFF** - Disable user specified spurious omission.
- **1 - ON** - Enable user specified spurious omission.

### Examples

```
CALC:MEAS2:PN:SPUR:OSSP ON
```

```
calculate2:measure2:pn:spurious:osspur:state ON
```

### Query Syntax

`CALCulate<ch>:MEASure<mnum>:PN:SPURious:OSSPur[:STATe]?

### Return Type

Boolean

### Default

OFF

----

### CALCulate<ch>:MEASure<mnum>:PN:SPURious:SENSibility <num>

**Applicable Models:** N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the spurious sensibility number.

### Parameters

- `<ch>`
  - Channel number of the measurement (optional).
- `<mnum>`
  - Measurement number for each measurement.
- `<num>`
  - Spurious sensibility value. The default is 3 (3 x standard deviation).

### Examples

```
CALC:MEAS2:PN:SPUR:SENS 3
```

```
calculate2:measure2:pn:spurious:sensibility 3
```

### Query Syntax

`CALCulate<ch>:MEASure<mnum>:PN:SPURious:SENSibility?

### Return Type

Numeric

### Default

Not applicable

----

### CALCulate<ch>:MEASure<mnum>:PN:SPURious:SORT <enum>

---

8089
Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the spurious table sorting order.

Parameters

<ch> Channel number of the measurement (optional).
<mnum> Measurement number for each measurement.
<enum> (Enumeration) Choose from:
- POWER - Sort spurious table by power.
- OFFSET - Sort spurious table by offset.

Examples

```
CALC:MEAS2:PN:SPUR:SORT POW
```
```
calculate2:measure2:pn:spurious:sort offset
```

Query Syntax

```
CALCulate<ch>:MEASure<mnum>:PN:SPURious:SORT?
```

Return Type  Enumeration

Default  POWER

---

CALCulate<ch>:MEASure<mnum>:PN:SPURious:THReshold:LEVel:MINimum <num>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the minimum spurious threshold level.

Parameters

<ch> Channel number of the measurement (optional).
<mnum> Measurement number for each measurement.
<num> Minimum spurious threshold level. The range is -500 dBC to 500 dBc.

Examples

```
```
```
calculate2:measure2:pn:spurious:threshold:level:minimum -500 dBc
```

Query Syntax

```
CALCulate<ch>:MEASure<mnum>:PN:SPURious:THReshold:LEVel:MINimum?
```

Return Type  Numeric

---

8090
CALCulate<ch>:MEASure<mnum>:PN:SPURious:THReshold:TABle:DATA <data>

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Read-Write) Sets and returns the spurious threshold table data.

Parameters

- `<ch>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.
- `<data>`: (Array) Spurious threshold table data: start freq, lower threshold (dB), upper threshold (dB).

Examples

```
CALC:MEAS2:PN:SPUR:THR:TAB:DATA 1,0,10
```

Query Syntax

```
CALCulate<ch>:MEASure<mnum>:PN:SPURious:THReshold:TABle:DATA?
```

Return Type

Array

Default

Not applicable

---

CALCulate<ch>:MEASure<mnum>:PN:SPURious:THReshold:TABle:DELe

Applicable Models: N522xB, N524xB with Phase Noise Option S93031xB

(Write-only) Deletes the spurious threshold table.

Parameters

- `<ch>`: Channel number of the measurement (optional).
- `<mnum>`: Measurement number for each measurement.

Examples

```
```

Default

Not applicable
MeasureY

Controls the display of Y-axis for various measurements.

CALCulate:MEASure:Y
AXIS
UNIT?
[:VALues]

Click a keyword to view the command details.

See Also
Synchronizing the Analyzer and Controller
SCPI Command Tree

CALCulate<ch>:MEASure<mnum>:Y:AXIS:UNIT?

Applicable Models: All

(Read-only) Returns the current units of the Y-axis (HZ | SEcond | MINute | HOUR | DAY | DB | DBM | DBMV | WATTs | FARad | HENRy | OHM | MHO | SIEMen | VOLT | DEGRee | RADians | METer | DPHZ | UNIT | NONE | TNORMalized | NTEMPerature | KELVin | CENTigrade | FAHRenheit | FEET | INCH | DBMAAMPere | VOLTAge | DBUV | PERCentage | DMVRoothz | DUVRoothz | DMARoothz | WPHZ | VROothz | ARooothz | DBC | DCPerhz | DBPerhz | HZPerhz | PRHz | VPHz | DBV|DEFault).

This command can be used for all Measurement Classes.

Parameters

<ch> Channel number of the measurement (optional).
<mnum> Measurement number for each measurement.

Examples

CALC:MEAS2:Y:AXIS:UNIT?

Return Type

Enumeration

Default

FREQuency

CALCulate<ch>:MEASure<mnum>:Y[:VALues]?

Applicable Models: All
(Read-only) Returns the Y-axis values for the selected measurement in the current units.

This command can be used for all Measurement Classes.

Parameters

<ch> Channel number of the measurement (optional).

<num> Measurement number for each measurement.

Examples

CALC:MEAS2:Y?

Return Type

Array of data

Default

Not applicable
Control SCPI
Control Auxiliary Commands

Specifies the settings to remotely control the Auxiliary IO connector.

Note: The PNA-X, N522xA and N523xA models do NOT have this connector. However, the following commands are used to control ADC voltages on the Power I/O connector: CONT:AUX:OUTP:VOLT and CONT:AUX:INPut:VOLT?. Sending other Control:AUX commands may result in unusual behavior.

CONTrol:AUXiliary

C FOOTswitch? INPUT OUTPUT PASSfail SWEepend

MODE MODE VOLtagge

[DATA] LOGic MODE VOLTage? LOGic MODE SCOPe POLicy STATus

Click on a blue keyword to view the command details.

see Also

Example Programs
Synchronizing the Analyzer and Controller
SCPI Command Tree
See a pinout and detailed description of the Power I/O Connector

CONTrol:AUXiliary:C[:DATA] <num>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Reads and writes a 4-bit value to Port C on the Aux I/O connector. This port is connected internally to the Handler IO connector. Therefore this command will also affect the state of Port C on the Handler IO

Parameters

<num> Data value. Choose any number 0 to 15.

Examples

CONTrol:AUXiliary:C:DATA 15
For Positive Logic Port C lines C0, C1, C2, C3 go High or if in Negative Logic they go Low.

CONTrol:AUXiliary:C:DATA?
A returned value of 15 when in Positive Logic indicates Port C lines C0, C1, C2, C3 are High, or if in Negative Logic they are Low.

Query Syntax

CONTrol:AUXiliary:C:DATA?
**Return Type**

Numeric

**Default**

0

---

**CONTroll:AUXiliary:C:LOGic <char>**

**Applicable Models:** N522xB, N523xB, N524xB

**(Read-Write)** Reads and writes the logic mode of Port C on the AUX IO. This port is connected to Port C of the Handler IO connector. Therefore, it will have the same logic setting.

**Parameters**

<char>  
Logic of Port C. Choose from:

POSitive - when a value of one is written the associate line goes High.

NEGative - when a value of one is written the associate line goes Low.

When Port C is in Output/Write mode, a change in logic causes the output lines to change state immediately. For example, Low levels change to High levels.

When Port C is in Input/Read mode, a change in logic does NOT cause the lines to change, but data read from Port C will reflect the change in logic.

**Examples**

CONT:AUX:C:LOG POS 'Positive logic is applied to Port C data.'

**Query Syntax**

CONTroll:AUXiliary:C:LOGic?

**Return Type**

Character

**Default**

NEGative

---

**CONTroll:AUXiliary:C:MODE <char>**

**Applicable Models:** N522xB, N523xB, N524xB

**(Read-Write)** Sets Port C to read or write mode. This port is connected to Port C of the Handler IO connector. Therefore, it will have the same mode setting.

**NOTE:** When Port C is set to INPut mode, data writes are NOT applied to the lines. MODE must be set to OUTPut mode before writing.

**Parameters**

<char>  
INPut - set the port for reading
OUTPut - set the port for writing

**Examples**

```
CONT:AUX:C:MOD INP 'set Port C to Input Mode for reading.
CONT:AUXiliary:C:MODE? 'queries the input/output mode that the port set to.
```

**Query Syntax**

CONT:AUXiliary:C:MODE?

**Return Type**

Character

**Default**

INPut

---

**CONTrol:AUXiliary:FOOTswitch[:STATe]?**

Applicable Models: N522xB, N523xB, N524xB

*(Read only)* Reads the Auxiliary connector Footswitch Input.

**Examples**

```
CONT:AUX:FOOT?
control:auxiliary:footswitch:state?
```

**Return Type**

Boolean

ON (or 1) = pressed

OFF (or 0) = released

**Default**

OFF (0) - Released

---

**CONTrol:AUXiliary:FOOTswitch:MODe <char>**

Applicable Models: N522xB, N523xB, N524xB

*(Read-Write)* This command sets the mode of the "FootSwitch In" line on the Auxiliary IO. These mode settings determine what occurs when the footswitch is pressed. Examine your results carefully when using these command modes.

**Parameters**

*<char>*

IGNore - While in this mode any Footswitch presses are ignored.

SWEep - While in this mode a Footswitch press will trigger the sweep.

**NOTE:** The instrument must be in Manual Trigger Mode.
RECall - While in this mode a Footswitch press will recall an instrument state. When more than one state is available each footswitch press recalls the next state, then starts over from the beginning.

MACR0 - While in this mode a Footswitch press will load and run a macro. When more than one macro are available each successive footswitch press loads and runs the next macro, then starts over from the beginning.

**Examples**

```plaintext
CONT:AUX:FOOT:MODE MACRo
```

This sets the footswitch mode to MACRo causing a macro to be loaded and run with a footswitch press.

```plaintext
CONTrol:AUXiliary:FOOTswitch:MODE?
```

This query returns the footswitch mode setting.

**Query Syntax**

```plaintext
CONTrol:AUXiliary:FOOTswitch:MODE?
```

**Return Type**

Character

**Default**

IGNore

---

**CONTrol:AUXiliary:INPut<n>:VOLTage?**

*Applicable Models: N522xB, N523xB, N524xB*  
*(Read-Only)*

Reads voltage on the Power I/O connector.

From the Control:Aux commands, ONLY this and `CONT:AUX:OUTP:VOLT` can be used on the PNA-X.

**Parameters**

<n>  
Port number. If unspecified, value is set to 1.

Choose from

1. Reads voltage on the AUX I/O connector (pin 14) or on the Power I/O connector Analog In 1 port (pin 7).
2. Reads voltage on the Power I/O connector Analog In 2 port (pin 8).
3. Reads voltage on Power I/O connector GndSens (pin 6) PNA-X only.

**Examples**

```plaintext
CONT:AUX:INPut2:VOLT?
```

**Return Type**

Numeric

**Default**

Not Applicable

---

**CONTrol:AUXiliary:OUTPut<n>:MODE <char>**
Applicable Models: N522xB, N523xB, N524xB

(Read-Write) This command sets the mode of the selected "Analog Out" line on the Power I/O connector. The modes give the user the option to have the requested voltage applied immediately or not until the sweep is done.

Parameters

<n>  
Port number. If unspecified, value is set to 1.

<char>  
WAIT - While in this mode any voltage changes sent to the selected analog out will only get applied to the output between sweeps.

NOWait - While in this mode any voltage changes sent to the selected analog out will occur right away without waiting until the end of a sweep.

Examples

CONT:AUX:OUTP1:MOD WAIT This sets the mode so that voltages sent to "Analog Out 1" are only applied at the end of a sweep.

CONT:AUX:OUTP2:MOD? This query returns the current mode for "Analog Out 2".

Query Syntax

CONTrol:AUXiliary:OUTPut<n>:MODe?

'Reads the output mode

Return Type  
Character

Default  
WAIT

CONTrol:AUXiliary:OUTPut<out>:VOLTage <num>

Applicable Models: N522xB, N523xB, N524xB, E5080B

(Read-Write)  
Sets and reads voltage on the Power I/O connector AnalogOut1|2 (PNA), AUX Out 1|2 (E5080B).

(PNA) From the Control:Aux commands, ONLY this and CONT:AUX:INPut:VOLT? can be used.

Note: The 9-pin PWR I/O (Power I/O) D connector on the rear-panel replaces much of the functionality of the AUX I/O connector on older VNA models. The Power I/O voltages can be set using the following methods:

- CONTrol:AUXiliary:OUTPut:VOLTage or put_OutputVoltage Method (no GUI equivalent, global scoped, and settings not saved as part of the instrument state)
- SOURce:DC:START and SOURce:DC:STOP (DC Source dialog is the GUI equivalent, channel scoped, and settings saved as part of the instrument state)
- Interface Control dialog (no remote equivalent, channel scoped, and settings saved as part of the instrument state)

To avoid unexpected behavior, choose one method only to set the Power I/O voltages.
### Parameters

**<out>**

DAC output number. Choose from:

1 - PNA: Output 1 (Aux I/O pin 3) and (Power I/O pin 3), E5080B AUX Out 1
2 - PNA: Output 2 (Aux I/O pin 2) and (Power I/O pin 4), E5080B AUX Out 2

**<num>**

Output Voltage. Choose a voltage value between -10 and +10 volts

### Examples

```
CONT:AUX:OUTP1:VOLT 5
control:auxiliary:output2:voltage 5
```

### Query Syntax

```
CONTroll:AUXiliary:OUTPut<out>:VOLTage?
```

'Reads the output DAC voltage

### Return Type

Numeric

### Default

0

---

### CONTroll:AUXiliary:PASSfail:LOGic <char>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets the logic of the PassFail line (pin 12) on the Material Handler IO (pin 33).

**Parameters**

**<char>**

Choose from:

- **POSitive** - Causes the PassFail line to have positive logic (high = pass, low = fail).
- **NEGative** - Causes the PassFail line to have negative logic (high = fail, low = pass).

**Examples**

```
CONT:AUX:PASS:LOG POS
control:auxiliary:passfail:logic negative
```

**Query Syntax**

```
CONTroll:AUXiliary:PASSfail:LOGic?
```

Return Type: Character

**Default**

**POSitive**

---

### CONTroll:AUXiliary:PASSfail:MODe <char>

Applicable Models: N522xB, N523xB, N524xB
(Read-Write) Sets and reads the default logical pass/fail state. This is the state the pass/fail line stays in until a failure occurs (if the mode is NOWait), or until an end-of-sweep condition occurs (if the modes is PASS or FAIL).

The end-of-sweep condition is determined by the **CONTrol:AUXiliary:PASSfail:SCOPe** command.

### Parameters

<table>
<thead>
<tr>
<th>&lt;char&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PASS</strong> - the line stays in PASS state until the end-of-sweep condition occurs, at which time the pass/fail line is set, and the pass/fail strobe (line 36) is activated.</td>
</tr>
<tr>
<td><strong>FAIL</strong> - the line stays in FAIL state until the end-of-sweep condition occurs, at which time the pass/fail line is set, and the pass/fail strobe (line 36) is activated.</td>
</tr>
<tr>
<td><strong>NOWait</strong> - the pass/fail line is set, and the pass/fail strobe (line 36) is activated as soon as a failure condition occurs.</td>
</tr>
</tbody>
</table>

### Examples

```
CONT:AUX:PASS:MODE NOW
control:auxiliary:passfail:mode fail
```

### Query Syntax

**CONTrol:AUXiliary:PASSfail:MODE?**

**Return Type**

Character

**Default**

**NOWait**

---

**CONTrol:AUXiliary:PASSfail:SCOPe** <char>

**Applicable Models:** N522xB, N523xB, N524xB

(Read-Write) Sets and reads the scope of pass/fail testing. The pass/fail line can report the status of all measurements on each channel, or all measurements on all channels. This command selects which option to use.

### Parameters

<table>
<thead>
<tr>
<th>&lt;char&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CHANnel</strong> - A pass/fail result is computed and written to the output pins at the end of all sweeps on a channel.</td>
</tr>
<tr>
<td><strong>GLOBal</strong> - A pass/fail result is computed and written to the output pins at the end of all sweeps on a channel.</td>
</tr>
</tbody>
</table>

If the pass/fail mode is NOWait (as set by **CONTrol:AUXiliary:PASSfail:MODE**), the status and strobe pins are written immediately. Otherwise the pins are written as indicated above. Regardless of the mode value, the pass/fail line is returned to its default state (as set by the **MODE** command) at the end of channel or group of channels.
Examples

CONT:AUX:PASS:SCOP CHAN
control:auxiliary:passfail:scope sweep

Query Syntax
CONTrol:AUXiliary:PASSfail:SCOPE?

Return Type
Character

Default
GLOBal

CONTrol:AUXiliary:PASSfail:POLicy <char>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets the policy used to determine how global pass/fail is computed.

Parameters

<char>
Name of the policy to use. Choose from:

- ALLTests - Pass/Fail Status returns PASS if all tests on all measurements pass.

- ALLMeas - Pass/Fail Status returns PASS if all measurements have associated tests, and all tests pass. FAIL is returned if even one measurement has no associated limit test.

Only those measurements which are not in HOLD mode contribute to the pass/fail result.

Examples

CONT:AUX:PASS:POL ALLM
control:auxiliary:passfail:policy alltests

Query Syntax
CONTrol:AUXiliary:PASSfail:POLicy?

Return Type
Character

Default
ALLTests

CONTrol:AUXiliary:PASSfail:STATus?

Applicable Models: N522xB, N523xB, N524xB

(Read-Only) Returns the most recent pass/fail status value. Use this command as follows:

- Set the VNA trigger scope to ALL
- Set the VNA trigger source to MANUAL or EXTERNAL.
- Trigger the VNA.
- Use the *OPC? command to determine when the sweep is complete.
- Use the CONT:AUX:PASS:STAT? query to obtain the global pass/fail result.
Return Type

Character - One of the following is returned:
PASS - all measurements not in HOLD mode have been swept, and all associated limit tests have passed.
FAIL - all measurements not in HOLD mode have been swept, and one or more limit tests failed according to the specified Pass/Fail policy.
NONE - status cannot be determined because measurements are in progress.

Default
Not Applicable

CONTrol:AUXiliary:SWEepend <char>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Specifies the event that will cause the Sweep End line pin 11) to go to a low (false) state of the Material Handler IO. The line will return to a high state after the appropriate calculations are complete.

Parameters

<char>
Choose from:
SWEep - the line goes low when each sweep is complete.
CHANnel - The line goes low when all of the sweeps for each channel is complete.
GLOBal - The line goes low when all the sweeps for all channels are complete.

Examples
CONT:AUX:SWE SWE
control:auxiliary:sweepend channel

Query Syntax
CONTrol:AUXiliary:SWEepend?

Return Type
Character

Default
SWEep
Control SCPI
Control External Test Set Commands

Specifies the settings to remotely control the External Test Set IO connector.

```
CONTrol:EXTernal
   TESTset
      DATA    INTerrupt?    RAWData     SWEpHoldoff?
```

Click on a blue keyword to view the command details.

See Also
- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree
- See a pinout and detailed description of the External Test Set IO connector

CONTrol:EXTernal:TESTset:DATa <addr>,<data>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Reads and writes 13 bits of data to the specified address using the AD0 through AD12 lines of the external test set connector. The instrument generates the appropriate timing signals (strokes the address, then the data) to control an external test set.

Parameters

- `<addr>`: Decimal equivalent of the 13 bit binary address.
- `<data>`: Decimal equivalent of the 13 bit binary data

Examples

```
CONT:EXT:TEST:DATA 12,3
CONTrol:external:testset:DATA 12,3
```

Query Syntax

```
CONTrol:EXTernal:TESTset:DATA? <addr>
```

'Reads the decimal equivalent of the binary data from the specified address

Return Type

Numeric

Default: Not Applicable
CONTrol:EXTernal:TESTset:INTerrupt?

Applicable Models: N522xB, N523xB, N524xB

(Read-Only) Reads the boolean state of the Interrupt In line (pin 13) on the external test set connector.

Examples

```
CONT:EXT:TEST:INT?
control:external:testset:interrupt?
```

Return Type

Boolean

False (0) - the line is being held at a TTL High.

True (1) - the line is being held at a TTL Low.

Default

Not Applicable

CONTrol:EXTernal:TESTset:RAWData <data>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Reads and writes 16 bits of data through the AD0 through AD12 and three timing lines of the external test set connector. Does NOT generate appropriate timing signals.

Note: During a WRITE, Bit 13 must always be low. Otherwise Bit 0-12 will tristate

Parameters

```
<data>
Decimal equivalent of the binary data.
```

Format of data WRITTEN with RAWData:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Bit</th>
<th>Signal name</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>0</td>
<td>AD0*</td>
</tr>
<tr>
<td>23</td>
<td>1</td>
<td>AD1*</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>AD2*</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>AD3*</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>AD4*</td>
</tr>
<tr>
<td>21</td>
<td>5</td>
<td>AD5*</td>
</tr>
<tr>
<td>20</td>
<td>6</td>
<td>AD6*</td>
</tr>
<tr>
<td>19</td>
<td>7</td>
<td>AD7*</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>AD8*</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>AD9*</td>
</tr>
</tbody>
</table>
* This Output will float if RLW (bit-13) is set high

### Examples

```
CONT:EXT:TEST:RAW 8001
CONT:EXT:TESTset:rawdata 1234
```

### Query Syntax

```
CONT:EXT:TESTset:RAWData?
```

### Return Format

Format of data **READ** with **RAWData**?

<table>
<thead>
<tr>
<th>Pin</th>
<th>Bit</th>
<th>Signal name</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>0</td>
<td>AD0*</td>
</tr>
<tr>
<td>23</td>
<td>1</td>
<td>AD1*</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>AD2*</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>AD3*</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>AD4*</td>
</tr>
<tr>
<td>21</td>
<td>5</td>
<td>AD5*</td>
</tr>
<tr>
<td>20</td>
<td>6</td>
<td>AD6*</td>
</tr>
<tr>
<td>19</td>
<td>7</td>
<td>AD7*</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>AD8*</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>AD9*</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>AD10*</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>AD11*</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>AD12*</td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>Sweep Holdoff In</td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>Interrupt In (inverted internally)</td>
</tr>
</tbody>
</table>
na 15 Always Zero, grounded internally

*These lines are dependent on the state of RLW (pin25).
Writing a 0(low) to RLW will set lines AD0-AD12 to write mode.
Writing a 1(high) to RLW will set lines AD0-AD12 to read mode.

Default
Not Applicable

**CONTroll:EXTernal:TESTset:SWEepholdoff?**

Applicable Models: N522xB, N523xB, N524xB

(Read-Only) Reads the Sweep Holdoff line (pin 2) on the external test set connector.

Examples

```
CONT:EXT:TEST:SWE?
control:external:testset:sweepholdoff?
```

Return Type

Boolean

TRUE (1) - the pin is set to a TTL High
FALSE (0) - the pin is set to TTL Low

Default
Not Applicable
Control SCPI
Control Handler Commands

Specifies the settings to remotely control the Material Handler IO connector.

Click on a blue keyword to view the command details.

See Also

Example Programs
Synchronizing the Analyzer and Controller
SCPI Command Tree
Learn about the Material Handler IO port.

CONTrol:HANDler:C:MODE <char>

Applicable Models: N522xB, N523xB, N524xB, E5080, All PXIe VNAs with M9341A/B

(Read-Write) Sets and reads the direction of data flow for Port C.

Parameters

<char> Direction of flow. Choose from:

INPut - Port C is used to input data

OUTPut - Port C is used to output data

Examples

CONT:HAND:C:MODE INP
control:handler:c:mode output

Query Syntax

CONTrol:HANDler:C:MODE?

Return Type

Character

Default

INPut
**CONTrol:HANDler:D:MODE <char>**

Applicable Models: N522xB, N523xB, N524xB, E5080, All PXIe VNAs with M9341A/B

*(Read-Write)* Sets and reads the direction of data flow for Port D.

**Parameters**

<char>  
Direction of flow. Choose from:

**INPut** - Port D is used to input data

**OUTPut** - Port D is used to output data

**Examples**

```
CONT:HAND:D:MODE INP
control:handler:d:mode output
```

**Query Syntax**

`CONTrol:HANDler:D:MODE?`

**Return Type**

Character

**Default**  
Input

---

**CONTrol:HANDler:<port>[:DATa] <num>**

Applicable Models: N522xB, N523xB, N524xB, E5080, All PXIe VNAs with M9341A/B

*(Read-Write)* Writes and reads data on the specified port.

**Parameters**

<port>  
Port identifier to set bits for. Choose from:

A,B,C,D,E,F,G,H

<num>  
The number of the data bits to set. Refer to the following table for the maximum number for each port. The minimum number for each port is 0.

<table>
<thead>
<tr>
<th>Port</th>
<th>Max allowable &lt;num&gt;</th>
<th>MSB..................................LSB</th>
<th>Write-only</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>255</td>
<td>A7...A0</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>255</td>
<td>B7...B0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>C</td>
<td>15</td>
<td>C3...C0</td>
<td>Read-Write</td>
</tr>
<tr>
<td>D</td>
<td>15</td>
<td>D3...D0</td>
<td>Read-Write</td>
</tr>
<tr>
<td>E</td>
<td>255</td>
<td>D3...D0 + C3...C0</td>
<td>Read-Write</td>
</tr>
<tr>
<td>F</td>
<td>65535</td>
<td>B7...B0 + A7...A0</td>
<td>Write-only</td>
</tr>
<tr>
<td>G</td>
<td>1048575</td>
<td>C3...C0 + B7...B0 + A7...A0</td>
<td>Write-only</td>
</tr>
<tr>
<td>H</td>
<td>16777215</td>
<td>D3...D0 + C3...C0 + B7...B0 + A7...A0</td>
<td>Write-only</td>
</tr>
</tbody>
</table>

**Note:** When writing to port G, port C must be set to output mode. When writing to port H, both port C and port D must be set to output mode. Use `CONT:HAND:C:MODE OUTP` and `CONT:HAND:D:MODE OUTP`.

**Examples**

```
CONT:HAND:A 254
control:handler:c:data 12
```

**Query Syntax**

`CONT:HAND:<port>:DATA?`

**Return Type**

Numeric

**Default**

Not Applicable

### CONT:HANDler[:EXTension]:INDex:LOGic <char>

**Applicable Models:** E5080B, All PXIe VNAs with M9341A/B, M937xA, P937xA, M980xA/P50xxA with I/O Adapter.

(Read-Write) Sets the logic of the index line. There is no soft front-panel element for this feature.

(M937xA) Index is on "Trig Out" port on the last PXIe module

**Parameters**

<char>

Choose from:

- **POSitive** - Causes the Index line to have positive logic (Low signal indicates that the measurement is complete).
- **NEGative** - Causes the Index lines to have negative logic (High signal indicates that the measurement is complete).
CONT:HAND:IND:LOG NEG
control:handler:extension:index:logic negative

Query Syntax
CONTrol:HANDler[:EXTension]:INDex:LOGIC?

Return Type
Character

Default
POSitive

CONTrol:HANDler[:EXTension]:INDex[:STATe] <bool>

Applicable Models: N522xB, N523xB, N524xB, E5080, All PXIe VNAs with M9341A/B

(Read-Write) Determines the control of Material Handler connector Pin 20. A "Mnemonic not implemented" error is returned if the VNA does NOT have this capability.

Parameters

<bool>
Choose from:
ON (1) - Pin 20 is controlled by the Index signal
OFF (0) - Pin 20 is controlled by Output Port B6

Examples
CONT:HAND:IND 1
control:handler:extension:index:state off

Query Syntax
CONTrol:HANDler[:EXTension]:INDex[:STATe]?

Return Type
Boolean

Default
OFF

CONTrol:HANDler[:EXTension]:RTRigger[:STATe] <bool>

Applicable Models: N522xB, N523xB, N524xB, E5080, All PXIe VNAs with M9341A/B

(Read-Write) Determines the control of Material Handler connector Pin 21. A "Mnemonic not implemented" error is returned if the VNA does NOT have this capability.

Parameters

<bool>
Choose from:
ON (1) - Pin 21 is controlled by the Ready for Trigger signal
OFF (0) - Pin 21 is controlled by Output Port B7

Examples
CONT:HAND:RTR 1
control:handler:extension:rtrigger:state off
**Query Syntax**

```
CONTol:HANDler[:EXTension]:RTRigger[:STATe]?
```

**Return Type**  
Boolean

**Default**  
OFF

---

**CONTol:HANDler:INPut?**

Applicable Models: N522xB, N523xB, N524xB, E5080, M9485A, All PXIe VNAs with M9341A/B

*(Read-Only)* Reads a hardware latch that captures high to low transitions on Input1 of the Material Handler IO. Reading the latch causes it to reset and is ready for the next transition. The hardware latch is only capable of capturing one transition per query. Additional transitions are ignored until after the next query.

Momentarily driving Input1 high, then low, will cause a transition to be detected and latched.

**Examples**

```
CONT:HAND:INP?
control:handler:input?
```

**Return Type**  
Integer - Returns a value of zero or one.

0 - No transition detected since last query.

1 - Transition detected.

**Default**  
Not Applicable

---

**CONTol:HANDler:LOGic <char>**

Applicable Models: N522xB, N523xB, N524xB, E5080

*(Read-Write)* Sets the logic of the Data ports A-H on the Handler connector. Some of these lines are connected internally to the AuxIO.

**Parameters**

**<char>**

Choose from:

- POSitive- Causes the Port lines to have positive logic (high = 1, low = 0).
- NEGative- Causes the Port lines to have negative logic (high = 0, low = 1).

For ports that are in output (write) mode, a change in logic causes the output lines to change state immediately. For example, Low levels change immediately to High levels.

For ports that are in input (read) mode (C,D,E only), a change in logic will be reflected when data is read from that port. For example, if a line read 0, the next read after a logic change will read 1.
CONT:HAND:LOG POS
control:handler:logic negative

Query Syntax
CONTrol:HANDler:LOGic?

Return Type
Character
Default NEGative

CONTrol:HANDler:OUTPut<port>:DAta <num>

Parameters

<port> Output port. Choose from:
1 - output 1 (default)
2 - output 2 (M9341 does not support this)

<num> 0 - Low
1 - High

Examples
CONT:HAND:OUTPut1 1
control:handler:output2:DATA 0

Query Syntax
CONTrol:HANDler:OUTPut<port>:DAta?

Return Type
Boolean
Default 0 - Low

CONTrol:HANDler:OUTPut<port>:USER[:DAta] <num>

Parameters

<port> (Read-Write) Sets or reads the specified USER output line.

Examples
CONT:HAND:OUTPut1 1
control:handler:output2:DATA 0

Query Syntax
CONTrol:HANDler:OUTPut<port>:USER[:DAta]?

Return Type
Boolean
Default 0 - Low

CONTrol:HANDler:OUTPut<port>:USER[:DAta] <num>

Parameters

<port> Type 1 and Type 2 configurations: Returns the last value written to the selected output pin.
Type 3 configuration: Returns the current state of the selected output pin. If an Input1 trigger occurs, the state may not be the same value as was written.

Examples
CONT:HAND:OUTPut1 1
control:handler:output2:DATA 0

Query Syntax
CONTrol:HANDler:OUTPut<num>:DAta?

Return Type
Boolean
Default 0 - Low

CONTrol:HANDler:OUTPut<port>:USER[:DAta] <num>

Parameters

<port> Type 1 and Type 2 configurations: Returns the last value written to the selected output pin.
Type 3 configuration: Returns the current state of the selected output pin. If an Input1 trigger occurs, the state may not be the same value as was written.
Learn about User Output.

Parameters

<port>  
USER Output port. Choose from:

1 - User output 1 (default)

2 - User output port. (M9341 does not support this)

<num>  
0 - Low
1 - High

Examples

CONT:HAND:OUTPut1:USER 1
control:handler:output2:user:data 0

Query Syntax

CONTrol:HANDler:OUTPut<num>:USER:DATA?

Return Type

Boolean

Default

0 - Low

CONTrol:HANDler:PASSfail:LATCh <bool>

Applicable Models: E5080,

(Read-Write) Enable the compatible mode with the E5071C for pass/fail status. If this is enabled, the pass/fail result of the Material Handler IO (pin33) is kept until next measurement. The default pass/fail state set is defined by CONTrol:HANDler:PASSfail:MODe <char>, and this function works independently from the command “CONTrol:HANDler:PASSfail:MODe <char>”. This mode does not provide full compatibility with the E5071C in the timing chart perspective.

Parameters

<char>  
Choose from:
OFF (0)- Off.
ON (1)- Enable Pass/Fail line latch mode. E5071C compatible.

Examples

CONT:HAND:PASS:LATC 1
control:handler:passfail:latch on

Query Syntax

CONTrol:HANDler:PASSfail:LATCh?

Return Type

Boolean

Default

OFF (0)
**CONTrol:HANDler:PASSfail:LOGic <char>**

Applicable Models: N522xB, N523xB, N524xB, E5080, All PXIe VNAs with M9341A/B

(Read-Write) Sets the logic of the PassFail line of the Material Handler IO (pin 33).

**Parameters**

<char>

Choose from:

POSitive- Causes the PassFail line to have positive logic (high = pass, low = fail).

NEGative- Causes the PassFail line to have negative logic (high = fail, low = pass).

**Examples**

```
CONT:HAND:PASS:LOG POS
cONTrol:HANDler:PASSfail:logic negative
```

**Query Syntax**

```
CONTrol:HANDler:PASSfail:LOGic?
```

**Return Type**

Character

**Default**

POSitive

---

**CONTrol:HANDler:PASSfail:MODE <char>**

Applicable Models: N522xB, N523xB, N524xB, E5080, All PXIe VNAs with M9341A/B

(Read-Write) Sets and reads the default logical pass/fail state. This is the state the pass/fail line stays in until a failure occurs (if the mode is NOWait), or until an end-of-sweep condition occurs (if the mode is PASS or FAIL).

The end-of-sweep condition is determined by the **CONTrol:HANDler:PASSfail:SCOPe** command.

**Parameters**

<char>

Choose from:

PASS- the line stays in PASS state until the end-of-sweep condition occurs, at which time the pass/fail line is set, and the pass/fail strobe (line 36) is activated.

FAIL- the line stays in FAIL state until the end-of-sweep condition occurs, at which time the pass/fail line is set, and the pass/fail strobe (line 36) is activated.

NOWait- the pass/fail line is set, and the pass/fail strobe (line 36) is activated as soon as a failure condition occurs.

**Examples**

```
CONT:HAND:PASS:MODE NOW
cONTrol:HANDler:PASSfail:mode fail
```

**Query Syntax**

```
CONTrol:HANDler:PASSfail:MODE?
```
CONTrol:HANDler:PASSfail:SCOPe <char>

Applicable Models: N522xB, N523xB, N524xB, E5080, All PXIe VNAs with M9341A/B

(Read-Write) Sets and reads the scope of pass/fail testing. The pass/fail line on the material handler port can report the status of all measurements on each channel, or all measurements on all channels. This command selects which option to use.

Parameters

<char>  
Choose from:

CHANnel - A pass/fail result is computed and written to the output pins at the end of all sweeps on a channel.

GLOBal - A pass/fail result is computed and written to the output pins at the end of all sweeps on a channel.

If the pass/fail mode is NOWait (as set by CONTrol:HANDler:PASSfail:MODe), the status and strobe pins are written immediately. Otherwise the pins are written as indicated above. Regardless of the mode value, the pass/fail line is returned to its default state (as set by the MODE command) at the end of channel or group of channels.

Examples

```
CONT:HAND:PASS:SCOP CHAN
control:handler:passfail:scope sweep
```

Query Syntax

CONTrol:HANDler:PASSfail:SCOPe?

Return Type  Character

Default  NOWait

CONTrol:HANDler:PASSfail:POLicy <char>

Applicable Models: N522xB, N523xB, N524xB, E5080, All PXIe VNAs with M9341A/B

(Read-Write) Sets the policy used to determine how global pass/fail is computed.

Parameters

<char>  
Name of the policy to use. Choose from:
ALLTests - **Pass/Fail Status** returns PASS if all tests on all measurements pass.

ALLMeas - **Pass/Fail Status** returns PASS if all measurements have associated tests, and all tests pass. FAIL is returned if even one measurement has no associated limit test.

Only those measurements which are not in HOLD mode contribute to the pass/fail result.

### Examples

```
CONT:HAND:P
control:handler:passfail:policy
```

### Query Syntax

```
CONTroll:HANDler:PASSfail:POLicy?
```

### Return Type

**Character**

### Default

ALLTests

---

**CONTroll:HANDler:PASSfail:STATus?**

**Applicable Models:** N522xB, N523xB, N524xB, E5080, M9485A, All PXIe VNAs with M9341A/B

*(Read-Only)* Returns the most recent pass/fail status value. Use this command as follows:

1. Set the VNA trigger scope to GLOBAL
2. Set the VNA trigger source to MANUAL or EXTERNAL.
3. Configure and enable Limit Testing.
4. Trigger the VNA.
5. Use the *OPC? command to determine when the sweep is complete.
6. Use the CONT:HAND:PASS:STAT? query to obtain the global pass/fail result.

### Return Type

**Character** - One of the following is returned:

- **PASS** - all measurements not in HOLD mode have been swept, and all associated limit tests have passed.
- **FAIL** - all measurements not in HOLD mode have been swept, and one or more limit tests failed according to the specified Pass/Fail policy.
- **NONE** - status cannot be determined because measurements are in progress.

### Default

Not Applicable

---

**CONTroll:HANDler:SWEepend <char>**

**Applicable Models:** N522xB, N523xB, N524xB, E5080, All PXIe VNAs with M9341A/B
(Read-Write) Specifies the event that will cause the Handler Sweep End line to strobe. The strobe is at least 10µs in duration, and is activated when all calculations for the associated measurement are complete. This line is connected internally to the Sweep End line of the AUX IO connector.

Parameters

<char> Choose from:
SWEep - the line goes low when each sweep is complete
CHANnel - the line goes low when all the sweeps for each channel is complete.
GLOBal - the line goes low when all sweeps for all channels are complete.

The default state of the passFail line (before a measurement occurs) and after a failure occurs is set by

  CONTrol:HANDler:PASSfail:MODe

Examples

  CONT:HAND:SWE SWE
  control:handler:sweepend channel

Query Syntax

  CONTrol:HANDler:SWEepe?

Return Type

  Character

Default

  GLOBal
# Display Colors

Controls the color settings of the VNA display.

```plaintext
<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISPlay:COLor</td>
</tr>
<tr>
<td>ABACkground</td>
</tr>
<tr>
<td>BACKground</td>
</tr>
<tr>
<td>GRAT1</td>
</tr>
<tr>
<td>GRAT2</td>
</tr>
<tr>
<td>ILaBel</td>
</tr>
<tr>
<td>LIM1</td>
</tr>
<tr>
<td>LOAD</td>
</tr>
<tr>
<td>RESet</td>
</tr>
<tr>
<td>STORe</td>
</tr>
<tr>
<td>TRACe</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
```

Click on a keyword to view the command details.

**see Also**

- Synchronizing the Analyzer and Controller
- Learn about Display and Print Colors
- SCPI Command Tree

---

**DISPlay:COLor<n>:ABACkground <num, num, num>**

**Applicable Models:** All

*(Read-Write)* Set and return the background color for the active window on the VNA display or hardcopy print.

**Parameters**
Colors to modify. Choose from:
1 - Display colors
2 - Print colors
If unspecified, <n> is set to 1 (Display colors).

**<n>, <num, num, num>**

To find RGB values: from the Display Colors dialog, click Change Color, then Define Custom Color.

<table>
<thead>
<tr>
<th>Examples</th>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
</table>
| DISP:COL:ABAC 10,10,10 | DISPlay:COLO<n>:ABACkground? | Numeric (n,n,n) | Display = 0,0,24 (Black)  
Print = 255,255,255 (White) |
| display:color1:abackground 80,80,80 | | | |

**DISPlay:COLO<n>:BACKground <num, num, num>**

**Applicable Models: All**
**(Read-Write)** Set and return the background color for the inactive windows on the VNA display or hardcopy print.

**Parameters**

| <n> | Colors to modify. Choose from:  
1 - Display colors  
2 - Print colors  
If unspecified, <n> is set to 1 (Display colors). |
|----------|-----------------------------|
| <num, num, num> | Numeric. Red, Green, and Blue (RGB values) that specify a color.  
To find RGB values: from the Display Colors dialog, click Change Color, then Define Custom Color. |

<table>
<thead>
<tr>
<th>Examples</th>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
</table>
| DISP:COL:BACK 10,10,10 | DISPlay:COLO<n>:BACKground? | Numeric (n,n,n) | Display = 0,0,0 (Black)  
Print = 255,255,255 (White) |
| display:color1:background 80,80,80 | | | |
**DISPlay:COLor<n>:GRAT1 <num, num, num>**

*Applicable Models: All*

*(Read-Write)* Set and return the labels and grid frame colors in the active window for the VNA display or hardcopy print. *(Active labels, Grid frame)*

**Parameters**

<n>  
Colors to modify. Choose from:

1 - Display colors  
2 - Print colors  

If unspecified, <n> is set to 1 (Display colors).

<num, num, num>  
Numeric. Red, Green, and Blue (RGB values) that specify a color.

To find RGB values: from the Display Colors dialog, click Change Color, then Define Custom Color.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DISP:COL:GRAT1 10,10,10</strong></td>
<td></td>
</tr>
<tr>
<td><strong>display:color1:grat1 80,80,80</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Query Syntax**

DISPlay:COLor<n>:GRAT1?

**Return Type**

Numeric (n,n,n)

**Default**

Display = 175,175,175  
Print = 0,0,0 (Black)

---

**DISPlay:COLor<n>:GRAT2 <num, num, num>**

*Applicable Models: All*

*(Read-Write)* Set and return the inner lines of all grid in all windows, and the grid frame in inactive windows for the VNA display or hardcopy print. *(GRID)*

**Parameters**

<n>  
Colors to modify. Choose from:

1 - Display colors  
2 - Print colors  

If unspecified, <n> is set to 1 (Display colors).

<num, num, num>  
Numeric. Red, Green, and Blue (RGB values) that specify a color.

To find RGB values: from the Display Colors dialog, click Change Color, then Define Custom Color.
Examples

```
DISP:COL:GRAT2 10,10,10
display:color1:grat2 80,80,80
```

Query Syntax

DISPlay:COlor<n>:GRAT2?

Return Type

Numeric (n,n,n)

Default

Display = 100,100,100
Print = 50,50,50

---

**DISPlay:COlor<n>:ILABel <num, num, num>**

**Applicable Models:** All

*(Read-Write)* Set and return the Inactive (not selected) Window Labels for the VNA display or hardcopy print.

**Parameters**

<n>

Colors to modify. Choose from:

1 - Display colors
2 - Print colors

If unspecified, <n> is set to 1 (Display colors).

<num, num, num>

Numeric. Red, Green, and Blue (RGB values) that specify a color.

To find RGB values: from the Display Colors dialog, click Change Color, then Define Custom Color.

Examples

```
DISP:COL:ILAB 10,10,10
display:color1:ilabel 80,80,80
```

Query Syntax

DISPlay:COlor<n>:ILABel?

Return Type

Numeric (n,n,n)

Default

Display = 160,160,160
Print = 0,0,0 (Black)

---

**DISPlay:COlor<n>:LIM1 <num, num, num>**

**Applicable Models:** All

*(Read-Write)* Set and return the limit line color of failed traces or failure indicators (dots) and the word Fail.

**Parameters**

<n>

Colors to modify. Choose from:
1 - Display colors
2 - Print colors

If unspecified, \(<n>\) is set to 1 (Display colors).

\(<\text{num, num, num}>\)

Numeric. Red, Green, and Blue (RGB values) that specify a color.

To find RGB values: from the Display Colors dialog, click Change Color, then Define Custom Color.

Examples

```
DISP:COL:LIMIT 10,10,10
display:color1:limit 80,80,80
```

Query Syntax

```
DISPlay:COLor\(<n>\):LIMIT?
```

Return Type

Numeric (n,n,n)

Default

Display = 255,20,20
Print = 255,20,20

**DISPlay:COLor\(<n>\):LOAD \(<value>\)**

**Applicable Models:** All

*(Write-only)* Load a color theme from a disc file.

**Parameters**

\(<n>\)
Colors to load. Choose from:

1 - Display colors
2 - Print colors

If unspecified, \(<n>\) is set to 1 (Display colors).

\(<value>\)
String. Filename of the stored theme. The .colors suffix is automatically appended.

By default, files are stored in C:\Program Files(x86)\Keysight\Network Analyzer\Colors. To store and load files from a different folder, specify the full path and filename.

Examples

```
DISP:COL:LOAD "myDisplayTheme"
display:color2:load "myPrintTheme"
```

Query Syntax

Not Applicable

Default

Not Applicable

**DISPlay:COLor\(<n>\):RESet**

**Applicable Models:** All
(Write-only) Resets the current theme to the default VNA colors.

Parameters

<n>
Colors to reset. Choose from:
1 - Display colors
2 - Print colors
If unspecified, <n> is set to 1 (Display colors).

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:COL:RES</td>
</tr>
<tr>
<td>display:color2:reset</td>
</tr>
</tbody>
</table>

Query Syntax

Not Applicable

Default

Not Applicable

DISPlay:COLor<n>:STORe <value>

Applicable Models: All

(Write-only) Saves the current color theme to a disc file.

Parameters

<n>
Colors to store. Choose from:
1 - Display colors
2 - Print colors
If unspecified, <n> is set to 1 (Display colors).

<value>
String. Filename. The .colors suffix is automatically appended.
By default, files are stored in C:\Program Files(x86)\Keysight\Network Analyzer\Colors. To store and load files from a different folder, specify the full path and filename.

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:COL:STOR &quot;myDisplayTheme&quot;</td>
</tr>
<tr>
<td>display:color2:store &quot;myPrintTheme&quot;</td>
</tr>
</tbody>
</table>

Query Syntax

Not Applicable

Default

Not Applicable

DISPlay:COLor<n>:TRACe<nth>:DATA <num, num, num>

Applicable Models: All

(Read-Write) Set and return the color of Data and Limit Lines for nth trace in a window.

Parameters
Colors to modify. Choose from:
1 - Display colors
2 - Print colors
If unspecified, <n> is set to 1 (Display colors).

Numeric. Relative trace number in the window for which colors are set. This is not necessarily the trace number. Learn more.
Choose from 1 to 8.
If unspecified, <nth> is set to 1 (first trace).

Numeric. Red, Green, and Blue (RGB values) that specify a color.
To find RGB values: from the Display Colors dialog, click Change Color, then Define Custom Color.

Examples

```
DISP:COL:TRAC2:DATA 10,10,10
display:color1:trace5:DATA 80,80,80
```

Query Syntax

```
DISPlay:COLor<n>:TRACe<nth>:DATA?
```

Return Type

Numeric (n,n,n)

Default

Varies for each trace.

```
DISP:COLor<n>:TRACe<nth>:MARKer <num, num, num>
```

**Applicable Models:** All
**(Read-Write)** Set and return the color of data trace markers for nth trace in a window.

Parameters

Colors to modify. Choose from:
1 - Display colors
2 - Print colors
If unspecified, <n> is set to 1 (Display colors).

Numeric. Relative trace number in the window for which colors are set. This is not necessarily the trace number. Learn more.
Choose from 1 to 8.
If unspecified, <nth> is set to 1 (first trace).

Numeric. Red, Green, and Blue (RGB values) that specify a color.
To find RGB values: from the Display Colors dialog, click Change Color, then Define Custom Color.
Examples

<table>
<thead>
<tr>
<th>Display Colors</th>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:COL:TRAC2:MARK 10,10,10</td>
<td>display:color1:trace5:marker 80,80,80</td>
<td>Numeric (n,n,n)</td>
<td>Varies for each trace.</td>
</tr>
</tbody>
</table>

Applicable Models: All

Set and return the memory trace color for nth trace in a window.

Parameters

<n>
Colors to modify. Choose from:
- 1 - Display colors
- 2 - Print colors
If unspecified, <n> is set to 1 (Display colors).

<nth>
Numeric. Relative trace number in the window for which colors are set. This is not necessarily the trace number. Learn more.
Choose from 1 to 8.
If unspecified, <nth> is set to 1 (first trace).

<num, num, num>
Numeric. Red, Green, and Blue (RGB values) that specify a color.
To find RGB values: from the Display Colors dialog, click Change Color, then Define Custom Color.

Examples

<table>
<thead>
<tr>
<th>Display Colors</th>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:COL:TRAC2:MEM 10,10,10</td>
<td>display:color1:trace5:memory 80,80,80</td>
<td>Numeric (n,n,n)</td>
<td>Varies for each trace.</td>
</tr>
</tbody>
</table>

Applicable Models: All

Set and return the color of memory trace markers for nth trace in a window.

Parameters

<n>

<nth>

<num, num, num>
Parameters

<n>
Colors to modify. Choose from:
1 - Display colors
2 - Print colors
If unspecified, <n> is set to 1 (Display colors).

<nth>
Numeric. Relative trace number in the window for which colors are set. This is not necessarily the trace number. Learn more.
Choose from 1 to 8.
If unspecified, <nth> is set to 1 (first trace).

<num, num, num>
Numeric. Red, Green, and Blue (RGB values) that specify a color.
To find RGB values: from the Display Colors dialog, click Change Color, then Define Custom Color.

Examples

DISP:COL:TRAC2:MMAR 10,10,10
display:color1:trace5:mmarker 80,80,80

Query Syntax

DISPlay:COLOr<n>:TRACe<nth>:MMARker?

Return Type

Numeric (n,n,n)

Default

Varies for each trace.
Local Lockout

Normally, a GPIB instrument is put in remote mode by asserting the Remote Enable (REN) GPIB line. At that time, all front panel keys (except the Local key) are locked to prevent user interaction.

Sending the LLO message over the GPIB further locks out the keyboard, mouse, and Local key during execution of your GPIB program. The syntax of the LLO message depends on the GPIB driver you are using. Consult your GPIB driver software users manual.

The VNA requires these two actions to occur in order:

- Controller sends the LLO (Local Lockout) message
- Controller asserts the REN (Remote Enable) GPIB line

The VNA will then go into remote mode with full lockout capability.

This feature is also supported using SICL over LAN.

Use the `LocalLockoutState` COM command when using TCPIP/LAN.
CorrCollGuidPSens
Sense:Correction:Collect:Guided:PSensor Commands

Configures the power sensors to be used during a SmartCal (Guided) Power Calibration.

Three of these commands can be used with Applications Channels. Learn more.

```
SENSe:CORRection:COLLect:GUIDed:PSEN<n>:CKIT
SENSe:CORRection:COLLect:GUIDed:PSEN<n>:CONN
```

Click on a keyword to view the command details.

**Notes**

EXCEPT for the following THREE commands, the commands listed on this page are supported ONLY on standard channels.

These can be used with Application channels. See NFX example.

```
SENSe:CORRection:COLLect:GUIDed:PSEN<n>:CKIT
SENSe:CORRection:COLLect:GUIDed:PSEN<n>:CONN
```

When using two sensors with a Dual Power Meter, use SOUR:POW:CORR:COLL:<pmChan>SEN:SEL to select a power sensor.

See Also

SENSe:CORRection:COLLect:GUIDed commands
SENSe<ch>:CORRection:COLLect:GUIDed:PSENor<pnum>:CKIT <kit>, [src]

(Read-Write) Specifies the calibration kit to be used when an adapter is necessary to connect the power sensor to the port <pnum> during a guided calibration. This command can also be used with Application channels. See NFX example. When used with Guided Power Cal, first enable a power cal using SENS:CORR:COLL:GUID:PSEN ON

Specify the connector type for the adapter with SENS:CORR:COLL:GUID:PSEN:CONN

Query the valid available kits for the connector with SENS:CORR:COLL:GUID:CKIT:CAT? <conn>

Specify the kit using this command.

Perform a query of this command. If the <kit> parameter was incorrectly entered, an error will be returned.

Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<pnum> Source port being calibrated. If unspecified, value is set to 1. For NFX cals, port number must be 1.

<kit> Calibration kit to be used for the specified port.

When using an ECal module, include the characterization name in the <kit> string. Use SENS:CORR:COLL:GUID:CKIT:CAT? to read the list of characterizations available in the module and in VNA disk memory.

If two or more identical ECal modules are connected to the VNA, the serial number must be included to distinguish the ECal modules.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2".

Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

See Remotely Specifying a Source Port

Examples

SENS:CORR:COLL:GUID:PSEN1:CKIT '85055A'
'The following includes a serial number when two or more ECal mods are connected
SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:CONNector <conn>, [src]

(Read-Write) Specifies the connector type for a power sensor when a power calibration is part of the Guided Calibration process. Valid connector names are stored within calibration kits. Some cal kits may include both male and female connectors. Therefore, specifying connector gender may be required.

This command can also be used with Application channels. See NFX example.

Follow these steps to ensure port connectors are specified correctly:

When used with Guided Power Cal, first enable a power cal using SENSe:CORR:COLL:GUID:PSEN ON


Set a connector type for each port using this command.

Perform a query of this command. If the connector type was incorrectly entered, an error will be returned.

Specify the cal kit to use for each port with SENSe:CORR:COLL:GUID:PSENsor:CKIT

Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<pnum> Source port being calibrated. If unspecified, value is set to 1.

<conn> String - Power sensor connector type to connect to the specified VNA port <pnum>.

Because the default for this argument is "Ignored", by specifying a connector type, you imply that you want to calibrate and correct for the effects of the adapter.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode.
balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2".

Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

See Remotely Specifying a Source Port

Examples

```
SENS:CORR:COLL:GUID:PSEN1:CONN "3.5 mm (50) male"
```

```
sense:correction:collect:guided:psensor1:connector "3.5 mm (50) female"
```

See example program

Query Syntax

```
SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:CONNector?
```

Return Type

String

Default

"Ignored"

---

**SENS<ch>:CORR<e>ction:COLL<e>ct:GUID<ed>PSENsor<pnum>:MULT<e>l<e>e:ADD <s<tring>>**

**Applicable Models:** All

*(Write-only) Adds a power sensor to be used during a Guided Power Calibration. Use when multiple power sensors are to be used to calibrate the entire frequency span. The Name argument is used to recognize the sensor in the User Interface. An item number is automatically assigned to the sensor. Use that number to refer to the sensor. The item number of the newly-added sensor is always equal to the number returned by SENS:CORR:COLL:GUID:PSEN:MULT:COUN? after the ADD.*

**Note:** The "multiple sensors" commands are supported ONLY on standard channels.

Learn about using multiple power sensors

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;ch&gt;</code></td>
<td>Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td><code>&lt;pnum&gt;</code></td>
<td>Source port being calibrated. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td><code>&lt;string&gt;</code></td>
<td>Power sensor name to add. The power sensor must be already configured as a PMAR device using this name. Learn how to remotely configure a PMAR device.</td>
</tr>
</tbody>
</table>

**Examples**

```
```

See example program

**Query Syntax**

```
SENSe<ch>:CORR<e>ction:COLL<e>ct:GUID<ed>PSENsor<pnum>:MULT<e>l<e>e:ADD <s<tring>>?
```

Not Applicable
SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<id>:CKIT <string>

(Read-Write) Set and read the Cal Kit to be used when an adapter is necessary to connect the power sensor to the cal plane and you choose to remove its effects from the measurement. Use this command when multiple power sensors are to be used to calibrate the entire frequency span. Learn about using multiple power sensors.

Applicable Models: All

Parameters

<ch>
Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<pnum>
Source port being calibrated. If unspecified, value is set to 1.

<id>
Power sensor item number. This 1-based number is assigned automatically when a power sensor is added using SENS:CORR:COLL:GUID:PSEN:MULT:ADD. Use SENS:CORR:COLL:GUID:PSEN:MULT:COUN? to return the number of power sensor items that are configured for use on the channel.

<string>
Calibration kit to be used for the specified source port. When using an ECal module, include the characterization name in the <kit> string. Use SENSE:CORR:COLL:GUID:CKIT:CAT? to read the list of characterizations available in the module and in VNA disk memory.

If two or more identical ECal modules are connected to the VNA, the serial number must be included to distinguish the ECal modules.

Examples

SENS:CORR:COLL:GUID:PSEN1:MULT2:CKIT '85055A'
'The following includes a serial number when two or more ECal mods are connected.'

sense:correction:collect:guided:psensor2:multiple1:ckit '85092-60010 ECal 10685'

See example program

Query Syntax

SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<id>:CKIT?

Return Type

String

Default

Not Applicable

SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<id>:CONNECTor <conn>

(Read-Write) Set and read the connector type of a power sensor to be used when an adapter is necessary to connect the power sensor to the cal plane and you choose to remove its effects from the measurement. Use this command when multiple power sensors are to be used to calibrate the entire frequency span. Learn about using multiple power sensors.

Applicable Models: All

Parameters
### SENSE<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<id>:CONNector?

**Description:**
Returns the string value of the power sensor connector type to connect to the specified source port <pnum>. Because the default for this argument is 'Ignored', by specifying a connector type, you imply that you want to calibrate and correct for the effects of the adapter.

**Query Syntax:**
```
SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<id>:CONNector?
```

**Return Type:**
String

**Default:**
"Ignored"

**Examples:**
```
SENS:CORR:COLL:GUID:PSEN1:MULT1:CONN "3.5 mm (50) male"
```

**See example program**

---

### SENSE<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<COUNT?

**Description:**
(Read-only) Returns the number of configured power sensors to be used during a Guided Power Calibration.

**Query Syntax:**
```
SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<COUNT?
```

**Return Type:**
Numeric

**Default:**
Not Applicable

**Examples:**
```
SENS:CORR:COLL:GUID:PSEN1:MULT1:COUN?
```

**See example program**

---

### SENSE<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<id>:FREQuency:STARt <num>

**Description:**
(Read-Write) Set and read the start frequency for the specified power sensor (port number). Use this command when multiple power sensors are to be used to calibrate the entire frequency span.

**Query Syntax:**
```
SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<id>:FREQuency:STARt <num>
```

---

**Parameters**
- `<ch>`: Channel being calibrated, depending on the `CHAN:MODE` setting. If unspecified, value is set to 1.
- `<pnum>`: Source port being calibrated. If unspecified, value is set to 1.
- `<id>`: Power sensor item number. This 1-based number is assigned automatically when a power sensor is added using `SENS:CORR:COLL:GUID:PSEN:ADD`. Use `SENS:CORR:COLL:GUID:PSEN:MULT:COUN?` to return the number of power sensor items that are configured for use on the channel.
- `<conn>`: String - Power sensor connector type to connect to the specified source port `<pnum>`. Because the default for this argument is 'Ignored', by specifying a connector type, you imply that you want to calibrate and correct for the effects of the adapter.
Parameters

<ch>
Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<pnum>
Source port being calibrated. If unspecified, value is set to 1.

{id>
Power sensor item number. This 1-based number is assigned automatically when a power sensor is added using SENS:CORR:COLL:GUID:PSEN:MULT:ADD. Use SENS:CORR:COLL:GUID:PSEN:MULT:COUN? to return the number of power sensor items that are configured for use on the channel.

<num>
Start frequency in Hz. Choose a value between the start frequency and stop frequency of the VNA.

Examples

SENS:CORR:COLL:GUID:PSEN1:MULT1:FREQ:STAR 1e9
See example program

Query Syntax

SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<id>:FREQuency:STARt?

Return Type

Numeric

Default

Start frequency of the VNA


(Read-Write) Set and read the stop frequency for the specified power sensor (port number). Use this command when multiple power sensors are to be used to calibrate the entire frequency span. Learn about using multiple power sensors

Parameters

<ch>
Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<pnum>
Source port being calibrated. If unspecified, value is set to 1.

{id>
Power sensor item number. This 1-based number is assigned automatically when a power sensor is added using SENS:CORR:COLL:GUID:PSEN:MULT:ADD. Use SENS:CORR:COLL:GUID:PSEN:MULT:COUN? to return the number of power sensor items that are configured for use on the channel.

<num>
Stop frequency in Hz. Choose a value between the start frequency and stop frequency of the VNA.

Examples

SENS:CORR:COLL:GUID:PSEN1:MULT1:FREQ:STOP 2e9
See example program

Query Syntax

SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<id>:FREQuency:STOP?
SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<id>:NAME <string>

(Read-Write) Set and read the name of the power sensor. Learn about using multiple power sensors.

Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<pnum> Source port being calibrated. If unspecified, value is set to 1.

<id> Power sensor item number. This 1-based number is assigned automatically when a power sensor is added using SENS:CORR:COLL:GUID:PSEN:MULT:ADD. Use SENS:CORR:COLL:GUID:PSEN:MULT:COUN? to return the number of power sensor items that are configured for use on the channel.

<string> Sensor name. The power sensor must be already configured as a PMAR device using this name. Learn how to remotely configure a PMAR device.

Examples

SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor1:MULT1:NAME "26GHzPwrSensor"

See example program

Query Syntax

SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple<id>:NAME?

Return Type

String

Default


(Applicable Models: All)

(Write-only) Deletes a power sensor from the sensors to be used during a Guided Power Calibration.

Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<pnum> Source port being calibrated. If unspecified, value is set to 1.

<id> Power sensor item number to remove. Use SENS:CORR:COLL:GUID:PSEN:MULT:COUN? to return the number of power sensor items that are configured for use on the channel.
### SENSE<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple[:STATe] <value>

*(Read-Write)* Enables the use of multiple power sensors to calibrate the entire frequency span of the channel. Learn about using multiple power sensors.

**Parameters**

- `<ch>`: Channel being calibrated, depending on the `CHAN:MODE` setting. If unspecified, value is set to 1.
- `<pnum>`: Source port being calibrated. If unspecified, value is set to 1. ONLY one port may be calibrated with a Guided Power Cal.
- `<value>`: Boolean
  - ON (or 1): Use multiple power sensors.
  - OFF (or 0): Do NOT use multiple power sensors.

**Examples**

```
SENS:CORR:COLL:GUID:PSEN1:MULT 0
sense:correction:collect:guided:psensor2:multiple:state ON
```

**Query Syntax**

```
SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:MULTiple[:STATe]? 
```

**Return Type**

Boolean

**Default**

OFF (0)

### SENSE<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:POWer:LEVel <value>, [src]

*(Read-Write)* Specifies the power level at which to perform a power calibration during a guided calibration. This command can also be used with Application channels. See NFX example.

When used with Guided Power Cal, first enable a power cal using `SENS:CORR:COLL:GUID:PSEN ON`.

**Parameters**

- `<ch>`: Channel being calibrated, depending on the `CHAN:MODE` setting. If unspecified, value is set to 1.
Source port being calibrated. If unspecified, value is set to 1. For NFX cals, port number must be 1.

Power level in dBm at which to perform the power calibration. Learn more.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2".

Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

See Remotely Specifying a Source Port

**Examples**

```
SENS:CORR:COLL:GUID:PSEN1:POW:LEV 0
Setting calibration power for Gain Compression Converters (GCX)
SENS:CORR:COLL:GUID:PSEN1:POW:LEV -21 'Sets the cal power over the input frequencies where port 1 is the input of the mixer
SENS:CORR:COLL:GUID:PSEN2:POW:LEV -6 'Sets the cal power over the output frequencies where port 2 is the output of the mixer
```

See example program

**Query Syntax**

SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:POWer:LEVel?

**Return Type**

Numeric

**Default**

0

SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:POWTable <file>, [src]

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Loads a file that defines a power table to be used during a SMC Guided Power Cal or Cal All Channels on a mmWave system. This feature is available because power sensors are NOT typically available at mmWave frequencies. Learn more.

**Parameters**

<ch>

Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<pnum>

Source port being calibrated. If unspecified, value is set to 1. For NFX cals, port number must be 1.
(String) Full path and filename of a *.prn file that defines the power table. An error is returned if the file is not found.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2".

Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

See Remotely Specifying a Source Port

Examples

SENS:CORR:COLL:GUID:PSEN1:POWT "C:/powertable1.prn"

SENSe:CORRection:COLLect:GUIDed:PSENsor<pnum>:POWTable?

Query Syntax

SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>:POWTable?

Return Type

String

Default

Not Applicable

SENSe<ch>:CORRection:COLLect:GUIDed:PSENsor<pnum>[:STATe] <value>, [src]

(Applicable Models: All)

(Read-Write) Enables Guided Power Cal and sets the source port to be calibrated.

Parameters

<ch> Channel being calibrated, depending on the CHAN:MODE setting. If unspecified, value is set to 1.

<pnum> Source port being calibrated. If unspecified, value is set to 1. ONLY one port may be calibrated with a Guided Power Cal.

<value> Boolean

ON (or 1) Perform Power Cal.
OFF (or 0) Do NOT perform Power Cal.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port VNA-x model with multiple sources such as "Port 1 Src2".

8139
Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

See Remotely Specifying a Source Port

Examples

SENS:CORR:COLL:GUID:PSEN1 0
sense:correction:collect:guided:psensor2:state ON

See example program

Query Syntax
SENSe<ch>:CORRection:COLLect:GUIDed:PSEN sor<pnum>[;STATe]?

Return Type
Boolean

Default
OFF (0)
These commands, along with the standard Guided commands, performs a SMC calibration on a frequency converting device.

Note: These commands replace the SENSE:CORRectionCOLLect:Session:SMC commands. These commands allow the entire SMC cal to be performed from the Guided cal interface.

These commands are also used to perform a GCx Calibration. See an example.

<table>
<thead>
<tr>
<th>SENSe:CORRectionCOLLect:GUIDed:SMC</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSIMulator</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>IMPort</td>
</tr>
<tr>
<td><strong>LO:PCAL[:STATE]</strong></td>
</tr>
<tr>
<td>PHASe</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>PWRCal</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Click on a red keyword to view the command details.

Red keywords are obsolete.

See Also
- All GUIDed commands
- Example Programs
- Learn about SMC Calibrations
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

NOTE: To configure a power meter and sensor see SOURce:POWer: commands.

Applicable Models: All with SMC Options (S9x082A/B, S9x083A/B)
(Read-Write) Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement. Learn more.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<p> Apply network to input or output of mixer. Choose from:
1 - Input of mixer
2 - Output of mixer

<string> Filename of the S2P used for embedding or de-embedding. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

Examples


Query Syntax


Return Type String

Default Not Applicable


Applicable Models: All with SMC Options (S9x082A/B, S9x083A/B)
(Read-Write) Allows you to embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement. Learn more.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<p> Apply network to input or output of mixer. Choose from:
1 - Input of mixer
2 - Output of mixer

<char> Choose from:

NONE - Do nothing with effects of S2P file.
EMBed - Add effects of S2P file from the measurement results.
DEEMbed - Remove effects of S2P file from the measurement results.

Examples


Query Syntax

SENSe<ch>:CORRection:COLLect:GUIDed:SMC:IMP <calName>, <dataset>

Applicable Models: All with SMC Options (S9x082A/B, S9x083A/B)
(Write-only) Imports Guided Power Cal or Phase Reference Cal into the current SMC calibrations.

For the Guided Power Cal:
The port of the mixer input must have the same source attenuator setting between the SMC channel and the
Guided Power Cal Set. The frequencies of the Guided Power Cal must include all the mixer frequencies.
Interpolation will be applied to the Guided Power Cal frequencies if they do not exactly match.

For the Phase Reference Cal:
The port of the mixer input must have the same source attenuator setting as used in the phase reference cal. The
phase reference cal must include all the mixer frequencies. Interpolation will be applied to the phase reference cal
frequencies if they do not exactly match. Learn more about Phase Reference Cal.

The following error message may appear (it is not written to the VNA Error Log):

Interpolation target is out of range. Cannot interpolate when incompatible frequency ranges occur.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1
<calName> (String) Name of existing Cal Set from which to import power data.
<dataset> (String) Name of the data set. Choose from:
POWER_STEP -import the Guided Power Cal data.
"POWER_AND_PHASE" - Import the Phase Reference + power cal data. When this command is
sent, the SMC Cal Method is automatically set to Use Phase Reference Cal. Learn more. There
is no other command to set this.

Examples

See example program
SENS:CORR:COLL:GUID:SMC:IMP "MyPhaseRefCal","POWER_AND_PHASE"
See example program

Query Syntax Not Applicable

Default Not Applicable


Applicable Models: All with SMC Options (S9x082A/B, S9x083A/B)
(Read-Write) Sets and returns whether or not the LO power cal step is included in the cal steps when an SMC or GCx cal is performed. Learn more.


**Parameters**

- `<cnum>`
  - Any existing channel number; if unspecified, value is set to 1.
- `<n>`
  - LO Stage. Choose 1.
- `<bool>`
  - LO Power Cal state. Choose from:
    - O or OFF - Skips over the LO Power Cal when calibrating.
    - 1 or ON - Includes a step for LO Power Cal when calibrating

**Examples**

```
SENS:CORR:COLL:GUID:SMC:LO1:PCAL 0
```

**Query Syntax**


**Return Type**

Boolean

**Default**

OFF or 0

---


Applicable Models: All with SMC Options (S9x082A/B, S9x083A/B) (Except M9485A)

(Read-Write) Set and return the known delay through the calibration mixer. Learn more.

**Parameters**

- `<ch>`
  - Any existing channel number. If unspecified, value is set to 1.
- `<char>`
  - Known delay through the calibration mixer in seconds.

**Examples**

```
```

**Query Syntax**


**Return Type**

Numeric

**Default**

0 seconds

---

To select Phase Reference Cal method for correcting an SMC+Phase measurement, use SENS:CORR:COLL:GUID:SMC:IMPort

Parameters

-ch-

Any existing channel number. If unspecified, value is set to 1

<char>

Choose from:

FIXed - use a known delay value set with SENS:CORR:COLL:GUID:SMC:PHAS:DEL
MIXer - use the S2P file set with SENS:CORR:COLL:GUID:SMC:PHAS:MIX

Examples


Query Syntax
SENSe<ch>:CORRection:COLLect:GUIDed:SMC:PHASe:METHod?

Return Type
Character

Default
FIXed

SENS<ch>:CORRection:COLLect:GUIDed:SMC:PHASe:MIXer <string>

Applicable Models: All with SMC Options (S9x082A/B, S9x083A/B) (Except M9485A)
(Write-only) Set the filename of the S2P file used to characterize the calibration mixer. Learn more.

Parameters

-ch-

Any existing channel number. If unspecified, value is set to 1

<string>

Calibration mixer filename. Use the following rules to specify path names:
The default folder is "D:\"
You can change the active directory using MMEMory:CDIRectory.
Specify only the file name if using the active directory.
You can also use an absolute path name to specify the folder and file.

Examples

SENS:CORR:COLL:GUID:SMC:PHAS:MIX "MyCalMixer.s2p"
sense2:correction:collect:guided:smc:phase:mixer "MyCalMixer.s2p"

Query Syntax
Not Applicable

Default
Not Applicable
Applicable Models: All with SMC Options (S9x082A/B, S9x083A/B)

(Write-only) Aborts a power cal. This command should be sent when a power cal is running.

**Parameters**

<ch>  
Channel number of the SMC cal being performed. If unspecified, value is set to 1.

**Examples**

```
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---


Applicable Models: All with SMC Options (S9x082A/B, S9x083A/B)

(Read-Write) Set whether to use the reference receiver for faster iteration during power cal or use a power meter.

This command should be set during calibration setup. Learn more.

**Parameters**

<ch>  
Any existing channel number. If unspecified, value is set to 1.

<bool>  
OFF or 0 - Use a power meter only.
ON or 1 - Use a power meter for the first reading and the reference receiver for all subsequent readings.

**Examples**

```
SENS:CORR:COLL:GUID:SMC:PWRC:REC 1
```

**Query Syntax**

```
```

**Return Type**

Boolean

**Default**

ON or 1

---


Applicable Models: All with SMC Options (S9x082A/B, S9x083A/B)

(Read-Write) Specifies whether to use a Thru standard or to use two power sensor connections during the power cal of an SMC calibration.

**Note:** This command must be sent BEFORE ALL other calibration commands.

**Parameters**

<ch>  
Any existing channel number. If unspecified, value is set to 1.
OFF or 0 - Perform Cal with Thru standard.
ON or 1 - Do NOT use a Thru, but instead perform separate power cals on Input and Output reference planes.

Examples

'The following is an example sequence of commands:
SENSe<ch>:CORRection:COLLect:GUIDed:CONN:PORT1 "APC 3.5 female"
SENSe<ch>:CORRection:COLLect:GUIDed:CONN:PORT2 "APC 3.5 female"
SENSe<ch>:CORRection:COLLect:GUIDed:CONN:PORT3 "Not used"
SENSe<ch>:CORRection:COLLect:GUIDed:CONN:PORT4 "Not used"
SENSe<ch>:CORRection:COLLect:GUIDed:CKIT:PORT1 "N4691-60006 ECal 02638"
SENSe<ch>:CORRection:COLLect:GUIDed:INIT

Query Syntax

SENSe<ch>:CORRection:COLLect:GUIDed:SMC:PWRCal:SEPare?  

Return Type

Boolean

Default

OFF or 0
CorrCollGuidVMC
Sense:Correction:Collect:Guided:VMC Commands

Performs a VMC calibration on a frequency converting device.

Note: These commands replace the SENSE:CORRectionCOLLect:Session:VMC commands. These commands allow the entire VMC cal to be performed from the Guided cal interface.

SENSe:CORRectionCOLLect:GUIDed:VMC

| FSIMulato
| NETWork
| FILename
| MODE
LO:PCAL[:STATE]

MiXer
| CHARacterize
| CAL
| FILename
| OPTION
| REVerse

| ECAL
| PORTmap

OPERation

Click on a red keyword to view the command details.

See Also
- All GUIDED Cal commands
- Example Programs
- Learn about VMC Calibration
- Synchronizing the Analyzer and Controller


Applicable Models: All with VMC Options (S9x083A/B)
(Read-Write) Allows you to embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement. Learn more.
Parameters

<ch>  Channel number to be calibrated. If unspecified, value is set to 1.

<x>  Apply network to input or output of mixer. Choose from:
1 - Input of mixer
2 - Output of mixer

<char>  Choose from:
NONE - Do nothing with effects of S2P file.
EMBed - Add effects of S2P file from the measurement results.
DEEMbed - Remove effects of S2P file from the measurement results.

Examples


Query Syntax

Return Type  Character

Default  NONE


Applicable Models: All with VMC Options (S9x083A/B)
(Read-Write) Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement. Learn more.

Parameters

<ch>  Channel number to be calibrated. If unspecified, value is set to 1.

<x>  Apply network to input or output of mixer. Choose from:
1 - Input of mixer
2 - Output of mixer

<string>  Filename of the S2P used for embedding or de-embedding. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

Examples

Query Syntax

Return Type  String

Default  Not Applicable

Applicable Models: All with VMC Options (S9x083A/B)

(Read-Write) Sets and returns whether or not the LO power cal step is included in the cal steps when a VMC cal is performed. Learn more.

Parameters

- `<cnum>`  
  Any existing channel number; if unspecified, value is set to 1.
- `<n>`  
  LO Stage. Choose 1.
- `<bool>`  
  LO Power Cal state. Choose from:
  - O or OFF - Skips over the LO Power Cal when calibrating.
  - 1 or ON - Includes a step for LO Power Cal when calibrating

Examples

```
SENS:CORR:COLL:GUID:VMC:LO1:PCAL 0
```

Query Syntax

```
```

Return Type  

Boolean

Default  

OFF or 0

---


Applicable Models: All with VMC Options (S9x083A/B)

(Read-Write) Specifies the .S2P filename used for mixer characterization. Use the VMC:MIXer:CHARacterize:CAL:OPTion command to load the file for mixer characterization. Once loaded, use this command to query the current filename or set a new filename.

Parameters

- `<ch>`  
  Channel number to be calibrated. If unspecified, value is set to 1.
- `<string>`  
  Filename of the S2P used for mixer characterization. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

Examples

```
```

Query Syntax

```
```

Return Type

String

Default

D:\default.s2p

---


Applicable Models: All with VMC Options (S9x083A/B)

(Read-Write) Sets the mixer characterization method to CKIT (Ecal or Mechanical) or load a previously-performed mixer characterization (*.s2p) file. Learn about the requirements of the characterization mixer file.
Parameters

<ch>
Channel number to be calibrated. If unspecified, value is set to 1.

<char>
Choose from:

CKIT - Use an ECal or Mechanical Cal Kit for the mixer characterization. Use SENS:CORR:COLL:GUID:CKIT:PORT to specify the cal kit. For the output of the calibration mixer, specify port 3. If port 3 is already used for the output of the DUT mixer, then specify port 4.

FILE, <filename> - Retrieve a mixer characterization file. Also specify the filename of the .S2P used for mixer characterization. Use the VMC:CHARacterize:CAL:FiLeName command to query the filename.

Examples

'S or

Query Syntax


Return Type

String

Default

CKIT


Applicable Models: All with VMC Options (S9x083A/B)
(Read-Write) Specifies the direction in which to characterize the calibration mixer. Learn more about the calibration mixer.

Parameters

<ch>
Channel number to be calibrated. If unspecified, value is set to 1.

<bool>

OFF (0) - Characterize the calibration mixer in the SAME direction as that specified in the mixer setup.

ON (1) - Characterize the calibration mixer in the REVERSE direction as that specified in the mixer setup.

Examples


Query Syntax


Return Type

Boolean

Default

OFF


Applicable Models: All with VMC Options (S9x083A/B)
(Read-Write) Sets the port mapping for the mixer characterization with ECal. This command is required if SENS:CORR:COLL:GUID:VMC:MIX:CHAR:CAL:OPT ECAL is specified.
Parameters

<ch> Channel number to be calibrated. If unspecified, value is set to 1.

<mod> 1 - Electronic Calibration Module

<string> Choose from:

"A1" - ECAL module port A is connected to VNA port 1
"B1" - ECAL module port B is connected to VNA port 1

Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
</table>

Query Syntax

SENSe<ch>:CORRection:COLLect:GUIDed:VMC:MIXer:ECAL:PORTmap <mod>?

Return Type String

Default "A1"

SENSe<ch>:CORRection:COLLect:GUIDed:VMC:OPERation <string>

Applicable Models: All with VMC Options (S9x083A/B)

(Read-Write) Perform either full VMC calibration or mixer characterization only.

Parameters

<ch> Channel number to be calibrated. If unspecified, value is set to 1.

<char> 'CAL' - full calibration and mixer characterization

'CHAR' - mixer characterization only (no reference mixer required) - Saves an .S2P file with the filename specified in SENSe<ch>:CORR:COLL:GUID:VMC:CHARacterize:CAL:FILename <filename>. If none is specified, a filename is automatically generated and can be queried using the filename command.

Examples

SENSe<ch>:CORRection:COLLect:GUIDed:VMC:OPER 'CAL'

Query Syntax

SENSe<ch>:CORRection:COLLect:GUIDed:VMC:OPERation?

Return Type String

Default "CAL"
ECalCharacterize
SENSe:CORR:CKIT:ECAL:CHARacterize Commands

Controls the settings used to perform an ECal User Characterization. These commands do NOT perform the calibration that is required before measuring the ECal module. An S-Parameter channel must already be calibrated. Learn more.

SENSe:CORR:CKIT:ECAL:CHARacterize:

ACQuire
CNUMber
CONNector
| CATalog?
| PORT<n>:SELect
DESCription
| PORT<n>:SELect
| [STEP]?
| USER
| VNA
DMEMory
| SAVE
ID
INIItiate
INSitu
| ENABle
| [STATe]
SAVE
STEPS?

Click on a keyword to view the command details.

Notes:

These commands provide for the following:
Measure the ECal module with adapters, cables, or fixtures to be included in the User Characterization. Allow descriptive text to be entered.

Save the User Characterization to the ECal module or VNA disk memory. Up to 12 User Characterizations can be stored in an ECal module. Learn more.

You can NOT perform a remote User Characterization of a 4-port ECal module using a 2-port VNA. This can only be done from the front panel user interface.

See Also

Example - Perform an ECal User Characterization
Learn about ECal User Characterization
Synchronizing the Analyzer and Controller
SCPI Command Tree

SENSe<ch>:CORRection:CKIT:ECAL:CHARacterize:ACQuire STAN<step>

Applicable Models: All
(Write-only) Initiates the measurement of the ECal module. The user characterization process must have been initiated first using SENS:CORR:CKIT:ECAL:CHAR:INIT. Currently, only ONE step is required to measure the ECal module.

Note: This command is an overlapped command. When *OPC is issued with it, the OPC bit in the VNAs Standard Event Status Register is not set until this command has completed its operation. Learn more.

Parameters

<ch> Channel number being calibrated. If unspecified, value is set to 1. Channel number being calibrated.

<step> Integer User characterization step number to be measured.

Examples

SENS:CORR:CKIT:ECAL:CHAR:ACQ STAN1, *OPC

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:CKIT:ECAL:CHARacterize:CNUMber <n>

Applicable Models: All
(Read-Write) Sets and reads the number to which the User Characterization will be stored in the ECal module. The number must be set before sending SENS:CORR:CKIT:ECAL:CHAR:INIT or the default value (1) will be used. This command is NOT necessary when saving the User Characterization to the VNA disk memory.

Parameters

<ch> Channel number being calibrated. If unspecified, value is set to 1
User Characterization number. Choose a value between 1 and 12.

**Examples**

```
SENS:CORR:KIT:ECAL:CHAR:CNUM 2
```

**Query Syntax**

```
SENS<ch>:CORRection:KIT:ECAL:CHARacterize:CNUMber?
```

**Return Type**

Numeric

**Default**

1

**SENSe:CORRection:KIT:ECAL:CHARacterize:CONNector:CATalog?**

*Applicable Models: All*

(Read-only) Returns a list of connector names that are valid for use with user-characterized ECAL modules. Use an item from the returned list to specify a connector for **SENS:CORR:KIT:ECAL:CHAR:CONN:PORT**.

Use only factory-defined connector types when you store a user characterization to VNA disk memory.

**Parameters**

None

**Examples**

```
```

*Example returned string: "APC 3.5 male, APC 3.5 female, Type N (50) female, Type N (50) male, APC 7, Type A (50), Type B"

**Return Type**

Comma-separated string

**Default**

Not Applicable


*Applicable Models: All*

(Read-Write) Specifies a connector type name for every ECAL module port used during the user characterization. Valid connector names are returned using **SENS:CORR:KIT:ECAL:CHAR:CONN:CAT?**.

This command refers to the ECAL ports by number instead of letter (1 = Port A, 2 = Port B, and so forth). The connector names should be set for the ports before sending **SENS:CORR:KIT:ECAL:CHAR:INIT**.

The following steps could be followed to ensure port connectors are specified correctly:

2. Specify a connector type using this command. If the <string> parameter was incorrectly entered, an error will be returned.

*Note: Use only factory-defined connector types when you store a user characterization to VNA disk memory.*

**Parameters**

- `<ch>`
  - Channel number being calibrated. If unspecified, value is set to 1

- `<n>`
  - ECAL test port number for which a connector type will be specified. Choose 1 to 2 for a 2-port ECAL module, 1 through 4 for a 4-port module.

- `<string>`
  - ECAL connector type and gender (if applicable).
When the User Characterization is to be stored in the ECal module, then the connector type is limited to a Factory-defined connector type. See the list.

When the User Characterization is to be stored in VNA disk memory, then the connector type can also be a User-defined connector type.

Examples

```
$SENSe:CORr:CKIT:ECAL:CHAR:CONN:PORT2 "APC 3.5 female"
```

Query Syntax

```
$SENSe<ch>:CORr:CKIT:ECAL:CHAR:CONN:PORT<n>[:SElECT]?
```

Return Type

String

Default

"No adapter"

---

**SENSe<ch>:CORr:CKIT:ECAL:CHAR:DESCrip[ion][:STEP]? <stepN>**

Applicable Models: All

(Read-only) Returns the connection description for the specified step of the ECal user characterization process. The user characterization process must have been initiated first using $SENSe:CORr:CKIT:ECAL:CHAR:INIT.

Parameters

<ch>  
Channel number being calibrated. If unspecified, value is set to 1

<stepN>  
Integer - User characterization step number for which a description will be returned. Use $SENSe:CORr:CKIT:ECAL:CHAR:STEP? to query the number of steps.

Examples

```
"Example return string:
Connect ECal Module Ports A and B to PNA Ports 1 and 2"
```

Return Type

String

Default

Not Applicable

---

**SENSe<ch>:CORr:CKIT:ECAL:CHAR:DESCrip[ion]:USER <string>**

Applicable Models: All

(Read-Write) Sets and reads the description of the person and/or company who is producing the ECal user characterization. This description is stored with the characterization.

Set this description before sending $SENSe:CORr:CKIT:ECAL:CHAR:INIT or the default (empty string) will be used.

Parameters

<ch>  
Channel number being calibrated. If unspecified, value is set to 1

<string>  
Descriptive text, limited to 19 characters maximum.
Examples

```
```

Query Syntax

```
```

Return Type

String

Default

" " (Empty String)

---

**SENSe<ch>:CORRection:CKIT:ECAL:CHARacterize:DESCription:VNA <string>**

Applicable Models: All
(Read-Write) Sets and reads a description of the Vector Network Analyzer used to perform the User Characterization. This description is stored with the user characterization.

Set this description before sending SENS:CORR:CKIT:ECAL:CHAR:INIT or the default (empty string) will be used.

Parameters

<ch>
Channel number being calibrated. If unspecified, value is set to 1

<string>
Descriptive text, limited to 14 characters maximum.

Examples

```
```

Query Syntax

```
```

Return Type

String

Default

" " (Empty String)

---


Applicable Models: All
(Read-Write) For each port of the ECal module that is going to be characterized, sets and reads the description of the adapters, cable, or fixture to be included in the user characterization. This command refers to the ECal ports by number instead of letter (1 = Port A, 2 = Port B, and so forth). This description is stored with the user characterization.

Set this description before sending SENS:CORR:CKIT:ECAL:CHAR:INIT or the default (empty string) will be used.

Parameters

<ch>
Channel number being calibrated. If unspecified, value is set to 1

<n>
ECal port number. Choose 1 to 2 for a 2-port ECal module, 1 to 4 for a 4-port module.

<string>
Descriptive text, limited to 24 characters maximum.

Examples

```
SENSe:CORR:CKIT:ECAL:CHAR:DESC:PORT1 "3.5 mm adapter, SN 00001"  
```

8157
SENSe<ch>:CORRection:CKIT:ECAL:CHARacterize:DMEMory:SAVE <charName>

(Application Models: All)
(Write-only) Completes an ECal user characterization by writing the characterization data to the VNA disk. To write the characterization data to the ECal module, use SENSe:CORR:CKIT:ECAL:CHAR:SAVE. A User Characterization can be saved to both VNA disk memory and ECal module memory.

Use this <charName> for performing future calibrations with this User Characterization.

Note: An ECal confidence check can NOT be performed remotely from User Characterizations that are stored on the VNA disk.

Parameters

<ch>
Channel number being calibrated. If unspecified, value is set to 1.

<charName>
String. User characterization name. Although there is no limit to the number of characters, only about 10 characters appear in the Cal Wizard dialog when selecting a user characterization for use.

Examples

SENSe:CORR:CKIT:ECAL:CHAR:DMEM:SAVE "DUT1 User Char"

Query Syntax

SENSe<ch>:CORRection:CKIT:ECAL:CHARacterize:ID <model, sn>

(Application Models: All)
(Read-Write) Selects the model and serial number of the ECal module to be characterized. This command does not Set the model and serial number of the ECal module.

Parameters

<ch>
Channel number being calibrated. If unspecified, value is set to 1

<model, sn>
Model and serial number of the ECal module to be characterized.

Examples

SENSe:CORR:CKIT:ECAL:CHAR:ID "N4433A,00001"

SENSe<ch>:CORRection:CKIT:ECAL:CHARacterize:ID?
SENSe<ch>:CORRection:CKIT:ECAL:CHARacterize:INITiate [bool]

Applicable Models: All

(Write-only) Initiates an ECal User Characterization. The specified channel number must be an S-parameter measurement channel. The channel must already be calibrated using the same, or greater number of VNA ports as the ECal module. Also, the calibrated VNA ports must begin with Port 1 and use sequential port numbers.

After this command is executed, subsequent commands can be used to query the number of measurement steps, issue the acquisition commands, query the connection description strings, and subsequently complete an Ecal User characterization.

Parameters

<ch> Channel number of a calibrated S-parameter channel. If unspecified, value is set to 1

[bool] Optional argument. If unspecified, value is set to 1.

ON (or 1) Check ECal memory to ensure that a new characterization with the channels current number of points will fit in the module memory. Select for User Characterizations to be stored in internal ECal memory.

OFF (or 0) Skip the check. Select for User Characterizations to be stored to VNA disk memory.

Examples

SENSe:CORR:CKIT:ECAL:CHAR:INIT
sense2:correction:kit:ecal:characterize:initiate off

Query Syntax

Not Applicable

Default

Not Applicable

SENSe<ch>:CORRection:CKIT:ECAL:CHARacterize:INSitu:ENABle?

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-only) Returns whether the device that was specified by SENS:CORR:CKIT:ECAL:CHAR:ID is a CalPod module, which is capable of being characterized as an in-situ device. Learn more.

Parameters

<ch> Channel number being calibrated. If unspecified, value is set to 1.

Examples

SENSe:CORR:CKIT:ECAL:CHAR:INSitu:ENABLE?

Return Type

Boolean

1 Device is a CalPod module

0 Device is NOT a CalPod module

Default

Not Applicable
**SENSe<ch>:CORRection:CKIT:ECAL:CHARacterize:INSitu[:STATe] <bool>**

Applicable Models: N522xB, N523xB, N524xB, M937xA, P937xA

(Read-Write) Sets or returns whether the device (CalPod module) that was specified by `SENS:CORR:CKIT:ECAL:CHAR:ID` will be characterized as an in situ device. Learn more.

**Parameters**

- `<ch>` Channel number being calibrated. If unspecified, value is set to 1.
- `<bool>` In situ state. Choose from:
  - ON (or 1) - Characterize the CalPod module as an in situ device.
  - OFF (or 0) - Do NOT characterize the CalPod module as an in situ device.

**Examples**

```
SENS:CORR:CKIT:ECAL:CHAR:INSitu 1
```

**Query Syntax**

```
SENSe<ch>:CORRection:CKIT:ECAL:CHARacterize:INSitu[:STATe]?
```

**Return Type**

Boolean

**Default**

ON or 1

---

**SENSe<ch>:CORRection:CKIT:ECAL:CHARacterize:SAVE**

Applicable Models: All

(Write-only) Completes an ECal user characterization by writing the characterization data to the ECal module memory. To write the characterization data to VNA disk memory, use `SENS:CORR:CKIT:ECAL:CHAR:DMEM:SAVE`. A User Characterization can be saved to both VNA disk memory and ECal module memory.

**Note:** This command is an overlapped command. When `*OPC` is issued with it, the OPC bit in the VNAs Standard Event Status Register is not set until this command has completed its operation. Learn more.

**Parameters**

- `<ch>` Channel number being calibrated. If unspecified, value is set to 1.

**Examples**

```
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**SENSe<ch>:CORRection:CKIT:ECAL:CHARacterize:STEPs?**

Applicable Models: All

(Read-only) Returns the number of steps required to measure the ECal module. Currently, only ONE is required.
### Parameters

<ch>

Channel number being calibrated. If unspecified, value is set to 1

### Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:CKIT:ECAL:CHAR:STEP?</td>
</tr>
</tbody>
</table>

### Return Type

Numeric

### Default

Not Applicable
MixerSegment

Configures Mixer Segments.

**SENSe:MIXer:SEGMen**

- ADD
- BWIDth
- CALCulate
- COUNt?
- DELet
  - ALL
- IF:FREQ:SIDeband

**INPut:FREQ:**

- FIXed
- MODE
- STARt
- STOP

**INPut:POWer**

**LO:FREQ:**

- FIXed
- ILTI
- MODE
- STARt
- STOP
Click on a red keyword to view the command details.

See Also

Example Programs
Learn about the Frequency Converter Application
Synchronizing the Analyzer and Controller
SCPI Command Tree

Scratch vs Applied Mixer Properties

Each mixer configuration has two sets of properties:

Scratch mixer contains the properties that have been set, but NOT YET applied. Send the SENSE<ch>:MIXer:APPLy command to copy these properties to the Applied mixer.

Applied mixer contains the properties that makeup the current mixer configuration.

Power settings are immediately applied to both the Scratch and Applied mixer.

A successful Calculate also perform an Apply.

Note: Queries always return the Applied mixer properties. Therefore, first send SENSE<ch>:MIXer:APPLy before querying new settings.

SENSe<ch>:MIXer:SEGMen<n>:ADD <value>

Applicable Models: All

(Write only) Adds the specified number of segments to the scratch mixer beginning at the last segment position. All segments are added with default settings.
Parameters

<ch>
  Channel number of the mixer measurement. If unspecified, value is set to 1.

<n>
  Position at which to add segments. Valid range is between 1 and the current segment count +1. Using count +1 adds segments to the end of the segment table. Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<value>
  Optional argument. Number of segments to add. If unspecified, 1 segment is added.

Examples

SENS:MIX:SEGM1:ADD 3 'Adds 3 segments beginning at position 1. For a preset state, this results in a total of 4 segments.'

Query Syntax

Not Applicable

Return Type

Numeric

Default

Not Applicable

SENSe<ch>:MIXer:SEGMen<n>:BWIDth <value>

Applicable Models: All
(Read/Write) Sets and returns the IF Bandwidth for the sweep segment.

Parameters

<ch>
  Channel number of the mixer measurement. If unspecified, value is set to 1.

<n>
  Existing segment for which IF Bandwidth is to be set.
  Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<value>
  IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the VNA model. (Click to see the lists.) If an invalid number is specified, the analyzer will round up to the closest valid number.
  This parameter supports MIN and MAX as arguments. Learn more.

Examples

SENS:MIX:SEGM1:BWID 1e3

Query Syntax

SENSe<ch>:MIXer:SEGMen<n>:BWIDth?

Send Apply before sending this query. Learn more.

Return Type

Numeric

Default

10 kHz
SENSe<ch>:MIXer:SEGMent<n>:CALCulate <char>

Applicable Models: All

(Write only) Calculates the Input, IF, or Output frequencies of the mixer setup and updates the channel settings.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Existing segment to calculate. Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<char> Mixer port to be calculated. Choose from:

<table>
<thead>
<tr>
<th>&lt;char&gt;</th>
<th>1st or only stage requires:</th>
<th>In addition, 2nd stage requires:</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPut</td>
<td>Output Start and Stop frequencies</td>
<td>IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>LO frequency</td>
<td>2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td>Output sideband (High or Low)</td>
<td>IF sideband (High or Low)</td>
</tr>
<tr>
<td>BOTH</td>
<td>NA</td>
<td>IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Both LO frequencies</td>
</tr>
<tr>
<td>OUTPut</td>
<td>Input Start and Stop frequencies</td>
<td>IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>LO frequency</td>
<td>2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td>Output sideband (High or Low)</td>
<td>IF sideband (High or Low)</td>
</tr>
<tr>
<td>LO_1</td>
<td>Input Start and Stop frequencies</td>
<td>IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>Output frequency</td>
<td>2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td>Output sideband (High or Low)</td>
<td>IF sideband (High or Low)</td>
</tr>
<tr>
<td>LO_2</td>
<td>NA</td>
<td>Input Start and stop frequencies</td>
</tr>
</tbody>
</table>
1st LO start and stop frequencies
Output frequency
IF sideband (High or Low)
Output sideband (High or Low)

Examples
SENS:MIX:SEGM2:CALC Output

Query Syntax
Not Applicable

Default
Not Applicable

**SENSe<ch>:MIXer:SEGment:COUNt?**

Applicable Models: All
(Read-only) Returns the number of segments on the Applied mixer.

Parameters

<ch> 
Channel number of the mixer measurement. If unspecified, value is set to 1.

Examples
SENS:MIX:SEGM:COUN?

Return Type
Numeric

Default
Not Applicable

**SENSe<ch>:MIXer:SEGment<n>:DELeTe <value>**

Applicable Models: All
(Write only) Removes the specified number of segments from the scratch mixer starting at the index position.

Parameters

<ch> 
Channel number of the mixer measurement. If unspecified, value is set to 1.

<n> 
Position at which to start removing segments. Valid index range is between 1 and the current segment count. Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.
Optional argument. Number of segments to remove. If unspecified, 1 segment is removed.

### Examples

```
SENS:MIX:SEG1:DEL 5 'Removes 5 segments beginning at the first position.
```

### Default

Not Applicable

---

**SENSe<ch>:MIxer:SEGMent:DELete:ALL**

Applicable Models: All

(Write only) Removes all segments from the scratch mixer.

#### Parameters

- `<ch>`: Channel number of the mixer measurement. If unspecified, value is set to 1.

#### Examples

```
SENS:MIX:SEG:DEL:ALL
```

#### Query Syntax

Not Applicable

#### Default

Not Applicable

---

**SENSe<ch>:MIxer:SEGMent<n>:DWELI**

Applicable Models: All

(Read-Write) Sets or returns the Input sweep mode of the segment.

#### Parameters

- `<ch>`: Channel number of the mixer measurement. If unspecified, value is set to 1.
- `<n>`: Existing segment for which Input frequency mode is to be set.

Use `SENS:MIX:SEG:MOUN?` to read the current count in the Applied Mixer.

#### Examples

```
SENS:MIX:SEG2:DWELI
```

#### Query Syntax

Send Apply before sending this query. Learn more.

#### Return Type

Character
SENSe<ch>:MIXer:SEGMen<n>:IF:FREQuency:SIDeband <char>

Applicable Models: All

(Read-Write) When two LO stages are used, sets or returns whether to select the sum or difference for the IF1 product. (Input + or - LO1 = IF1)

This setting corresponds to the buttons on LO1 on the Mixer Setup dialog.

This setting is ignored when ONE LO stage is selected.

Also set SENS:MIX:OUTP:FREQ:SID to LOW or HIGH to set the output frequency of the mixer.

Parameters

<ch> Channel number of the mixer measurement. If unspecified, value is set to 1.

<n> Existing segment for which IF Sideband is to be set. Use SENS:MIX:SEG:COUNT? to read the current count in the Applied Mixer.

<char> Sideband value. Choose from

LOW - Difference (-)
HIGH - Sum (+)

Examples

SENSe:MIX:SEG2:IF:FREQ:SID LOW

Query Syntax

SENSe<ch>:MIXer:SEGMen<n>:IF:FREQuency:SIDeband?

Send Apply before sending this query. Learn more.

Return Type

Character

Default

LOW

SENSe<ch>:MIXer:SEGMen<n>:INPut:FREQuency:FIXed <value>

Applicable Models: All

(Read-Write) Sets or returns the Input fixed frequency of the segment. Also, set SENS:MIX:SEG:INP:FREQ:MODE FIXED.

Parameters

<ch> Channel number of the mixer measurement. If unspecified, value is set to 1.
### Input Fixed Frequency

Existing segment for which input fixed frequency is to be set.

Use `SENS:MIX:SEGM:COUN?` to read the current count in the Applied Mixer.

**Examples**

```
SENSe2:MIXer:SEGMent2:INPut:FREQ:FIX 1e9
```

**Query Syntax**

```
SENSe<ch>:MIXer:SEGMent<n>:INPut:FREQ:FIXed?
```

Send Apply before sending this query. Learn more.

**Return Type**

Numeric

**Default**

Start frequency of the VNA.

---

### Input Frequency Mode

`SENSe<ch>:MIXer:SEGMent<n>:INPut:FREQuency:MODE <char>`

Applicable Models: All

(Read-Write) Sets or returns the Input sweep mode of the segment.

**Parameters**

- `<ch>`: Channel number of the mixer measurement. If unspecified, value is set to 1.
- `<n>`: Existing segment for which Input frequency mode is to be set.
  
  Use `SENS:MIX:SEGM:COUN?` to read the current count in the Applied Mixer.
- `<char>`: Input sweep mode. Choose either FIXED or SWEPT

**Examples**

```
SENS:MIX:SEGm2:INP:FREQ:MODE FIXED
```

**Query Syntax**

```
SENSe<ch>:MIXer:SEGMent<n>:INPut:FREQuency:MODE?
```

Send Apply before sending this query. Learn more.

**Return Type**

Character

**Default**

Swept

---

### Input Start Frequency

`SENSe<ch>:MIXer:SEGMent<n>:INPut:FREQuency:STARt <value>`

Applicable Models: All

(Read-Write) Sets or returns the Input start frequency value of the segment. Also, set `SENS:MIX:SEGM:FREQ:MODE SWEPT`.

---
Parameters

<ch>  
Channel number of the mixer measurement. If unspecified, value is set to 1.

<n>  
Existing segment for which Input start frequency is to be set.

Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<value>  
Input Start frequency. Choose a value between the start and stop frequency of the VNA.

Examples

SENSe2: MIXer: SEGment2: INPut: FREQ: START 1000000000

Query Syntax

SENSe<ch>: MIXer: SEGment<n>: INPut: FREQuency: STARt?

Send Apply before sending this query. Learn more.

Return Type

Numeric

Default

Start frequency of the VNA.

SENSe<ch>: MIXer: SEGment<n>: INPut: POWer <value>

Applicable Models: All

(Read-Write) Sets or returns the Input stop frequency value of the segment.

Parameters

<ch>
Channel number of the mixer measurement. If unspecified, value is set to 1.

<n>
Existing segment for which Input stop frequency is to be set.

Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<value>  
Input Stop frequency. Choose a value between the start and stop frequency of the VNA.

Examples

SENSe2: MIXer: SEGment2: INPut: FREQ: STOP 1000000000

Query Syntax

SENSe<ch>: MIXer: SEGment<n>: INPut: FREQuency: STOP?

Send Apply before sending this query. Learn more.

Return Type

Numeric

Default

Stop frequency of the VNA.
Applicable Models: All
(Read-Write) Sets or returns the Input power value of the segment.

Parameters

<ch>
Channel number of the mixer measurement. If unspecified, value is set to 1.

<n>
Existing segment for which Input power is to be set.
Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<value>
Input power level in dBm. Choose a value within the power range of the VNA.

Examples
SENSe2:MIXer:SEGMent2:INPut:POWer 0

Query Syntax
SENSe<ch>:MIXer:SEGMent<n>:INPut:POWer?
Send Apply before sending this query. Learn more.

Return Type
Numeric

Default
-15 dBm

SENSe<ch>:MIXer:SEGMent<n>:LO<x>:FREQuency:FIXed <value>

Applicable Models: All
(Read-Write) Sets or returns the LO1 and LO2 fixed frequency value of the segment.

Parameters

<ch>
Channel number of the mixer measurement. If unspecified, value is set to 1.

<n>
Existing segment for which the LO1 and LO2 fixed frequency value is to be set.
Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<x>
LO stage number for which the fixed frequency is to be set. Choose 1 or 2.
Set number of stages for the mixer using SENS<ch>:MIXer:STAGe.

<value>
The LO1 and LO2 fixed frequency value in Hz. Choose a value within the frequency range of the source.

Examples
SENSe2:MIXer:SEGMent2:LO1:FREQ:FIX 1e9

Query Syntax
SENSe<ch>:MIXer:SEGMent<n>:LO<x>:FREQuency:FIXed?
Send Apply before sending this query. Learn more.

Return Type
Numeric
SENSe<ch>:MIXer:SEGment<n>:LO<x>:FREQuency:ILT<bool>

Applicable Models: All
(Read-Write) Specifies whether to use the Input frequency that is greater than the LO or less than the LO. To learn more, see the mixer setup dialog box help.

Parameters

<ch> Channel number of the mixer measurement. If unspecified, value is set to 1.

<n> Existing segment for which the LO1 and LO2 fixed frequency value is to be set. Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<x> LO stage number for which ILT is to be set. Choose 1 or 2.

<bool> ON (1) - Use the Input that is Greater than the specified LO. OFF (0) - Use the Input that is Less than the specified LO.

Examples

SENSe2:MIXer:SEGment2:LO1:FREQ:ILT 1

Query Syntax

SENSe<ch>:MIXer:SEGment<n>:LO<x>:FREQuency:ILT?

Send Apply before sending this query. Learn more.

Return Type

Boolean

Default

OFF

SENSe<ch>:MIXer:SEGment<n>:LO<x>:FREQuency:MODE <char>

Applicable Models: All
(Read-Write) Sets or returns the LO1 or LO2 sweep mode of the segment.

Parameters

<ch> Channel number of the mixer measurement. If unspecified, value is set to 1.

<n> Existing segment for which LO frequency mode is to be set. Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<x> LO stage number for which ILT is to be set. Choose 1 or 2.
Set number of stages for the mixer using `SENS<ch>:MIXer:STAGe`.

**Examples**

```
SENS:MIX:SEGM2:LO1:FREQ:MODE FIXED
```

**Query Syntax**

```
SE Nes<ch>:MIXer:SEGMent<n>:LO<n>:FREQuency:MODE?
```

Send **Apply** before sending this query. Learn more.

**Return Type**

Character

**Default**

Fixed

---

**SENS<ch>:MIXer:SEGMent<n>:LO<x>:FREQuency:STARt <value>**

**Applicable Models:** All

(Read-Write) Sets or returns the LO start frequency value of the segment.

**Parameters**

- `<ch>`
  - Channel number of the mixer measurement. If unspecified, value is set to 1.

- `<n>`
  - Existing segment for which LO start frequency is to be set.
    - Use `SENS:MIX:SEGM:COUN?` to read the current count in the **Applied Mixer**.

- `<x>`
  - LO stage number. Choose 1 or 2.
    - Set number of stages for the mixer using `SENS<ch>:MIXer:STAGe`.

- `<value>`
  - LO Start frequency. Choose a value between the start and stop frequency of the VNA.

**Examples**

```
SENS2:MIXer:SEGMent2:LO1:FREQ:STARt 1000000000
```

**Query Syntax**

```
SENS<ch>:MIXer:SEGMent<n>:LO<x>:FREQuency:STARt?
```

Send **Apply** before sending this query. Learn more.

**Return Type**

Numeric

**Default**

Start frequency of the VNA.

---

**SENS<ch>:MIXer:SEGMent<n>:LO<x>:FREQuency:STOP <value>**

**Applicable Models:** All

(Read-Write) Sets or returns the LO stop frequency value of the segment.

---
Parameters

<ch> Channel number of the mixer measurement. If unspecified, value is set to 1.

<n> Existing segment for which LO stop frequency is to be set. Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<x> LO stage number. Choose 1 or 2. Set number of stages for the mixer using SENSe<ch>:MIXer:STAGE.

<value> LO Stop frequency. Choose a value between the start and stop frequency of the VNA.

Examples
SENSe2:MIXer:SEGMen2:LO<x>:FREQ:STOP 1000000000

Query Syntax
SENSe<ch>:MIXer:SEGMen>n:LO<x>:FREQuency:STOP?
Send Apply before sending this query. Learn more.

Return Type Numeric

Default Stop frequency of the VNA.

SENSe<ch>:MIXer:SEGMen<n>:LO<x>:POWer <value>

Applicable Models: All
(Read-Write) Sets or returns the LO power of the segment.

Parameters

<ch> Channel number of the mixer measurement. If unspecified, value is set to 1.

<n> Existing segment for which LO power is to be set. Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<x> LO stage number. Choose 1 or 2. Set number of stages for the mixer using SENSe<ch>:MIXer:STAGe.

<value> LO power level in dBm. Choose a value within the power range of the VNA.

Examples
SENSe2:MIXer:SEGMen2:INPut:POWer 0

Query Syntax
SENSe<ch>:MIXer:SEGMen<n>:INPut:POWer?
Send Apply before sending this query. Learn more.

Return Type Numeric
**Default**

-10 dBm

**SENSe<ch>:MIXer:SEGMen<n>:OUTPut:FREQuency:FIXed <value>**

Applicable Models: All

(Read-Write) Sets or returns the Output fixed frequency of the segment. Also, set `SENS:MIX:SEG:M:INP:FREQ:MODE FIXED`.

**Parameters**

- `<ch>`: Channel number of the mixer measurement. If unspecified, value is set to 1.
- `<n>`: Existing segment for which Output fixed frequency is to be set. Use `SENS:MIX:SEG:M:COUN?` to read the current count in the Applied Mixer.
- `<value>`: Output fixed frequency. Choose a value between the start and stop frequency of the VNA.

**Examples**

```bash
SENSe2:MIXer:SEGMen2:OUTP:FREQ:FIX 1e9
```

**Query Syntax**


Send Apply before sending this query. Learn more.

**Return Type**

Numeric

**Default**

Start frequency of the VNA.

---

**SENSe<ch>:MIXer:SEGMen<n>:OUTPut:FREQuency:MODE <char>**

Applicable Models: All

(Read-Write) Sets or returns the Output sweep mode of the segment.

**Parameters**

- `<ch>`: Channel number of the mixer measurement. If unspecified, value is set to 1.
- `<n>`: Existing segment for which Output frequency mode is to be set. Use `SENS:MIX:SEG:M:COUN?` to read the current count in the Applied Mixer.
- `<char>`: Output sweep mode. Choose either FIXED or SWEPT

**Examples**

```bash
```

8175
SENSe<ch>:MIXer:SEGMent<n>:OUTPut:FREQuency:SIDeband <char>

Applicable Models: All

(Read-Write) When two LO stages are used, sets or returns whether to select the sum or difference for the Output product. Use for both 1 or 2 stage mixers.

This setting corresponds to the \( \pm \) buttons on the Output on the Mixer Frequency dialog.

Also set SENS:MIX:IF:FREQ:SID to LOW or HIGH to determine the IF frequency for 2-stage mixers.

Parameters

<ch> Channel number of the mixer measurement. If unspecified, value is set to 1.

<n> Existing segment for which Output Sideband is to be set. Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<char> Sideband value. Choose from

LOW - Difference (-)

HIGH - Sum (+)

Examples

\[ \text{SENS:MIX:SEG2:OUTP:FREQ:SID LOW} \]

Query Syntax

SENSe<ch>:MIXer:SEGMent<n>:OUTPut:FREQuency:SIDeband?

Send Apply before sending this query. Learn more.

Return Type

Character

Default

LOW

SENSe<ch>:MIXer:SEGMent<n>:OUTPut:FREQuency:STARt <value>

Applicable Models: All

(Read-Write) Sets or returns the Output start frequency value of the segment. Also, set SENS:MIX:SEGM:FREQ:MODE SWEPT.
Parameters

<ch> Channel number of the mixer measurement. If unspecified, value is set to 1.

<n> Existing segment for which Output start frequency is to be set.

Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<value> Output Start frequency. Choose a value between the start and stop frequency of the VNA.

Examples

SENSe<ch>:MIXer:SEGMent<n>:OUTPut:FREQ:STARt 1000000000

Query Syntax

SENSe<ch>:MIXer:SEGMent<n>:OUTPut:FREQ:STARt?

Send Apply before sending this query. Learn more.

Return Type

Numeric

Default

Start frequency of the VNA.

SENSe<ch>:MIXer:SEGMent<n>:OUTPut:FREQ:STOP <value>

Applicable Models: All

(Read-Write) Sets or returns the Output stop frequency value of the segment.

Parameters

<ch> Channel number of the mixer measurement. If unspecified, value is set to 1.

<n> Existing segment for which Output stop frequency is to be set.

Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<value> Output Stop frequency. Choose a value between the start and stop frequency of the VNA.

Examples

SENSe2:MIXer:SEGMent2:OUTPut:FREQ:STOP 1000000000

Query Syntax

SENSe<ch>:MIXer:SEGMent<n>:OUTPut:FREQ:STOP?

Send Apply before sending this query. Learn more.

Return Type

Numeric

Default

Stop frequency of the VNA.

SENSe<ch>:MIXer:SEGMent<n>:OUTPut:POWer <value>
Applicable Models: All
(Read-Write) Sets or returns the Output power value of the segment.

Parameters

**<ch>**
Channel number of the mixer measurement. If unspecified, value is set to 1.

**<n>**
Existing segment for which Output power is to be set.
Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

**<value>**
Output power level in dBm. Choose a value within the power range of the VNA.

**Examples**
SENSe2:MIXer:SEGMen2:OUTPut:POWer 0

**Query Syntax**
SENSe<ch>:MIXer:SEGMent<n>:OUTPut:POWer?
Send Apply before sending this query. Learn more.

**Return Type**
Numeric

**Default**
-10 dBm

---

SENSe<ch>:MIXer:SEGMent<n>:POINts <value>

Applicable Models: All
(Read/Write) Sets and returns the number of data points for the sweep segment.

Parameters

**<ch>**
Channel number of the mixer measurement. If unspecified, value is set to 1.

**<n>**
Existing segment for which number of points is to be set.
Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

**<value>**
Number of data points. Choose any number between 1 and the VNA maximum number of points.
This parameter supports MIN and MAX as arguments. Learn more.

**Examples**
SENSe:MIX1:POIN 3

**Query Syntax**
SENSe<ch>:MIXer:SEGMent<n>:POINts?
Send Apply before sending this query. Learn more.

**Return Type**
Numeric
SENS<ch>:MIXer:SEGMen<n>:STATe <bool>

Applicable Models: All
(Read/Write) Sets and returns the ON/OFF state for the segment.

Parameters

<ch> Channel number of the mixer measurement. If unspecified, value is set to 1.

<n> Existing segment to be set ON or OFF.
Use SENS:MIX:SEGM:COUN? to read the current count in the Applied Mixer.

<bool> Segment state. Choose from:
ON (or 1) - Turns the segment ON.
OFF (or 0) - Turns the segment OFF.

Examples

SENS:MIX:SEG1:STATe 1

Query Syntax

SENSe<ch>:MIXer:SEGMen<n>:STATe?

Send Apply before sending this query. Learn more.

Return Type

Boolean

Default

ON
Role

Sense:Role Commands

Controls the path configuration settings.

SENSe:ROLE:
| CATalog?
| DEVice

Click on a keyword to view the command details.

See Also

Example Programs
Synchronizing the Analyzer and Controller
SCPI Command Tree

SENSe<ch>:ROLE:CATalog?

Applicable Models: N522xB, N523xB, N524xB
(Read-only) This command replaces SENs:MIX:ROLE:CATalog?
Returns the roles for which sources can be used for the channel.

Parameters

<ch> Channel number of the measurement. If unspecified, value is set to 1.

Examples
SENS:ROLE:CAT?

Return Type
Comma-separated list of double-quoted strings.

Default
Not Applicable

SENSe<ch>:ROLE:DEVice <role>,<source>

Applicable Models: N522xB, N523xB, N524xB
(Read-Write) This command replaces SENs:MIX:ROLE:DEVice
Set and return the source to be used in the specified role. For example, use this command to set a source name to be used as the RF2 tone for a Swept IMD channel.

Parameters

<ch> Channel number of the measurement. If unspecified, value is set to 1.
(String) Role of the source. Not context-sensitive. Use SENSe:ROLE:CATalog? to read the valid roles for the channel.

(String) Source name to be used in the specified role. Use SYSTem:CONFigure:EDEVice:CAT? to read a list of configured sources.

Example:
SENSe2:ROLE:DEV "RF2", "MyEsq"

Query Syntax
SENSe<ch>:ROLE:DEVice? <role>

Return Type
String

Default
Not Applicable
## SweepPulse

Configures the channel to make pulse measurements using the Integrated Pulse Application.

### SENSE:SWEep:PULSe

<table>
<thead>
<tr>
<th>CWTime</th>
<th>DETectmode[:AUTO]</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRIVE[:AUTO]</td>
<td>IFBW[:AUTO]</td>
</tr>
<tr>
<td>IFGain[:AUTO]</td>
<td>PRF[:AUTO]</td>
</tr>
<tr>
<td>MODE</td>
<td>PRIMary:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>PROFile</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>SWGate</td>
<td>TIMing[:AUTO]</td>
</tr>
<tr>
<td></td>
<td>WIDeband[:STATE]</td>
</tr>
</tbody>
</table>

Click on a keyword to view the command details.

### See Also

- Example Program using these commands.
- SENS:PULSe commands used to configure the pulse generators.
- PNA-X IF Path Block diagram
- SENS:IF configuration commands
- Synchronizing the Analyzer and Controller
- SCPI Command Tree
SENSe<ch>:SWEep:PULSe:CWTime[:AUTO] <bool>

Applicable Models: VNAs with Pulsed RF Measurement Option

(Read-Write) This command replaces SENSe:SWEep:PULSe:IFBW[:AUTO]. Sets and returns the state of automatic CW sweep time (used in Pulse Profile mode). This setting is labeled Autoselect Profile Sweep Time on the UI.

Parameters

<ch>          Channel number of the pulse measurement. If unspecified, value is set to 1.
<bool>        Choose from:

OFF (or O) - The Sweep Time is not changed automatically.

ON (or 1) - In Pulse Profile mode, adjusts the default X-axis start time to zero and the stop time to double the Pulse Width. This allows you to see one complete pulse.

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:SWEep:PULSe:CWT 1</td>
<td></td>
</tr>
<tr>
<td>sense2:sweep:pulse:cwtime:auto off</td>
<td></td>
</tr>
</tbody>
</table>

Query Syntax

SENSe<ch>:SWEep:PULSe:CWTime[:AUTO]?

Return Type

Boolean

Default

ON

SENSe<ch>:SWEep:PULSe:DETectmode[:AUTO] <bool>

Applicable Models: VNAs with Pulsed RF Measurement Option

(Read-Write) Set pulse mode automatically or manually (Narrowband or Wideband) for the channel. This setting is labeled Autoselect Pulse Detection Method on the UI.

Parameters

<ch>          Channel number of the pulse measurement. If unspecified, value is set to 1.
<bool>        Choose from:

OFF (or O) - Manually set the pulse mode. Use SENSe:SWEep:PULSe:WIDE to set the pulse mode.

ON (or 1) - Automatically set the pulse mode.

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:SWEep:PULSe:DET 1</td>
<td></td>
</tr>
<tr>
<td>sense2:sweep:pulse:detectmode:auto off</td>
<td></td>
</tr>
</tbody>
</table>
SENSe<ch>:SWEep:PULSe:DETectmode[:AUTO]?

Return Type: Boolean
Default: ON

SENSe<ch>:SWEep:PULSe:DRIVe[:AUTO] <bool>

Applicable Models: VNAs with Pulsed RF Measurement Option (Read-Write) In Narrowband pulse mode, set the drive for the source modulation automatically or manually.

Parameters
- `<ch>`: Channel number of the pulse measurement. If unspecified, value is set to 1.
- `<bool>`: Choose from:
  - OFF (or O) - Manually set source modulation.
  - ON (or 1) - Automatically set pulse gen 1 as the modulation source and pulse gen 2 as the gate source for all gates.

Examples
SENSe:SWE:PULS:DRIV 1
sense2:sweep:pulse:drive:auto off

SENSe<ch>:SWEep:PULSe:IFBW[:AUTO] <bool> - Superseded

Applicable Models: VNAs with Pulsed RF Measurement Option (Except M9485A) (Read-Write) This command is replaced with SENS:SWEep:PULSe:CWTime[:AUTO]. In Wideband pulse mode, set the IF bandwidth automatically or manually.

Parameters
- `<ch>`: Channel number of the pulse measurement. If unspecified, value is set to 1.
- `<bool>`: Choose from:
  - OFF (or O) - Manually set the IFBW for the measurement.
  - ON (or 1) - Automatically set the IFBW for the measurement.
Examples
SENSe:SWEep:PULSe:IFBW 1
SENSe2:SWEep:PuLPSe:IFBW:auto off

Query Syntax
SENSe<ch>:SWEep:PULSe:IFBW[:AUTO]?

Return Type
Boolean

Default
ON

SENSe<ch>:SWEep:PULSe:IFGain[:AUTO] <bool>

Applicable Models: VNAs with Pulsed RF Measurement Option (Except M9485A, M980xA, P50xxA)

(Read-Write) In Narrowband pulse mode, set the IF Gain automatically or manually. This setting is labeled Autoselect IF Path Gain and Loss on the UI.

Parameters

<ch>
Channel number of the pulse measurement. If unspecified, value is set to 1.

<bool>
Choose from:

OFF (or O) - Manually set the IF Gain for the measurement.
ON (or 1) - Automatically set the IF Gain for the measurement.

Examples
SENSe:SWEep:PULSe:IFG 1
SENSe2:SWEep:PuLPSe:IFG:auto off

Query Syntax
SENSe<ch>:SWEep:PULSe:IFGain[:AUTO]?

Return Type
Boolean

Default
ON

SENSe<ch>:SWEep:PULSe:MODE <char>

Applicable Models: VNAs with Pulsed RF Measurement Option

(Read-Write) Sets and returns the pulse measurement state for the channel.

Parameters

<char>
Channel number of the pulse measurement. If unspecified, value is set to 1.

<char>
Choose from:

OFF - Turn OFF pulse measurements.
STD - Turn ON standard pulse measurements.
PROFILE - Turn ON pulse profile measurements.

Examples

```
SENS:SWE:PULS:MODE PROFILE
sense2:sweep:pulse:mode off
```

Query Syntax
SENSe<ch>:SWEep:PULSe:MODE?

Return Type
Character

Default
OFF

-----

**SENSe<ch>:SWEep:PULSe:PRF[:AUTO] <bool>**

Applicable Models: VNAs with Pulsed RF Measurement Option (Except M9485A, M980xA, P50xxA)

*(Read-Write)* In Narrowband pulse mode, choose to set the Pulse Repetition Frequency automatically or manually. This is labeled "Optimize Pulse Frequency" on the user-interface. To make changes manually, use SENS:SWE:PULS:PRIM:FREQ or SENS:SWE:PULS:PRIM:PER.

Parameters

- `<ch>` Channel number of the pulse measurement. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - OFF (or O) - Manually set the PRF for the measurement.
  - ON (or 1) - Automatically set the PRF for the measurement.

Examples

```
SENS:SWE:PULS:PRF 1
sense2:sweep:pulse:prf:auto off
```

Query Syntax
SENSe<ch>:SWEep:PULSe:PRF[:AUTO]?

Return Type
Boolean

Default
ON

-----

**SENSe<ch>:SWEep:PULSe:PRIMary:CLOCk <string>**

Applicable Models: VNAs with Pulsed RF Measurement Option

*(Read-Write)* Sets and returns the Primary Clock setting and is controlled by the internal or external pulse generator which is the primary pulse clock. [Learn more.](#)
Note: If there is a defined pulse generator in the external devices list, its name will be displayed in the Pulse Setup dialog Primary Clock control and it will also be an accepted choice for this command.

Parameters

`<ch>` Channel number of the pulse measurement. If unspecified, value is set to 1.

`<string>` Primary clock (Internal or External).

Examples

```
SENS:SWE:PULS:PRIM:CLOC "Internal"
sense2:sweep:pulse:primary:clock "External"
```

Query Syntax

```
SENi<ch>:SWEep:PULSe:PRIMary:CLOCk?
```

Return Type

String

Default

N/A

```
SENS<ch>:SWEep:PULSe:PRIMary:FREQuency <value>
```

Applicable Models: VNAs with Pulsed RF Measurement Option

(Read-Write) Sets and returns the primary pulse measurement frequency. The frequency is 1/period, so this value can also be set using SENS:SWE:PULS:PRIM:PERiod.

Note: On the Pulse Setup dialog, this command is a 'Basic setting, which is intended to be used with the 'Auto' selections set to ON.

Parameters

`<ch>` Channel number of the pulse measurement. If unspecified, value is set to 1.

`<value>` Primary frequency.

Examples

```
SENS:SWE:PULS:PRIM:FREQ 1e9
sense2:sweep:pulse:primary:frequency 1e6
```

Query Syntax

```
SENi<ch>:SWEep:PULSe:PRIMary:FREQuency
```

Return Type

Value

Default

1 kHz

```
SENS<ch>:SWEep:PULSe:PRIMary:PERiod <value>
```

Applicable Models: VNAs with Pulsed RF Measurement Option

(Read-Write) Sets and returns the primary pulse period. The period is 1/frequency, so this value can also be set using SENS:SWE:PULS:PRIM:FREQ.
Note: On the Pulse Setup dialog, this command is a 'Basic' setting, which is intended to be used with the 'Auto' selections set to ON.

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Channel number of the pulse measurement. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;value&gt;</td>
<td>Primary period in seconds.</td>
</tr>
</tbody>
</table>

### Examples

```
SENS:SWE:PULS:PRIM:PER 1e-6
sense2:sweep:pulse:primary:period 1e-3
```

### Query Syntax

```
SENSe<ch>:SWEep:PULSe:PRIMary:PERiod?
```

### Return Type

**Value**

### Default

1 msec

---

**SENSe<ch>:SWEep:PULSe:PRIMary:WIDth <value>**

**Applicable Models:** VNAs with Pulsed RF Measurement Option

(Read-Write) Sets and returns the primary pulse width.

Note: On the Pulse Setup dialog, this command is a 'Basic' setting, which is intended to be used with the 'Auto' selections set to ON.

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Channel number of the pulse measurement. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;value&gt;</td>
<td>Primary pulse width in seconds.</td>
</tr>
</tbody>
</table>

### Examples

```
SENS:SWE:PULS:PRIM:WIDTh
sense2:sweep:pulse:primary:width
```

### Query Syntax

```
SENSe<ch>:SWEep:PULSe:PRIMary:WIDTh?
```

### Return Type

**Value**

### Default

100 microseconds

---

**SENSe<ch>:SWEep:PULSe:PROFile:STARt <value>**

**Applicable Models:** VNAs with Pulsed RF Measurement Option

(Read-Write) Sets and returns the start time of the pulse. Pulse profile measurements provide a time domain (CW frequency) view of the pulse envelope. Profiling is performed using a measurement technique that "walks" a narrow receiver "snapshot"
across the width of the pulse. This is analogous to using a camera to take many small snapshots of a wide image, then piecing them together to form a single, panoramic view.

Parameters

<ch> Channel number of the pulse measurement. If unspecified, value is set to 1.

<value> Start time in seconds.

Note: The start value cannot be negative.

Examples

SENS:SWE:PULS:PROF:STAR 1e-6
sense2:sweep:pulse:profile:start 1e-3

Query Syntax
SENSe<ch>:SWEep:PULSe:PROFile:STARt?

Return Type Value

Default 0

---

SENS<ch>:SWEep:PULSe:PROFile:STOP <value>

Applicable Models: VNAs with Pulsed RF Measurement Option

(Read-Write) Sets and returns the stop time of the pulse. Pulse profile measurements provide a time domain (CW frequency) view of the pulse envelope. Profiling is performed using a measurement technique that "walks" a narrow receiver "snapshot" across the width of the pulse. This is analogous to using a camera to take many small snapshots of a wide image, then piecing them together to form a single, panoramic view.

Parameters

<ch> Channel number of the pulse measurement. If unspecified, value is set to 1.

<value> Stop time in seconds.

Note: The stop value cannot be negative.

Examples

SENS:SWE:PULS:PROF:STOP 1e-6
sense2:sweep:pulse:profile:stop 1e-3

Query Syntax SENSe<ch>:SWEep:PULSe:PROFile:STOP?

Return Type Value

Default N/A

---

SENS<ch>:SWEep:PULSe:SHAPe <char>

Applicable Models: VNAs with Pulsed RF Measurement Option
Sets and returns the pulse shape type.

**Parameters**

- `<ch>`
  Channel number of the pulse measurement. If unspecified, value is set to 1.

- `<char>`
  Choose from:

  **Normal** - Normal pulse shape

  **Fast** - Fast rise time with low on/off ratio. This is available only for M980xA and P50xxA with S9x025A/B (M980xA and P50xxA with S9x024B do not support FAST mode.)

**Examples**

```
SENS:SWE:PULS:SHAP "FAST"
sense2:sweep:pulse:shape "Fast"
```

**Query Syntax**

`SENS<ch>:SWEep:PULSe:SHAPe?`

**Return Type**

Character

**Default**

Normal

---

`SENS<ch>:SWEep:PULSe:SHAPe:CATalog?`

Applicable Models: VNAs with Pulsed RF Measurement Option

(Read-Write) Returns the list of available pulse shape type.

**Parameters**

- `<ch>`
  Channel number of the pulse measurement. If unspecified, value is set to 1.

**Examples**

```
SENS:SWE:PULS:SHAP:CAT?
sense2:sweep:pulse:shape:catalog?
```

**Query Syntax**

`SENS<ch>:SWEep:PULSe:SHAPe:CATalog?`

**Return Type**

Character

**Default**

Not applicable

---

`SENS<ch>:SWEep:PULSe:SWGate[:STATe] <bool>`

Applicable Models: VNAs with Pulsed RF Measurement Option (Except M9485A, M980xA, P50xA)
When set to OFF, the improved software gating sensitivity is turned OFF and all data outside the measurement band is zeroed. This setting is used for troubleshooting purposes.

Parameters

<ch> Channel number of the pulse measurement. If unspecified, value is set to 1.

<bool> Choose from:

OFF (or O) - Turn OFF software gating.

ON (or 1) - Turn ON software gating.

Examples

SENS:SWE:PULS:SWG 0
sense2:sweep:pulse:swgate:state on

Query Syntax

SENSe<ch>:SWEep:PULSe:SWGate[:STATe]?

Return Type

Boolean

Default

ON

SENS<ch>:SWEep:PULSe:TIMing[:AUTO] <bool>

Applicable Models: VNAs with Pulsed RF Measurement Option

(Read-Write) In Narrowband pulse mode, choose to set the delay and width automatically or manually. This setting is labeled Autoselect Width and Delay on the UI.

Parameters

<ch> Channel number of the pulse measurement. If unspecified, value is set to 1.

<bool> Choose from:

OFF (or O) - Manually set the delay and width for the measurement.

ON (or 1) - Automatically set the delay and width for the measurement.

Examples

SENS:SWE:PULS:TIM 1
sense2:sweep:pulse:timing:auto off

Query Syntax

SENSe<ch>:SWEep:PULSe:TIMing[:AUTO]?

Return Type

Boolean

Default

ON

SENS<ch>:SWEep:PULSe:WIDeband[:STATe] <bool>
Applicable Models: VNAs with Pulsed RF Measurement Option (Except M9485A, M980xA, P50xxA)

*(Read-Write)* Set and read the pulse mode detection method.

**Parameters**

- `<ch>`: Channel number of the pulse measurement. If unspecified, value is set to 1.
- `<bool>`: Choose from:
  - OFF (or 0) - Narrowband mode.
  - ON (or 1) - Wideband mode

**Examples**

```
SENS:SWE:PULS:WID 1
sense2:sweep:pulse:wideband:state off
```

**Query Syntax**

`SENSe<ch>:SWEep:PULSe:WIDeband[:STATe]?`

**Return Type**

Boolean

**Default**

Based on pulse width
TDRCorrection
SENSe:CORRection:TDR Commands

These commands control loss compensation, fixture compensation after an ECal, auto port extension, and reference impedance value.

<table>
<thead>
<tr>
<th>SENSe:CORRection:TDR COLLECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLComp</td>
</tr>
<tr>
<td>LOAD</td>
</tr>
<tr>
<td>OPEN</td>
</tr>
<tr>
<td>SAVE</td>
</tr>
<tr>
<td>THRU</td>
</tr>
<tr>
<td>ECAL</td>
</tr>
<tr>
<td>FCOMp</td>
</tr>
<tr>
<td>IMMmediate</td>
</tr>
<tr>
<td>IMMmediate</td>
</tr>
</tbody>
</table>

DCONstant
EXTension
  | AUTO                          |
  | IMMmediate                    |
  | STANndard                     |
RIMPedance

Click on a red keyword to view the command details.

see Also
Synchronizing the Analyzer and Controller
SCPI Command Tree

SENSe<cnum>:CORRection:TDR:COLLection:DLComp:LOAD <int>

Applicable Models: All with TDR Options (S9x011A/B)
This command executes load measurement, as a part of Loss Compensation sequence.

**Parameters**
- `<cnum>`
  - Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<int>`
  - Port number where load measurement is executed. The range is 1 to 4.

**Examples**
- `SENS:CORR:TDR:COLL:DLC:LOAD 1`
- `sense:correction:tdr:collection:dlcomp:load 1`

**Default**
- Not Applicable

---

This command executes open (with Thru) measurement, as a part of Loss Compensation sequence. This is used for Single Ended 1 Port only.

**Parameters**
- `<cnum>`
  - Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<int>`
  - Port number where open measurement is executed. The range is 1 to 4.

**Examples**
- `SENS:CORR:TDR:COLL:DLC:OPEN 1`
- `sense:correction:tdr:collection:dlcomp:open 1`

**Default**
- Not Applicable

---

This command saves the result of the Loss Compensation sequence.

**Parameters**
- `<cnum>`
  - Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

**Examples**
- `SENS:CORR:TDR:COLL:DLC:SAVE`

**Default**
- Not Applicable
SENSe<cnum>:CORRection:TDR:COLLection:DLComp:THRU <num>,<num>

Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) This command executes a thru measurement, as a part of Loss Compensation sequence.

Parameters

- `<cnum>`
  - Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

- `<num>,<num>`
  - Enter the port numbers where the thru is connected. For example, 1,2 indicates that the thru is connected between ports 1 and 2. The range is 1 to 4.

Examples

```
SENS:CORR:TDR:COLL:DLC:THRU 1,2
sense:correction:tdr:collection:dlcomp:thru 1,2
```

Default

Not Applicable

---

SENSe<cnum>:CORRection:TDR:COLLection:ECAL:FCOMp:IMMediate

Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) This command executes fixture compensation after ECAL.

Parameters

- `<cnum>`
  - Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

Examples

```
```

Default

Not Applicable

---

SENSe<cnum>:CORRection:TDR:COLLection:ECAL:IMMediate

Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) This command executes full calibration using the ECal module.

Parameters

- `<cnum>`
  - Channel number of the measurement. If unspecified, `<cnum>` is set to 1.

Examples

```
SENS:CORR:TDR:COLL:ECAL:IMM
sense:correction:tdr:collection:ecal:immediate
```

Default

Not Applicable
SENSe\(<cnum>\):CORRection:TDR:DCONstant \(<value>\)

Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the dielectric constant value.

**Parameters**

- \(<cnum>\) Channel number of the measurement. If unspecified, \(<cnum>\) is set to 1.
- \(<value>\) Dielectric constant value. The range is 10m to 100.

**Examples**

```
SENS:CORR:TDR:DCON 0.01
sense:correction:tdr:dconstant 0.01
```

**Query Syntax**

SENSe\(<cnum>\):CORRection:TDR:DCONstant?

**Return Type**

Double

**Default**

1

---

SENSe\(<cnum>\):CORRection:TDR:EXTension:AUTO:IMMediate

Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) This command executes deskew (auto port extension).

**Parameters**

- \(<cnum>\) Channel number of the measurement. If unspecified, \(<cnum>\) is set to 1.

**Examples**

```
SENS:CORR:TDR:EXT:AUTO:IMM
sense:correction:tdr:extension:auto:immediate
```

**Default**

Not Applicable

---

SENSe\(<cnum>\):CORRection:TDR:EXTension:AUTO:STANdard \(<enum>\)

Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the standard for deskew (auto port extension).

**Parameters**

- \(<cnum>\) Channel number of the measurement. If unspecified, \(<cnum>\) is set to 1.
- \(<enum>\) Standard for deskew. Choose from:
  - OPEN - Open termination.
SHORT - Short termination.

Examples

SENS:CORR:TDR:EXT:AUTO:STAN OPEN
sense:correction:tdr:extension:auto:standard open

Query Syntax
SENSecnum>:CORRection:TDR:EXTension:AUTO:STANdard?

Return Type
String

Default
OPEN

SENSe<cnum>:CORRection:TDR:RIMPedance <value>

Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the reference impedance value.

Parameters

<cnum> Channel number of the measurement. If unspecified, <cnum> is set to 1.

<value> Reference impedance value. The range is 1m to 10M.

Examples

SENS:CORR:TDR:RIMP 0.001
sense:correction:tdr:rimpedance 0.001

Query Syntax
SENSecnum>:CORRection:TDR:RIMPedance?

Return Type
Double

Default
50
Temperature

Applicable Models: This feature is available on ALL models except the N523x models and the 4-port N5244A and N5245A models with DSP Version 4.

**(Read-only)** Returns the temperature on the Receiver microcircuit.

Learn more about Receiver Temperature.

**Parameters**

- `<char>`: Choose from CELSius or FAHRenheit

**Examples**

- `SENS:TEMP? CELS`
- `sense:temperature? fahrenheit`

**Return Type**

- Numeric

**Default**

- Not Applicable
SENSe:ZA Commands

These commands control the compensation in the Impedance Measurement class.

```
SENSe:ZA
    CORRection:CSET:IMPort
    | FIXTure
    | SPARm

FIXTURE
    | RESPonse:FILE
    | CKIT
    | OPEN
    | G
    | C
    | SHORt
    | R
    | L

PORT
```

Click on a red keyword to view the command details.

See Also

- Synchronizing the Analyzer and Controller
- SCPI Command Tree

---

**SENSe<cnum>:ZA:CORRection:CSET:IMPort:FIXTure <bool>**

Applicable Models: All with Impedance Measurement Options (S9x041B)

*(Read-Write)* Sets and returns the ON | OFF state of Fixture Compensation. The Fixture Compensation data from the selected Cal Sets is enabled.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
ON | OFF state. Choose from
0 - Fixture Compensation is disabled
1 - Fixture Compensation is enable

Examples
SENS:ZA:CORR:CSET:IMP:FIXT ON
sense2:za:correction:cset:import:fixture 1

Query Syntax
SENSe<cnm>:ZA:CORRection:CSET:IMPort:FIXTure?

Return Type
Boolean

Default
OFF

SENSe<cnm>:ZA:CORRection:CSET:IMPort:SPARm <bool>

Applicable Models: All with Impedance Measurement Options (S9x041B)
(Read-Write) Sets and returns the ON | OFF state of S-Parameter calibration. The S-Parameter calibration data from the selected Cal Sets is enabled.

Parameters

<cnm>
Any existing channel number. If unspecified, value is set to 1

<bool>
ON | OFF state. Choose from
0 - S-Parameter calibration is disabled
1 - S-Parameter calibration is enable

Examples
SENS:ZA:CORR:CSET:IMP:SPAR ON
sense2:za:correction:cset:import:sparam 1

Query Syntax
SENSe<cnm>:ZA:CORRection:CSET:IMPort:SPARm?

Return Type
Boolean

Default
OFF

SENSe<cnm>:ZA:FIXTure:RESPonse:FILE <FileName>

Applicable Models: All with Impedance Measurement Options (S9x041B)
This command selects the fixture (16198A-010) calibration data file (.dat) which is provided by Keysight.

**Parameters**

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<FileName>` calibration data file

**Examples**

```
SENS:ZA:FIXT:SEL "C:\Fixture_Data_xxxxxxxxx.dat"
sense:za:fixture:select "C:\Fixture_Data_xxxxxxxxx.dat"
```

**Query Syntax**

`SENSe<cnum>:ZA:FIXTure:SELect?`

**Return Type**

String

**Default**

Null

---

`SENSe<cnum>:ZA:FIXTure:CKIT:OPEN:G <value>`

Applicable Models: All with Impedance Measurement Options (S9x041B)

(Read-Write) This command sets/returns the G value of Open Standard for the fixture compensation.

**Parameters**

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
- `<value>` G value of Open Standard

**Examples**

```
sense:za:fixture:ckit:open:g 1e-5
```

**Query Syntax**

`SENSe<cnum>:ZA:FIXTure:CKIT:OPEN:G?`

**Return Type**

Numeric

**Default**

0

---

`SENSe<cnum>:ZA:FIXTure:CKIT:OPEN:C <value>`

Applicable Models: All with Impedance Measurement Options (S9x041B)

(Read-Write) This command sets/returns the capacitance value of Open Standard for the fixture compensation.

**Parameters**

- `<cnum>` Channel number of the measurement. If unspecified, `<cnum>` is set to 1.
### Sense:C:CIT:OPEN:C

- **C** value of Open Standard

**Examples**

```
```

**Query Syntax**

`SENSe<cnum>:ZA:FIXTure:CKIT:OPEN:C?`

**Return Type**

Numeric

**Default**

0

### Sense:C:CIT:SHORT:R

- **<value>** R value of Open Standard

**Examples**

```
```

**Query Syntax**

`SENSe<cnum>:ZA:FIXTure:CKIT:SHORT:R?`

**Return Type**

Numeric

**Default**

0

### Sense:C:CIT:SHORT:L

- **<value>** L value of Open Standard

**Examples**

```
```

**Query Syntax**

`SENSe<cnum>:ZA:FIXTure:CKIT:SHORT:L?`

**Return Type**

Numeric

**Default**

0
**Query Syntax**  
SENSe<cnum>:ZA:FIXTure:CKIT:SHORt:L?

**Return Type**  
Numeric

**Default**  
0

---

**SENSe<cnum>:ZA:PORT <string>**

Applicable Models: All with Impedance Measurement Options (S9x041B)  
(Read-Write) This command sets/returns the port for the impedance

**Parameters**

- `<cnum>`: Channel number of the measurement. If unspecified, <cnum> is set to 1.
- `<string>`: Port number

**Examples**

- `SENS:ZA:PORT "1"`
- `sense:za:port "2"`

**Query Syntax**  
SENSe<cnum>:ZA:PORT?

**Return Type**  
Numeric

**Default**  
"1"
SourceDC
SOURce:DC commands

Controls the internal and external DC sources.

SOURce:DC:
  | CATalog?
  | CURRent
  | CLAMp
    | NEGative
    | POSitive
  | RANGe
  | DATA
  | ENABle
  | LIMit
    | MAXimum
    | MINimum
  | LOCK:OUTPut:RELay:CLOSed
  | PROTection
    | CATalog?
    | ENABle
    | LEVel
    | RESet
  | SEQuencing[:STATe]
    | STIMe
  | STARt
  | STATe
  | STOP
  | TYPE
Click on a keyword to view the command details.

**Note:** The VNA internal DC sources are named "AO1" and "AO2" using CAPITAL O (NOT zero - 0)

See Also
- Learn about DC Source control
- Configure an Ext DC Device (SCPI commands)
- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree
- Remotely Specifying a Source Port

---

**SOURce<ch>:DC:CATalog?**

Applicable Models: N524xB, E5080B, All PXIe/USB VNAs

*(Read-only)* Returns the names of the configured and active DC sources for the specified channel. Use `SYST:CONF:EDEV:CAT?` to read a list of all configured external devices, including inactive devices.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1</td>
</tr>
</tbody>
</table>

**Examples**

```
SOUR:DC:CAT?
'Returns the following...
"AO1,AO2,MyDCSupply"
'A01 and A02 are PNA-X internal DC Sources
```

**Return Type**

String, Names are separated by commas.

**Default**

Not Applicable
SOURce<ch>:DC:CURRent:CLAMP:NEGative <name>,<num>

Applicable Models: All PXIe VNAs

(Read-Write) Sets and returns the negative current limit (in Amps) when output regulation is in voltage priority mode. This command is available for PXI SMU only. This is the same function with "IKtM911xOutputChannelCurrent Interface” section, “NegativeLimit” property of the M911x driver.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

{name} String. DC Source Name. The name and port must EXACTLY match those in the DC Source Control dialog.

<num> Negative current limit value in Amps

Examples

SOUR2:DC:CURR:CLAMP:NEG "SMU1", -2

Query Syntax

SOURce<ch>:DC:VOLTage:CLAMP:NEGative? <name>

Return Type

Numeric

Default

-3.06

SOURce<ch>:DC:CURRent:CLAMP:POSitive <name>,<num>

Applicable Models: All PXIe VNAs

(Read-Write) Sets and returns the positive current limit (in Amps) when output regulation is in voltage priority mode. This command is available for PXI SMU only. This is the same function with "IKtM911xOutputChannelCurrent Interface” section, “PositiveLimit” property of the M911x driver.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

{name} String. DC Source Name. The name and port must EXACTLY match those in the DC Source Control dialog.

"SMU*" represents the SMU DC Meter name. “*” is the SMU module number.

<num> Positive current limit value in Amps

Examples

SOUR2:DC:CURR:CLAMP:POS "SMU1", 2

8206
SOURce<ch>:DC:VOLTage:CLAm:POSitive? <name>

Return Type  Numeric
Default  3.06

SOURce<ch>:DC:CURR:RANGe <name>,<num>

Applicable Models: All PXIe VNAs

(Read-Write) Sets and returns a current output range in Amps. This command is available for both PXI SMU and Digital/Analog I/O M9341B. For PXI SMU, this is the same function with “IKtM911xOutputChannelCurrent.Range Property” section of the M911x driver.

Parameters

<ch>  Any existing channel number. If unspecified, value is set to 1
[M9341B] If different setting is set between channels, the channel which is not same as active channel will be blocked to avoid wearing out an internal mechanical relay.
SENSe:SWEep:BLOCked? can detect the blocked status.

<name>  String. DC Source Name. The name and port must EXACTLY match those in the DC Source Control dialog.

[SMU] "SMU*" represents the SMU DC Meter name. “*” is the SMU module number.

[M9341B] "AO1“ or “AO2“

<num>  [SMU] Current range in Amps (0.01 or 3)

[M9341B] Current range in Amps. The range is 0.5 to 0.05, but Analog Out2 can’t be set to 0.5. If an invalid number is specified, the analyzer will round up to the closest valid number.

AO1 (Mode1: 0.5, Mode2: 0.05)
AO2 (Mode1: 0.1, Mode2: 0.05)

Examples
SOUR2:DC:CURR:RANG "SMU1",1m

Query Syntax  SOURce<ch>:DC:CURRent:RANGe? <name>
Return Type  Numeric
Default  (SMU) 3, (M9341B) 0.05

SOURce<ch>:DC:DATA <name[,port]>,<data>
(Read-Write) Sets and returns the DC stimulus values per point to be sent from the specified DC source. This setting overrides the Start and Stop DC settings for the channel. Only the values that are set with this command can be read by this command. The read command does NOT read the values that are set using the Start and Stop settings.

**Parameters**

- `<ch>`
  Any existing channel number. If unspecified, value is set to 1

- `<name[,port]>`
  String. The name and port must EXACTLY match those in the DC Source Control dialog.
  - **name** - DC Source Name. See note for specifying internal DC Sources.
  - **[port]** - Optional. VNA port for DC Source data. This is equivalent to the `<per port>` setting in the DC Source Control dialog.
  If unspecified, data is applied to ALL ports.
  
  **Important**: When you specify a port for this command, it must also be included for ALL DC Source commands that have this optional [port] specifier.

- `<data>`
  The stimulus value array to be set. The size of the array should be equal to the sweep point number.

**Examples**

```plaintext
'Set 3 data points on the channel
scpi.Execute "SENS:SWE:POIN 3"

'Enable DC Source output
scpi.Execute "SOUR:DC:ENAB ON"

'Set AO1 state to be always ON (no port value)
scpi.Execute "SOUR:DC:STATe 'AO1',ON"

'Set DC value for each data point
scpi.Execute "SOUR:DC:DATA 'AO1',1,5,1"

'Read data back
data = scpi.Execute ("SOUR:DC:DATA? 'AO1'")
msgbox data
```

**Query Syntax**

SOURce<ch>:DC:DATA? <name[,port]>

**Return Type**

Comma-separated values

**Default**

Not Applicable
Applicable Models: N524xB, E5080B, All PXIe/USB VNAs

(Read-Write) Sets and returns the ON / Off state of all configured DC sources for the specified channel. This setting is the same as the checkbox at the top of the DC Source Control dialog. 
Individual DC sources must ALSO be enabled using SOUR:DC:STATe.
Use SYST:CONF:EDEV:CAT? to read a list of all configured external devices.
Use SOUR:DC:CAT? to read a list of ACTIVE configured DC source names.

Parameters

- `<ch>`: Any existing channel number. If unspecified, value is set to 1.
- `<bool>`: ON / Off state. Choose from:
  - ON or 1 - All DC sources enabled.
  - OFF or 0 - All DC sources disabled.

Examples

```
SOUR:DC:ENAB 0
```

Query Syntax

SOURce<ch>:DC:Enable?

Return Type

Boolean

Default

1

---

SOURce<ch>:DC:LIMit:MAXimum <name>,<num>

Applicable Models: All

(Read-Write) Sets and returns the Max DC limit value for a DC source.

Parameters

- `<ch>`: Any existing channel number. If unspecified, value is set to 1.
- `<name>`: String. DC Source Name. See note for specifying internal DC Sources.
  The name must EXACTLY match those in the DC Source Control dialog.
- `<num>`: Max DC limit value. Choose a value within the range of the DC source.

Examples

```
SOUR2:DC:LIM:MAX "myDCSource",10
source:dc:limit:maximum "myDCSource",10
```
**SOURce<ch>:DC:LIMit:MAXimum? <name>**

**Return Type**
Numeric

**Default**
10

---

**SOURce<ch>:DC:LIMit:MINimum <name>,<num>**

Applicable Models: All

*(Read-Write)* Sets and returns the Min DC limit value for a DC source.

**Parameters**

- `<ch>`
  Any existing channel number. If unspecified, value is set to 1.

- `<name>`
  String. DC Source Name. See note for specifying internal DC Sources.
  The name must EXACTLY match those in the DC Source Control dialog.

- `<num>`
  Min DC limit value. Choose a value within the range of the DC source.

**Examples**

```
SOUR2:DC:LIM:MIN "myDCSource",-10
source:dc:limit:minimum "myDCSource",-10
```

---

**SOURce<ch>:DC:LOCK:OUTPut:RELay:CLOSed <name>, <bool>**

Applicable Models: All PXIe VNAs

*(Read-Write)* Sets and reads the locked-down state of the output relays on SMU units with output relays. When enabled/locked, the output relays remain closed. When disabled, the output relays open and close as the output toggles on and off. This command is available for PXI SMU only. This is the same function with “IKtM911xOutputChannelRelay Interface” section, “LockEnabled” property of the M911x driver.

**Parameters**

- `<ch>`
  Any existing channel number. If unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
**<name>**

String. DC Source Name. The name must EXACTLY match those in the DC Source Control dialog.

**<bool>**

ON / Off state. Choose from:

ON or 1 - The output relays remain closed

OFF or 0 - The output relays open and close as the output toggles on and off.

**Examples**

SOUR:DC:LOCK:OUTP:REL:CLOS 0

**Query Syntax**

SOURce<ch>:DC:LOCK:OUTPut:RELay:CLOSe? <name>

**Return Type**

Boolean

**Default**

0

---

**SOURce<ch>:DC:PROTection:CATalog?**

Applicable Models: All PXIe VNAs, E5080B

*(Read-only)* Returns the module list or name of DC source in which protection function is activated due to over current or voltage. This command is available for both PXI SMU and Digital/Analog I/O M9341B and E5080B.

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

**Examples**

SOUR:DC:PROT:CAT?

**Query Syntax**

SOURce<ch>:DC:PROTection:CATalog?

**Return Type**

String, Names are separated by commas. "NONE" is returned if no modules are detected.

**Default**

Not Applicable

---

**SOURce<ch>:DC:PROTection:ENABle <name>, <bool>**

Applicable Models: All PXIe VNAs

*(Read-Write)* Sets and reads the DC output protection state. This command is available for PXI SMU only.

**Parameters**
### SOURce<ch>:DC:PROTection:ENABle <name>, <bool>

Applicable Models: All PXIe VNAs

**(Read-Write)** Sets and reads the protection voltage level for the DC output. This command is available for PXI SMU only.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;ch&gt;</code></td>
</tr>
<tr>
<td><code>&lt;name&gt;</code></td>
</tr>
<tr>
<td><code>&lt;bool&gt;</code></td>
</tr>
</tbody>
</table>

**Examples**

```
SOUR:DC:PROT:ENAB "SMU1", 0
```

**Query Syntax**

```
SOURce<ch>:DC:PROTection:ENABle? <name>
```

**Return Type**

Boolean

**Default**

1

### SOURce<ch>:DC:PROTection:LEVel <name>, <num>

Sets and reads the protection voltage level for the DC output. This command is available for PXI SMU only.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;ch&gt;</code></td>
</tr>
<tr>
<td><code>&lt;name&gt;</code></td>
</tr>
<tr>
<td><code>&lt;bool&gt;</code></td>
</tr>
</tbody>
</table>

**Examples**

```
SOUR:DC:PROT:LEV "SMU1", 10
```

**Query Syntax**

```
SOURce<ch>:DC:PROTection:LEVel? <name>
```

**Return Type**

Numeric

**Default**

14.4

### SOURce<ch>:DC:PROTection:RESet <name>

8212
Applicable Models: All PXIe VNAs, E5080B

(Write Only) Resets the instrument's output protection circuit after a protection condition occurs. If successful, the output will return to the output enabled state. Remove the fault condition that caused the protection trip before resetting the protection circuit in order to prevent the protection from tripping again. This command is available for both PXI SMU and Digital/Analog I/O M9341B and E5080B.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

<string> String. DC Source Name. The name must EXACTLY match those in the DC Source Control dialog.

[SMU] "SMU*" represents the SMU DC Meter name. "*" is the SMU module number.

[M9341B] "AO1" or "AO2"

[E5080B] "AO1" or "AO2" or both

Examples

SOUR:DC:PROT:RES "SMU1"

Query Syntax Not Applicable

Return Type Not Applicable

Default Not Applicable

SOURce<ch>:DC:NAMes?

Applicable Models: N524xB, E5080B, All PXIe/USB VNAs

(Read-only) Reads the DC source names

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

Examples

SOUR:DC:NAM?

Query Syntax SOURce<ch>:DC:NAMes?

Return Type

Default Not Applicable
SOURce<ch>:DC:SEQuencing[:STATe] <bool>

Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the DC output sequencing state. This command is valid for SMU sources only. Other DC sources, such as “AO1”, “AO2”, do not have the sequencing control. This command is available for PXI SMU only.

Parameters


Any existing channel number. If unspecified, value is set to 1.

String. DC Source Name. The name must EXACTLY match those in the DC Source Control dialog.

ON / Off state. Choose from:

ON or 1
OFF or 0

Examples

SOUR:DC:SEQ 1

Query Syntax SOURce<ch>:DC:SEQuencing[:STATe]?

Return Type Boolean

Default 0

SOURce<ch>:DC:SEQuencing:STIMe <num>

Applicable Models: All PXIe VNAs

(Read-Write) Sets and reads the DC output sequencing settling time. This command is available for PXI SMU only. Other DC sources, such as “AO1”, “AO2”, do not have the sequencing control.

Parameters


Any existing channel number. If unspecified, value is set to 1.

String. DC Source Name. The name must EXACTLY match those in the DC Source Control dialog.

Source settling time in second.

Examples

SOUR:DC:SEQ:STIM 1
### Query Syntax

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>SOURce&lt;ch&gt;:DC:SEQuencing:STIMe?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
</tbody>
</table>

### SOURce<ch>:DC:STARt <name[,port]>,<num>

**Applicable Models:** N524xB, E5080B, All PXIe/USB VNAs

*(Read-Write)* Sets and returns start DC value for the specified DC source.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1
- `<name[,port]>`: String. The name and port must EXACTLY match those in the DC Source Control dialog.
  - name - DC Source Name. See note for specifying internal DC Sources.
  - [port] - Optional. VNA port for DC Source start value. This is equivalent to the <per port> setting in the DC Source Control dialog.
  - If unspecified, DC Start value is applied for ALL ports.

**Examples**

```plaintext```
SOUR2:DC:STARt "myDCSource",3
source:dc:start "AO1,Port 1",1
```

**Query Syntax**

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>SOURce&lt;ch&gt;:DC:START? &lt;name[,port]&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>.5</td>
</tr>
</tbody>
</table>

### SOURce<ch>:DC:STATe <name[,port]>,<bool>

**Applicable Models:** N524xB, E5080B, All PXIe/USB VNAs

*(Read-Write)* Sets and returns the ON / Off state of the specified DC source and port.

Use SYST:CONF:EDEV:CAT? to read a list of all configured external devices.
Use `SOUR:DC:CAT?` to read a list of ACTIVE configured DC source names

Parameters

- `<ch>`
  Any existing channel number. If unspecified, value is set to 1

- `<name[,port]>`
  String. The name and port must EXACTLY match those in the DC Source Control dialog. Name - DC Source Name. See note for specifying internal DC Sources.
  [port] - Optional. VNA port for DC Source state. This is equivalent to the <per port> setting in the DC Source Control dialog.

  If unspecified, DC is ALWAYS ON or OFF for ALL ports.

  **Important:** When you specify a port for this command, it must also be included for ALL DC Source commands that have this optional [port] specifier.

- `<bool>`
  ON / Off state. Choose from:
  ON or 1 - DC source/port enabled.
  OFF or 0 - DC source/port disabled.

Examples

- 'Set AO1 to always ON
  `source:dc:state "AO1",1`
  'Set MyDCSource to ON when RF source Port 1 is ON
  `SOUR:DC:STAT "MyDCSource,Port 1",ON`
  'Read state for MyDCSource,Port 1
  `SOUR:DC:STAT? "MyDCSource,Port 1"`

Query Syntax

```
SOURce<ch>:DC:STATe? <name[,port]>
```

Return Type: Boolean

Default: 0 - OFF

**SOURce<ch>:DC:STOP <name[,port]>,<num>**

Applicable Models: N524xB, E5080B, All PXIe/USB VNAs

(Read-Write) Sets and returns stop DC value for the specified DC source.

Parameters

- `<ch>`
  Any existing channel number. If unspecified, value is set to 1
<name[,port]>

String. The name and port must EXACTLY match those in the DC Source Control dialog.

name - DC Source Name. See note for specifying internal DC Sources.

[port] - Optional. VNA port for DC Stop value. This is equivalent to the <per port> setting in the DC Source Control dialog.

If unspecified, DC Stop is applied for ALL ports.

Important: When you specify a port for this command, it must also be included for ALL DC Source commands that have this optional [port] specifier.

<num>

Stop value. Choose a value within the range of the DC source.

Examples

SOUR2:DC:STOP "myDCSource", 3

Query Syntax

SOURce<ch>:DC:STOP?<name[,port]>

Return Type

Numeric

Default

0

SOURce<ch>:DC:TYPE <name>,<char>

Applicable Models: All PXIe VNAs

(Read-Write) Sets and returns the output regulation mode. This command is available for PXI SMU only. This is the same function with “IKtM911xOutputChannel Interface” section, “PriorityMode” property of the M911x driver.

Parameters

<ch>

Any existing channel number. If unspecified, value is set to 1

$name$

String. DC Source Name. The name and port must EXACTLY match those in the DC Source Control dialog.

<char>

Source control type.

VOLTage - voltage priority

CURRent - current priority

Examples

SOUR2:DC:TYPE "SMU1",VOLT

Query Syntax

SOURce<ch>:DC:TYPE? <name>
Return Type: Character

Default: VOLTage

**SOURce<ch>:DC:VOLTage:CALIbrate:DATE <name>**

Applicable Models: All PXIe VNAs

*(Read only)* Returns the date of execution output voltage calibration. This command is available for M9341B only.

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

<name> String. DC Source Name. The name and port must EXACTLY match those in the DC Source Control dialog “AO1” or “AO2” or both.

**Examples**

```
SOUR2:DC:VOLT:CALB:DATE? "AO1,AO2"
```

**Query Syntax**

SOURce<ch>:DC:VOLTage:CALIbrate:DATE? <name>

**Return Type** String. Comma separated numbers representing year, month and day.

**Default** Not Applicable

---

**SOURce<ch>:DC:VOLTage:CALIbrate:EXECute <name>**

Applicable Models: All PXIe VNAs

*(Write only)* Execute the output voltage calibration for both AO1 and AO2. The calibration is finished within 1 second, so this is not an overlapped command. This command is available for M9341B only.

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1. This command is common for all channels and the setting is ignored.

<name> String. DC Source Name. The name and port must EXACTLY match those in the DC Source Control dialog “AO1” or “AO2” or both. Even either "AO1" or "AO2" is selected, both ports are calibrated at the same time.
### SOURce<ch>:DC:VOLTage:CALIbrate:TIME <name>

**Applicable Models:** All PXIe VNAs

*(Read only)* Returns the time of execution output voltage calibration. If the calibration is not executed, “[SCPI: 1111]: Calibration data is missing.” is returned. This command is available for M9341B only.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1. This command is common for all channels and the setting is ignored.
- `<name>`: String. DC Source Name. The name and port must EXACTLY match those in the DC Source Control dialog “AO1” or “AO2” or both.

**Examples**

```
SOUR2:DC:VOLT:CALB:TIME "AO1, AO2"
```

**Query Syntax**

SOURce<ch>:DC:VOLTage:CALIbrate:TIME? <name>

**Return Type**

String. Comma separated numbers representing hours, minutes and seconds.

**Default**

Not Applicable

---

### SOURce<ch>:DC:VOLTage:BANDwidth <name>,<string>

**Applicable Models:** All PXIe VNAs

*(Read-Write)* Sets and returns a voltage bandwidth to optimize the output response time with capacitive loads. This command is available for PXI SMU only. This is the same function with “IKtM911xOutputChannelVoltage.Bandwidth Property” section of the M911x driver.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1
<name> String. DC Source Name. The name and port must EXACTLY match those in the DC Source Control dialog. "SMU*" represents the SMU DC Meter name. "*" is the SMU module number.

<string> Source control type. "LOW", "HIGH1", "HIGH2", or "HIGH3"

Examples SOUR2:DC:VOLT:CLAM "SMU1", "LOW"

Query Syntax SOURce<ch>:DC:VOLTage:CLAMp? <name>

Return Type String

Default "LOW"

SOURce<ch>:DC:VOLTage:CLAMp <name>,<num>

Applicable Models: All PXIe VNAs

(Read-Write) Sets and returns the positive voltage limit in Volts when output regulation is in current priority mode. This command is available for PXI SMU only. This is the same function with "IKtM911xOutputChannelVoltage Interface" section, “Limit” property of the M911x driver.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<string> String. DC Source Name. The name and port must EXACTLY match those in the DC Source Control dialog. "SMU*" represents the SMU DC Meter name. "*" is the SMU module number.

<num> Limit value in Volts

Examples SOUR2:DC:VOLT:CLAM "SMU1", 5

Query Syntax SOURce<ch>:DC:VOLTage:CLAMp? <name>

Return Type Numeric

Default 6.12

SOURce<ch>:DC:VOLTage:RANGe <name>,<num>
Applicable Models: All PXIe VNAs

(Read-Write) Sets and returns a voltage output range in Volts. This command is available for both PXI SMU only. This is the same function with “IKtM911xOutputChannelVoltage.Range Property” section of the M911x driver.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

$name$ String. DC Source Name. The name and port must EXACTLY match those in the DC Source Control dialog.

"SMU*" represents the SMU DC Meter name. “*” is the SMU module number.

<num> Voltage range (6 or 13)

Examples

SOUR2:DC:VOLT:RANG "SMU1", 13

Query Syntax

SOURce<ch>:DC:VOLTage:RANGE? <name>

Return Type

Numeric

Default

6
Source Multi-Dimensional Sweep

Controls for setting up a multi-dimensional sweep.

**SOURce:**

<table>
<thead>
<tr>
<th>DC:</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIimension:</td>
</tr>
<tr>
<td>ORDer</td>
</tr>
<tr>
<td>[:STATe]</td>
</tr>
<tr>
<td>DIimension:</td>
</tr>
<tr>
<td>CATalog?</td>
</tr>
<tr>
<td>COUNT?</td>
</tr>
<tr>
<td>POINTs</td>
</tr>
<tr>
<td>REPeat:</td>
</tr>
<tr>
<td>COUNT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FREQuency:</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIimension:</td>
</tr>
<tr>
<td>ORDer</td>
</tr>
<tr>
<td>[:STATe]</td>
</tr>
<tr>
<td>FIXed</td>
</tr>
<tr>
<td>STARt</td>
</tr>
<tr>
<td>STOP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PHASe:</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIimension:</td>
</tr>
<tr>
<td>ORDer</td>
</tr>
<tr>
<td>[:STATe]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>POWer:</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIimension:</td>
</tr>
<tr>
<td>ORDer</td>
</tr>
<tr>
<td>[:STATe]</td>
</tr>
</tbody>
</table>
Click on a keyword to view the command details.

**see Also**

- Example Programs
- Synchronizing the Analyzer and Controller
- SCPI Command Tree

### SOURce<ch>:DC:DImension:ORDer <name[,port]>,<value>

**Applicable Models:** N522xB, N523xB, N524xB

(Read-Write) Set and read the order for the specified DC source in the multi-dimensional sweep.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Any existing SA channel. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;name[,port]&gt;</td>
<td>String. The name and port must EXACTLY match those in the DC Source Control dialog.</td>
</tr>
<tr>
<td>[port]</td>
<td>Optional. PNA port for DC Source start value. This is equivalent to the &lt;per port&gt; setting in the DC Source Control dialog.</td>
</tr>
</tbody>
</table>

- **Important:** When you specify a port for this command, it must also be included for ALL DC Source commands that have this optional [port] specifier.

<table>
<thead>
<tr>
<th>Examples</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOUR:DC:DIM:ORDER &quot;myDCSource&quot;,3</td>
<td></td>
</tr>
<tr>
<td>source:dc:dimension:order &quot;AO1,Port 1&quot;,1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default</th>
<th>Return Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Integer</td>
<td></td>
</tr>
</tbody>
</table>

### SOURce<ch>:DC:DImension[:STATe] <name[,port]>,<bool>

**Applicable Models:** N522xB, N523xB, N524xB

(Read-Write) Set and read the specified DC source’s ON/OFF state in the multi-dimensional sweep.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Any existing SA channel. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;name[,port]&gt;</td>
<td>String. The name and port must EXACTLY match those in the DC Source Control dialog.</td>
</tr>
<tr>
<td>[port]</td>
<td>Optional. PNA port for DC Source start value. This is equivalent to the &lt;per port&gt; setting in the DC Source Control dialog.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce&lt;ch&gt;:DC:DImension:ORDer? &lt;name[,port]&gt;</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Default</th>
<th>Return Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Integer</td>
<td></td>
</tr>
</tbody>
</table>
If unspecified, DC Start value is applied for ALL ports.

**Important:** When you specify a port for this command, it must also be included for ALL DC Source commands that have this optional [port] specifier.

Choose from:

0 - **OFF** - Disable the specified DC source in multi-dimensional sweep.

1 - **ON** - Enable the specified DC source in multi-dimensional sweep.

**Examples**

```
SOUR:DC:DIM "AO1",1
source:dc:dimension "MyDCSource,Port 1",ON
```

**Query Syntax**

```
SOURce<ch>:DC:DIEMension[:STATe]? <name[,port]>
```

**Return Type**

Boolean

**Default**

0

---

**SOURce<ch>:DIMension<dim>:CATalog?**

**Applicable Models:** N522xB, N523xB, N524xB

(Read-only) Read the names of source domains in the multi-dimensional sweep whose state is ON and whose dimension order is the specified dimension order.

**Parameters**

- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
- `<dim>`: Dimension order. If unspecified, value is set to 1.

**Examples**

```
SOUR:DIM3:CAT?
'
Returns the following...
"SOUR:FREQ1,SOUR:FREQ3"
```

**Return Type**

String, Names are separated by commas.
Not Applicable

**Default**

Not Applicable

---

**SOURce<ch>:DIMension:COUNt?**

**Applicable Models:** N522xB, N523xB, N524xB

(Read-only) Read the highest dimension order in the multi-dimensional sweep.

**Parameters**

- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.

**Examples**

```
SOUR:DIM:COUN?
```

**Return Type**

Numeric

**Default**

0

---

**SOURce<ch>:DIMension<dim>:POINts <value>**

**Applicable Models:** N522xB, N523xB, N524xB

(Read-Write) Set and read the point count for the specified dimension order in the multi-dimensional sweep.
PARAMETERS

Any existing SA channel. If unspecified, value is set to 1.
Dimension order. If unspecified, value is set to 1.

VALUE

Point count. Choose an integer value of 1 or higher.

Examples

SOUR:DIM2:POIN 3
source:dimension4:points 4

QUERY SYNTAX

SOURce<ch>:DIМension<dim>:POINts?

RETURN TYPE

Integer

Default

1

SOURce<ch>:DIМension<dim>:REP:COUNT <value>

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Set and read the repeat count for the specified dimension order in the multi-dimensional sweep.

Parameters

Any existing SA channel. If unspecified, value is set to 1.
Dimension order. If unspecified, value is set to 1.

VALUE

Repeat count. Choose an integer value of 1 or higher.

Examples

SOUR:DIM2:REP:COUN 3
source:dimension4:repeat:count 4

QUERY SYNTAX

SOURce<ch>:DIМension<dim>:REP:COUNT?

RETURN TYPE

Integer

Default

1

SOURce<ch>:FREQuency<port>:DIМension:ORDer <value>[,src]

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Set and read the source frequency domain’s order in the multi-dimensional sweep.

Parameters

Any existing SA channel. If unspecified, value is set to 1.
Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
Dimension order. Choose an integer value of 1 or higher.

Examples

SOUR:FREQ:DIM:ORDER 2
source2:frequency:dimension:order 4,"Port 1 Src2"

QUERY SYNTAX

SOURce<ch>:FREQuency<port>:DIМension:ORDer?

RETURN TYPE

Integer

Default

1

SOURce<ch>:FREQuency<port>:DIМension[:STATe] <bool>[,src]
Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Set and read the source frequency domain's ON/OFF state in the multi-dimensional sweep.

Parameters
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<port>` Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.
- `<bool>` Choose from:
  - 0 - OFF - Disable the specified source frequency domain in multi-dimensional sweep.
  - 1 - ON - Enable the specified source frequency domain in multi-dimensional sweep.
- `[src]` String. (NOT case sensitive). Source port. Optional. Use SOUR:CAT? to return a list of valid port names. While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

Examples
```
SOUR:FREQ:DIM 1
source2:frequency:dimension 0,"Port 1 Src2"
```

Query Syntax
```
SOURce<ch>:FREQuency<port>:DIMension[:STATe]?
```

Return Type
Boolean

Default
0

Applicable Models: N522xB, N523xB, N524xB

(Read-WRITE) Set and read the fixed frequency value for a specific port.

Parameters
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<port>` Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<num>` fixed frequency value.
- `[src]` String. (NOT case sensitive). Source port. Optional. Use SOUR:CAT? to return a list of valid port names. While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

Examples
```
SOUR:FREQ:FIX 1e9
source2:frequency:fixed 1e9,"Port 1 Src2"
```

Query Syntax
```
SOURce<ch>:FREQuency<port>[:FIXed]?
```

Return Type
Numeric

Default
1 GHz

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Set and read the start frequency value for a specific port.

Parameters
- `<ch>` Any existing SA channel. If unspecified, value is set to 1.
- `<port>` Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

Examples
```
SOUR:FREQ:STARt 1e9
source2:frequency:start 1e9,"Port 1 Src2"
```

Query Syntax
```
SOURce<ch>:FREQuency<port>[:STARt]?
```

Return Type
Numeric

Default
1 GHz
### SOURce<ch>:FREQuency<port>:STARt <num>,[src]

Start frequency value.

- **Parameters**
  - `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
  - `<port>`: Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.

- **Examples**
  - SOUR:FREQ:STAR 1e9
  - source2:frequencystart 1e9,"Port 1 Src2"

- **Query Syntax**
  - SOURce<ch>:FREQuency<port>:STARt?

- **Return Type**
  - Numeric

- **Default**
  - Minimum frequency of the PNA.

### SOURce<ch>:FREQuency<port>:STOP <num>[,src]

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Set and read the stop frequency value for a specific port.

- **Parameters**
  - `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
  - `<port>`: Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.

- **Examples**
  - SOUR:FREQ:STOP 1e9
  - source2:frequencystop 1e9,"Port 1 Src2"

- **Query Syntax**
  - SOURce<ch>:FREQuency<port>:STOP?

- **Return Type**
  - Numeric

- **Default**
  - Maximum frequency of the PNA.

### SOURce<ch>:PHASe<port>:DIMension:ORDer <value>[,src]

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Set and read the source phase domain's order in the multi-dimensional sweep.

- **Parameters**
  - `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
  - `<port>`: Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.

- **Examples**
  - SOUR:PHAS:DIM:ORDER 2
SOURce<ch>:PHAS<port>:DIMension[:STATe] <bool>[,src]

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Set and read the source phase domain’s ON/OFF state in the multi-dimensional sweep.

Parameters

- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
- `<port>`: Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<bool>`: Choose from:
  - 0 - OFF - Disable the specified source phase domain in multi-dimensional sweep.
  - 1 - ON – Enable the specified source phase domain multi-dimensional sweep.

Examples

SOUR:PHAS:DIM 1

source2:phase:dimension:order 0,"Port 1 Src2"

SOURce<ch>:POWer<port>:DIMension:ORDER <value>[,src]

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Set and read the source power domain’s order in the multi-dimensional sweep.

Parameters

- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
- `<port>`: Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<value>`: Dimension order. Choose an integer value of 1 or higher.

Examples

SOUR:POW:DIM:ORDER 2

source2:power:dimension:order 4,"Port 1 Src2"
SOURce<ch>:POWer<port>:DIMension[:STATe] <bool>[,src]

**Applicable Models:** N522xB, N523xB, N524xB

(Read-Write) Set and read the source power domain’s ON/OFF state in the multi-dimensional sweep.

**Parameters**

- `<ch>`: Any existing SA channel. If unspecified, value is set to 1.
- `<port>`: Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.
- `<bool>`: Choose from:
  - 0 - OFF - Disable the specified source power domain in multi-dimensional sweep.
  - 1 - ON - Enable the specified source power domain in multi-dimensional sweep.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an **external source**, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as “Port 1 Src2”. Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

**Examples**

```
SOUR:FREQ:DIM 1
source2:power:dimension 0,"Port 1 Src2"
```

**Query Syntax**

SOURce<ch>:POWer<port>:DIMension[:STATe]?

**Return Type**

Boolean

**Default**

0
SourcePhase

Makes phase control settings (Option S93088A)

```
SOURce:PHASe:
  | CONTROL
  | COUPLE[:STATE]
  | ITERation
  | TOLERance
  | CORREction
  | DATA
  | [:STATE]
  | EXTERNAL
  | CATalog?
  | PORT
  | [FIXed]
  | MODE
  | CATalog?
  | [:VALUE]
  | PARameter
  | CATalog?
  | MODE
  | CATalog?
  | PORT
  | [:VALUE]
  | OFFSET
  | CORREction
  | DATA
  | [:STATE]
```
SOURce<ch>:=PHASE<port>:CONTrol:COUPle[:STATe] <bool>[,src]

Applicable Models: N522xB, N524xB

(Read-Write) Write and read whether to couple phase control settings (IFBW, Tolerance, Max Iterations).

**Parameters**

- `<ch>`
  Channel number of phase control measurement. If unspecified, value is set to 1.

- `<port>`
  Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

- `<bool>`
  Coupling state. Choose from:

  - ON or 1 - Couple phase control settings. The phase control settings from `<port>` are copied to the other phase-controlled ports.
  - OFF or 0 - Do NOT couple phase control settings. The phase control settings for each phase-controlled port are made independently.

- `[src]`

  While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the
<port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

### Examples

```
SOUR:PHAS2:CONT:COUP 1
source2:phase:control:couple:state ON,"Port 1 Src2"
```

### Query Syntax

```
SOURce<ch>:PHASe<port>:CONTrol:COUPle[:STATe]? [src]
```

### Return Type

Boolean

### Default

OFF

---

SOURce<ch>:PHASe<port>:CONTrol:ITERation <num>[,src]

Applicable Models: N522xB, N524xB

(Read-Write) Write and read the maximum number of background phase sweeps to perform.

### Parameters

- `<ch>`
  - Channel number of phase control measurement. If unspecified, value is set to 1.

- `<port>`
  - Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

- `<num>`
  - Number of background sweep iterations. Choose a value between 1 and 25.

- `[src]`

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

### Examples

```
SOUR:PHAS2:CONT:ITER 3
source2:phase:control:iteration 4,"Port 1 Src2"
```

### Query Syntax

```
SOURce<ch>:PHASe<port>:CONTrol:ITERation? [src]
```

### Return Type

Numeric

### Default

10
### SOUR<ch>:PHA<port>:CONT:TO<ch>le<port>:TOLerance <num>[,.src]

#### Parameters

- `<ch>`: Channel number of phase control measurement. If unspecified, value is set to 1.
- `<port>`: Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.
- `<num>`: Tolerance for background sweeps in degrees. Choose a value between 1 and 5.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

Use **SOUR:CAT?** to return a list of valid port names.

#### Examples

```
SOUR:PHAS2:CONT:TOL 3
source2:phase:control:tolerance 6,"Port 1 Src2"
```

#### Query Syntax

```
SOURce<ch>:PHASe<port>:CONTRol:TOLerance? [src]
```

#### Return Type

Numeric

#### Default

1 degree

---

### SOUR<ch>:PHA<port>:COR<ch>rection:DATA <data>[,.src]

#### Parameters

- `<ch>`: Channel number of phase control measurement. If unspecified, value is set to 1.
- `<port>`: Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.

---
Phase offset data array.

[String. (NOT case sensitive). Source port. Optional argument.]

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

### Examples

```
SOUR:PHAS2:CORR:DATA
source2:phase:correction:data 10,15,20,"Port 1 Src2"
```

### Query Syntax

```
SOURce<ch>:PHASe<port>:CORRection:DATA? [src]
```

### Return Type

Depends on FORMat:DATA

---

**SOURce<ch>:PHASe<port>:CORRection[:STATe] <bool>[,src]**

Applicable Models: N522xB, N524xB

(Read-Write) Write and read whether to use the phase correction offset array.

### Parameters

- **<ch>**
  
  Channel number of phase control measurement. If unspecified, value is set to 1.

- **<port>**
  
  Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

- **<bool>**
  
  Phase correction array state.

  - ON (1) Apply phase correction offset array.
  - OFF(0) Do NOT apply phase correction offset array.

- **[src]**
  

  While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.
Use `SOUR:CAT?` to return a list of valid port names.

**Examples**

```plaintext
SOUR:PHAS2:CORR 1
source2:phase:correction:state OFF,"Port 1 Src2"
```

**Query Syntax**

```
SOURce<ch>:PHASe<port>:CORRection[:STATe]? [src]
```

**Return Type**

Boolean

**Default**

OFF

---

**SOURce<ch>:PHASe<port>:EXTernal:CATalog?**

**Applicable Models:** N522xB, N524xB

*(Read-only)* Returns the available internal ports that the external port can be set to. This command is only intended to be executed for external sources. The command will execute if the port is not an external source but does not do anything meaningful.

**Parameters**

- `<ch>`
  
  Channel number of phase control measurement. If unspecified, value is set to 1.

- `<port>`
  
  External port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.

- `[src]`
  

  While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

  Use `SOUR:CAT?` to return a list of valid port names.

**Examples**

```plaintext
SOUR:PHAS2:EXT:CAT?
source2:phase:external:catalog?
```

**Return Type**

String of comma-separated ports.

**Default**

Not Applicable

---

**SOURce<ch>:PHASe<port>:EXTernal:PORT <num>[,src]**

**Applicable Models:** N522xB, N524xB
Sets and returns the internal port that the external port is routed through. This command is only intended to be executed for external sources. The command will execute if the port is not an external source but does not do anything meaningful.

**Parameters**

- `<ch>`: Channel number of phase control measurement. If unspecified, value is set to 1.
- `<port>`: External port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<num>`: Internal port number.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

**Examples**

The following sets source port "Port 1 Src2" on channel 2 to use "Port 1" for the measurement
SOUR2:PHAS:EXT:PORT 1,"PORT 1 Src2"

The following sets source port 2 on channel 1 to use Port 1 for the measurement
source:phas2:external:port 1

**Query Syntax**

SOURce<ch>:PHASe<port>:EXTernal:PORT? [src]

**Return Type**

Character

**Default**

3

---

SOURce<ch>:PHAS<port>[:FIXed] <num>[,src]

Applicable Models: N522xB, N524xB

(Read-Write) Write and read the fixed phase value. Must not be in logarithmic sweep.

**Parameters**

- `<ch>`: Channel number of phase control measurement. If unspecified, value is set to 1.
Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

Phase value in degrees. Choose a value between -360 and 360.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

Examples

SOUR:PHAS2 60
source2:phase:fixed 120,"Port 1 Src2"

Query Syntax

SOURce<ch>:PHASe<port>[:FIXed]? [src]

Return Type

Numeric

Default

0 degrees

SOURce<ch>:PHASE<port>:MODE:CATalog?

Applicable Models: N522xB, N524xB

(Read-only) Returns the available phase control modes for the specified port.

Parameters

<ch> Channel number of phase control measurement. If unspecified, value is set to 1.

<port> Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.
### Examples

```
SOUR:PHAS2:MODE:CAT?
source2:phase:mode:catalog?
```

### Return Type

String of comma-separated phase control modes.

For example, OFF, OPENloop, PARameter, REFerence.

### Default

Not Applicable

---

**SOURce<ch>:PHASe<port>:MODE[:VALue] <char>[,src]**

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the Phase Control mode.

#### Parameters

- `<ch>`: Channel number of phase control measurement. If unspecified, value is set to 1.
- `<port>`: Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<char>`: Choose from:
  - OFF - Turn phase control OFF
  - OPENloop - Sets a raw phase value for either swept phase or fixed phase, but no receivers are used to control the phase.
  - PARameter - Sets and controls the phase of the signal at <port>.
  - REFerence (Read-only) - Use SOUR:PHAS:PARarameter to set.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

#### Examples

```
SOUR:PHAS2:MODE PAR
source2:phase:mode:value off
```

#### Query Syntax

```
SOURce<ch>:PHASe<port>:MODE[:VALue]? [src]
```

#### Return Type

Character

#### Default

OFF

---

**SOURce<ch>:PHASE<port>:PARameter <string>[,src]**
Applicable Models: N522xB, N524xB

(Read-Write) Write and read the ratioed receivers (parameter) to use for phase control.

Parameters

- `<ch>`
  Channel number of phase control measurement. If unspecified, value is set to 1.

- `<port>`
  Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.

- `<string>`
  Ratioed parameter. Choose any two VNA physical receivers. Use either standard receiver notation ("R/R3") or logical receiver notation ("a1/a3"). Separate the two receiver names by a forward slash '/'. For example: "a3/a1".

- `[src]`

  While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

  Use SOUR:CAT? to return a list of valid port names.

Examples

- SOUR:PHAS2:PAR: "a3/a1", "Port 3"
- source2:phase:parameter "a3/a1", "Port 3"

Query Syntax

- SOURce<ch>:PHASe<port>:PARameter?
  Returns the ratioed parameter name.

Return Type

- String

Default

- "a1/b1"

SOURce<ch>:PHASe<port>:PARameter:CATalog?

Applicable Models: N522xB, N524xB

(Read-only) Returns the available parameters that are set using the SOURce:PHASe:PARameter command. This command applies to phase control only. This command does not apply to DIQ. DIQ does not restrict the parameters that can be set. It shows an empty string if nothing is settable. Usually this means that the reference port has not been set yet. SOUR:PHAS:REF:PORT <port> needs to be set first.

For an external source, SOURce:PHASe:EXTernal:PORT <port> should be set first, then the reference port, then the control parameter. But I try to figure out the relevant information, if the external port or reference is not set.
Parameters

<ch> Channel number of phase control measurement. If unspecified, value is set to 1.

<port> Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

Examples

SOUR:PHAS2:REFCAT?

source2:phase:reference:catalog?

Return Type

String of comma-separated phase control parameters.

Default Not Applicable

SOURce<ch>:PHAS<port>:PARameter:MODE <char>[,src]

Applicable Models: N522xB, N524xB
(Read-Write) Sets and returns the Phase Control mode.

Parameters

<ch> Channel number of phase control measurement. If unspecified, value is set to 1.

<port> Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<char> Choose from:
OFF - Turn phase control OFF
OPENloop - Sets a raw phase value for either swept phase or fixed phase, but no receivers are used to control the phase.
PARameter - Sets and controls the phase of the signal at <port>.
REFerence (Read-only) - Use SOUR:PHAS:PARameter to set.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the
<port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use **SOUR:CAT?** to return a list of valid port names.

### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
</table>
| **SOUR:PHAS2:PAR:MODE PAR**
**source2:phase:parameter:mode off** | Character | OFF |

### Query Syntax

**SOUR**:<ch>:**PHASE**:<port>:**PARAMETER**:**MODE**? [src]

### Return Type

Character

### Default

OFF

---

**SOURce**:<ch>:**PHASE**:<port>:**PARameter**:**MODE**:**CATalog**?

**Applicable Models:** N522xB, N524xB

**Parameters**

- **<ch>**
  - Channel number of phase control measurement. If unspecified, value is set to 1.

- **<port>**
  - Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

- **[src]**

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an **external source**, **true mode balanced port**, or one of the **Source 2 outputs** on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use **SOUR:CAT?** to return a list of valid port names.

### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
</table>
| **SOUR:PHAS2:PAR:MODE:CAT?**

For example, OFF, OPENloop, PARameter, REFerence.
Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the reference port for the Phase Control measurement.

Parameters

<ch>
Channel number of phase control measurement. If unspecified, value is set to 1.

<port>
Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<num>
Reference port number. ONLY specific ports are available to be a reference for each source port. Learn more.

<src>

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

Examples

The following sets source port "Port 1 Src2" on channel 2 to use "Port 1" as the phase reference
SOUR2:PHAS:PAR:PORT 1,"PORT 1 Src2"

The following sets source port 2 on channel 1 to use Port 1 as the phase reference
source:phas2:PAR:PORT 1

Query Syntax

SOURce<ch>:PHASe<port>:PARameter:PORT? [src]

Return Type

Character

Default

3

SOURce<ch>:PHASe<port>:PARameter[:VALue] <string>[,src]

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the ratioed receivers (parameter) to use for phase control.

Parameters

<ch>
Channel number of phase control measurement. If unspecified, value is set to 1.
Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

Ratioed parameter. Choose any two VNA physical receivers. Use either standard receiver notation ("R/R3") or logical receiver notation ("a1/a3").

Separate the two receiver names by a forward slash '/'. For example: "a3/a1".

**Note:** Phase control does not allow any two VNA physical receivers. Can only choose from an option of ax/ay, ay/ax, or ax/bx, where x is the control port and y is the reference port. DIQ application allows any two VNA physical receivers.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

**Examples**

```
SOUR:PHAS2:PAR: "a3/a1","Port 3"
source2:phase:parameter:value "a3/a1","Port 3"
```

**Query Syntax**

```
SOURce<ch>:PHASe<port>:PARameter[:VALue]?
```

Returns the ratioed parameter name.

**Return Type**

String

**Default**

"a1/b1"

---

**SOURce<ch>:PHASe<port>:POFFset:CORRection:DATA <data>[,src]**

Applicable Models: N522xB, N524xB

*(Read-Write)* Write and read a ratio amplitude offset array. This allows the setting of arbitrary impedance, which is used for active load applications.

**Parameters**

- `<ch>`: Channel number of phase control measurement. If unspecified, value is set to 1.
- `<port>`: Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<num>`: Ratio amplitude offset data array.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use **SOUR:CAT?** to return a list of valid port names.

### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SOUR:PHAS2:POFF:CORR:DATA</strong></td>
<td></td>
</tr>
<tr>
<td>source2:phase:poffset:correction:data ,&quot;Port 1 Src2&quot;</td>
<td></td>
</tr>
</tbody>
</table>

### Return Type

Depends on **FORMat:DATA**

### Default

Not Applicable

**SOURce<ch>:PHASE<port>:POFFset:CORRection[:STATe] <bool>[,src]**

Applicable Models: N522xB, N524xB

(Read-Write) Write and read whether to use the ratio amplitude offset array.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Channel number of phase control measurement. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;port&gt;</td>
<td>Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.</td>
</tr>
<tr>
<td>&lt;bool&gt;</td>
<td>Phase correction array state.</td>
</tr>
<tr>
<td></td>
<td>ON (1) Apply ratio amplitude offset array.</td>
</tr>
<tr>
<td></td>
<td>OFF(0) Do NOT apply ratio amplitude offset array.</td>
</tr>
</tbody>
</table>

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use **SOUR:CAT?** to return a list of valid port names.

### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SOUR:PHAS2:POFF:CORR 1</strong></td>
<td></td>
</tr>
</tbody>
</table>
source2:phase:poffset:correction:state OFF,"Port 1 Src2"

**Query Syntax**

SOURce<ch>:PHASe<port>:POFFset:CONTrol[:STATe]?

**Return Type**

Boolean

**Default**

OFF

---

SOURce<ch>:PHASe<port>:POFFset:FIXed <num>[,src]

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Write and read the fixed power ratioed value. Must NOT be in power sweep to use this value during phase control.

**Parameters**

<ch> Channel number of phase control measurement. If unspecified, value is set to 1.

<port> Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<num> Fixed power ratio value within the allowable range of the VNA.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

**Examples**

SOUR:PHAS2:POFF:FIX 1

source2:phase:poffset:fixed -5,"Port 1 Src2"

**Query Syntax**

SOURce<ch>:PHASe<port>:POFFset:FIXed? [src]

**Return Type**

Numeric

**Default**

0 dBc

---

SOURce<ch>:PHASe<port>:POFFset:STARt <num>[,src]

**Applicable Models:** N522xB, N524xB

*(Read-Write)* Write and read the start power ratioed value. Must also send SENS:SWE:TYPE POWER to put the analyzer into power sweep mode.
### Parameters

- `<ch>`
  - Channel number of phase control measurement. If unspecified, value is set to 1.

- `<port>`
  - Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.

- `<num>`
  - Start power ratio value in dBc. Must be within the allowable range of the VNA

- `[src]`

  While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, `[src]` takes priority.

  Use `SOUR:CAT?` to return a list of valid port names.

### Examples

```
SOUR:PHAS2:POFF:STAR 0
source2:phase:poffset:start -5,"Port 1 Src2"
```

### Query Syntax

```
SOURce<ch>:PHASE<port>:POFFset:STARt? [src]
```

### Return Type

Numeric

### Default

0 dBc

---

**`SOURce<ch>:PHASE<port>:POFFset:STOP <num>[,src]`**

Applicable Models: N522xB, N524xB

(Read-Write) Write and read the start power ratioed value. Must also send `SENS:SWE:TYPE POWer` to put the analyzer into power sweep mode.

### Parameters

- `<ch>`
  - Channel number of phase control measurement. If unspecified, value is set to 1.

- `<port>`
  - Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.

- `<num>`
  - Stop power ratio value in dBc. Must be within the allowable range of the VNA

- `[src]`
While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

**Examples**

```
SOUR:PHAS2:POFF:STOP 0
source2:phase:poffset:stop -5,"Port 1 Src2"
```

**Query Syntax**

```SOURce<ch>:PHASe<port>:POFFset:STOP? [src]```

**Return Type**

Numeric

**Default**

0 dBc

---

**SOURce<ch>:PHASE<port>:REFERENCE:CATalog?**

Applicable Models: N522xB, N524xB

(Read-only) Returns the available ports that can be used as phase control reference ports for the phase controlled port.

**Parameters**

- `<ch>`
  - Channel number of phase control measurement. If unspecified, value is set to 1.

- `<port>`
  - Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

- [src]

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

**Examples**

```
SOUR:PHAS2:REF:CAT?
source2:phase:reference:catalog?
```

**Return Type**

String of comma-separated phase control reference ports.

**Default**

Not Applicable
SOURce<ch>:PHASe<port>:REFerence:PORT <num>[,src]

Applicable Models: N522xB, N524xB

(Read-Write) Sets and returns the reference port for the Phase Control measurement.

**Parameters**

- `<ch>`: Channel number of phase control measurement. If unspecified, value is set to 1.
- `<port>`: Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<num>`: Reference port number. ONLY specific ports are available to be a reference for each source port. Learn more. Use the SOURce:PHASE:REFerence:CATalog? command to see which ports are available.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

**Examples**

The following sets source port "Port 1 Src2" on channel 2 to use "Port 1" as the phase reference.
SOUR2:PHAS:REF:PORT 1,"PORT 1 Src2"

The following sets source port 2 on channel 1 to use Port 1 as the phase reference.
source:phas2:REF:PORT 1

**Query Syntax**

SOURce<ch>:PHASe<port>:REFerence:PORT? [src]

**Return Type**

Character

**Default**

3

-------------------

SOURce<ch>:PHASe<port>:STARt <num>[,src]

Applicable Models: N522xB, N524xB

(Read-Write) Write and read the start value of phase sweep. Must also send SENS:SWE:TYPE PHASE to put the analyzer into phase sweep mode.

**Parameters**

- `<ch>`
- `<port>`
- `<num>`
- `[src]`
<ch>
Channel number of phase control measurement. If unspecified, value is set to 1.

<port>
Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<num>
Start phase value in degrees. Choose a value between -360 and 360.

[src]

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

Examples
SOUR:PHAS2:STAR 60
source2:phase:start 120,"Port 1 Src2"

Query Syntax
SOURce<ch>:PHASe<port>:STARt? [src]

Return Type
Numeric

Default
0 degrees

SOURce<ch>:PHASe<port>:STOP <num>[,src]

Applicable Models: N522xB, N524xB
(Read-Write) Write and read the stop value of phase sweep. Must also send SENS:SWE:TYPE PHASE to put the analyzer into phase sweep mode.

Parameters

<ch>
Channel number of phase control measurement. If unspecified, value is set to 1.

<port>
Phase controlled port number. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<num>
Stop phase value in degrees. Choose a value between -360 and 360.

[src]

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the
<port> argument performs the same function. If both arguments are specified, [src] takes priority.

Use SOUR:CAT? to return a list of valid port names.

### Examples

```plaintext
SOUR:PHAS2:STOP 60
source2:phase:stop 120,"Port 1 Src2"
```

### Query Syntax

SOURce<ch>:PHASe<port>:STOP? [src]

### Return Type

Numeric

### Default

0 degrees
SourceRxLeveling
SOURce:POWer:ALC:MODE:RECeiver Commands

Controls Receiver Leveling.

**SOUR:POW:ALC[:MODE]:**

RECeiver:
ACquisition:
  - MODE
  - FAST
  - FTYPe
  - IFBW

ITERation
  - ENABle
  - VALue

LSPC
MODulation
  - APERture
    - OFFSet
  - SPAN
  - [:STATe]
  - BANDwidth
    - NOISe

OFFSet
RATio?
REFerence
SAFE[:STATe]
  - MAX
  - MIN
STEP
[:STATE] TOLERance

Click on a keyword to view the command details.

**See Also**
- Example Programs
- Learn about Receiver Leveling
- Synchronizing the Analyzer and Controller
- SCPI Command Tree
- Remotely Specifying a Source Port

### SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:ACQuisition:MODE <char>, [src]

**Applicable Models:** N522xB, N523xB, N524xB

*(Read-Write)* Sets all ports to pre-sweep or point leveling mode for the specified channel. A channel cannot be set to both pre-sweep and point leveling. For point leveling, the source power is adjusted per point until the leveling receiver reports that the power is within tolerance (or the maximum iteration setting is reached). When the iteration is done, it moves immediately to the next point. This function is enabled using the SOURce:POWer:ALC[:MODE]:RECeiver:ITERation:ENABle command.

Learn more about Receiver Leveling

**Parameters**

- `<ch>`
  - *Channel number to be receiver leveled. If unspecified, value is set to 1*

- `<port>`
  - *Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.*

- `<char>`
  - *Acquisition mode. Choose from:*
    - PRESweep - Leveling sweeps are performed in the background (not visible) before every measurement sweep to measure and apply source correction data.
    - POINt - Source power is adjusted per point.

- `[src]`

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the
<port> argument performs the same function. If both arguments are specified, [src] takes priority.

### Examples

```
SOUR:POW:ALC:REC:ACQ:MODE PRES
source2:power2:alc:mode:receiver:acquisition:mode presweep
```

See ReceiverLeveling example

### Query Syntax

```
SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:ACQuisition:MODE? [src]
```

### Return Type

Character

### Default

PRESweep

---

**SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:FAST <bool>, [src]**

**Applicable Models:** N522xB, N523xB, N524xB

**(Read-Write)** Sets and returns the state of a separate IFBW setting for leveling sweeps. ON allows a higher (faster) IFBW than the measurement sweep. It also causes leveling sweeps to be noisier.

Learn more about Receiver Leveling

### Parameters

**<ch>**

Channel number to be receiver leveled. If unspecified, value is set to 1

**<port>**

Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

**<bool>**

Separate IFBW setting state.

ON or 1 - Separate IFBW setting. Specify IFBW using SOUR:POW:ALC:MODE:REC:IFBW

OFF or 0 - Same IFBW as the measurement sweep. Specify IFBW using Sens:BWID

**[src]**


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

### Examples

```
SOUR:POW:ALC:REC:FAST 1
source2:power2:alc:mode:receiver:fast off
source:power:alc:mode:receiver:fast off,"Port 1 Src2"
```
See ReceiverLeveling example

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Return Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce&lt;ch&gt;:POWer&lt;port&gt;:ALC[:MODE]:RECeiver:FAST? [src]</td>
<td>Boolean</td>
<td>ON</td>
</tr>
</tbody>
</table>

**SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:FTYPe <char>, [src]**

Applicable Models: N522xB, N523xB, N524xB

*(Read-Write)* Sets and returns the frequency range to use for receiver leveling. On the user interface, this is the "Receiver frequency is determined by:" setting.

Learn more about Receiver Leveling

**Parameters**

- `<ch>`
  - Channel number to be receiver leveled. If unspecified, value is set to 1

- `<port>`
  - Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

- `<char>`
  - Frequency range. Choose from:
    - AUTO - always uses the frequency range that is assigned to the measurement receiver.
    - INPut - Mixer/Converter input frequency range.
    - OUTPut - Mixer/Converter input frequency range.
    - RECeiver - FOM Receiver frequency range.
    - SOURce - FOM Source frequency range

- [src]

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

- `SOUR:POW:ALC:REC:FTYP AUTO`
- `source2:power2:alc:mode:receiver:ftype input`
- `source:power:alc:mode:receiver:ftype output,"Port 1 Src2"`

See ReceiverLeveling example
SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:IFBW <value>, [src]

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets and returns the IFBW to be used for leveling sweeps. Enable separate IFBW for leveling sweeps using SOUR:POW:ALC:MODE:REC:FAST 1

Learn more about Receiver Leveling

Parameters

<ch> Channel number to be receiver leveled. If unspecified, value is set to 1

<port> Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<value> IFBW for leveling sweeps in Hz. The list of valid IF Bandwidths is different depending on the VNA model. (Click to see the lists.) If an invalid number is specified, the analyzer will round up to the closest valid number.

This parameter supports MIN and MAX as arguments. Learn more.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

SOUR:POW:ALC:REC:IFBW 100E3
source2:power2:alc:mode:receiver:ifbw 100khz
source:power:alc:mode:receiver:ifbw 70e3,"Port 1 Src2"

See ReceiverLeveling example

Query Syntax SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:IFBW? [src]

Return Type Numeric

Default 100 kHz
SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:ITERation:ENABle <bool>, [src]

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Enables or disables the receiver leveling maximum iteration search function set using the SOURce:POWer:ALC[:MODE]:RECeiver:ACQuisition:MODE. The maximum iterations are set using the SOURce:POWer:ALC[:MODE]:RECeiver:ITERation:VALue command.

Learn more about Receiver Leveling

Parameters

- **<ch>**
  - Channel number to be receiver leveled. If unspecified, value is set to 1

- **<port>**
  - Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

- **<bool>**
  - Safe mode state.
    - ON or 1 - Enable receiver leveling maximum iteration search.
    - OFF or 0 - Disable receiver leveling maximum iteration search. The prior value for maximum iterations is retained.

- **[src]**

  While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW:ALC:REC:ITER:ENAB 1
source2:power2:alc:mode:receiver:iteration:enable on
source:power:alc:mode:receiver:iteration:enable off,"Port 1 Src2"
```

See ReceiverLeveling example

Query Syntax

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:ITERation:VALue <value>, [src]

Return Type

Boolean

Default

ON
Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets and returns the maximum iterations to be used in order to achieve the tolerance setting.

Learn more about Receiver Leveling

Parameters

<ch>  
Channel number to be receiver leveled. If unspecified, value is set to 1

<port>  
Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<value>  
Max iterations. Choose a value 0 to 50.

Note: Entering a value of 0 is allowed and sets the receiver leveling to Prior-Sweep mode. Learn more.

[src]  

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

source2:power2:alc:mode:receiver:iteration:value 10  
source:power:alc:mode:receiver:iteration:value 7,"Port 1 Src2"

See ReceiverLeveling example

Query Syntax

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:ITERation:VALue? [src]

Return Type

Numeric

Default

10

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:LSPC <bool>, [src]

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets and returns the state of Use Last Result for Source Power Cal. When Leveling Mode is switched back to Internal, this feature turns Source Power Cal correction ON using the latest receiver leveling correction data.

Learn more about Receiver Leveling
Parameters

<ch>  
Channel number to be receiver leveled. If unspecified, value is set to 1

<port>  
Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<bool>  
State of Use Last Result for Source Power Cal.
ON or 1 - When Leveling Mode is switched back to Internal, Source Power Cal correction is turned ON using the latest receiver leveling correction data.
OFF or 0 - When Leveling Mode is switched back to Internal, Source Power Cal correction is NOT turned ON.

[src]  

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples
SOUR:POW:ALC:REC:LSPC 1
source2:power2:alc:mode:receiver:lspc off
source:power:alc:mode:receiver:lspc off,"Port 1 Src2"

See ReceiverLeveling example

Query Syntax
SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:LSPC? [src]

Return Type
Boolean

Default
OFF

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:MODulation:APERture:OFFSet <value>, [src]

Applicable Models: N524xB with S93070xB, and/or N522xB, N523xB, N524xB with S9x09xA/B, S9x090A/B, M980xA/P50xxA with 090/190 + S9x090B and/or S95070xA/B (M980xA only)

(Read-Write) Sets and returns the offset of the frequency aperture used to measure the signal power. Since the ideal modulation signal is known, the total power is calculated from this value. The SOURce:POWer:ALC:MODE:RECeiver:MODulation:APERture:STATe must be set to ON to enable this setting.

Learn more about Receiver Leveling

Parameters
Channel number to be receiver leveled. If unspecified, value is set to 1

Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

Receiver leveling aperture offset in Hz.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples


Query Syntax

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:MODulation:APERture:OFFSet? [src]

Return Type

Numeric

Default

0

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:MODulation:APERture:SPAN <value>, [src]

Applicable Models: N524xB with S93070xB, and/or N522xB, N523xB, N524xB with S9x09xxA/B, S9x090A/B, M980xA/P50xxA with 090/190 + S9x090B and/or S95070xA/B (M980xA only)

(Read-Write) Sets and returns the span of the frequency aperture used to measure the signal power. Since the ideal modulation signal is known, the total power is calculated from this value. The SOURce:POWer:ALC:MODE:RECeiver:MODulation:APERture:STATe must be set to ON to enable this setting.

Learn more about Receiver Leveling

Parameters

<ch>

Channel number to be receiver leveled. If unspecified, value is set to 1

<port>

Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<value>

Receiver leveling aperture span in Hz.

[src]

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

source2:power2:alc:mode:receiver:modulation:aperture:span
10e6,"MXG_Vector"

**Query Syntax**

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:MODulation:APERture:SPAN? [src]

**Return Type**

Numeric

**Default**

10e6

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:MODulation:APERture[:STATe] <bool>, [src]

Applicable Models: N524xB with S93070xB, and/or N522xB, N523xB, N524xB with S9x09xxA/B, S9x090A/B, M980xA/P50xxA with 090/190 + S9x090B and/or S95070xA/B (M980xA only)

*(Read-Write)* Sets and returns the state of the receiver leveling aperture settings for the specified source for Modulation Distortion measurements. Enabling the aperture settings will measure the power more quickly by reducing the span of the measurement. Enter the span and offset of the frequency aperture used to measure the signal power. Since the ideal modulation signal is known, the total power is calculated from this value.

**Learn more about Receiver Leveling**

**Parameters**

- `<ch>`
  Channel number to be receiver leveled. If unspecified, value is set to 1

- `<port>`
  Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

- `<bool>`
  State of receiver leveling aperture settings.
  ON or 1 - Enable span and offset aperture settings.
  OFF or 0 - Disable span and offset aperture settings.

- `[src]`

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the
<port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
SOUR:POW:ALC:REC:MOD:APER 1
source2:power2:alc:mode:receiver:modulation:aperture:state 1,"MXG_Vector"
```

**Query Syntax**

```
SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:MODulation:APERture[:STATe]? [src]
```

**Return Type**

Boolean

**Default**

OFF

---

**SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:MODulation:BANDwidth:NOISe <value>, [src]**

Applicable Models: N524xB with S93070xB, and/or N522xB, N523xB, N524xB with S9x09xxA/B, S9x090A/B, M980xA/P50xxA with 090/190 + S9x090B and/or S95070xA/B (M980xA only)

*(Read-Write)* Sets and returns the receiver noise bandwidth value for the specified source for Modulation Distortion measurements. Noise bandwidth is equal to the Resolution bandwidth divided by the Vector Average factor. The `SOURce:POWer:ALC:MODE:RECeiver:FAST` command must be set to ON to enable this setting.

Learn more about Receiver Leveling

**Parameters**

- `<ch>`
  - Channel number to be receiver leveled. If unspecified, value is set to 1

- `<port>`
  - Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

- `<value>`
  - Noise bandwidth value in Hz.

- `[src]`

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
SOUR:POW:ALC:REC:MOD:BAND:NOIS 1e3
```

**Query Syntax**

```
SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:MODulation:BANDwidth:NOISe? [src]
```
SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:OFFSet <value>, [src]

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets and returns the power level offset value.

Learn more about Receiver Leveling

Parameters

- `<ch>`: Channel number to be receiver leveled. If unspecified, value is set to 1
- `<port>`: Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<value>`: Power level offset in dB. Choose a value between +200 and -200.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW:ALC:REC:OFFS 10
source2:power2:alc:mode:receiver:offset 5
source:power:alc:mode:receiver:offset 7,"Port 1 Src2"
```

See ReceiverLeveling example

Query Syntax

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:RATio? [src]

Return Type

Numeric

Default

0

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:RATio? [src]

Applicable Models: N522xB, N523xB, N524xB
(Read-only) Returns the receiver ratio to be used with receiver leveling. This receiver ratio parameter is the same as the one set in \texttt{SOUR:PHAS:PARameter}.

Learn more about Receiver Leveling

**Parameters**

- \texttt{<ch>}\hspace{1cm} Channel number to be receiver leveled. If unspecified, value is set to 1
- \texttt{<port>}\hspace{1cm} Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- [src]\hspace{1cm} String. (NOT case sensitive). Source port. Optional. Use \texttt{SOUR:CAT?} to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an \textit{external source}, \textit{true mode balanced port}, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the \texttt{<port>} argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

\texttt{SOUR:POW:ALC:REC:RAT 10}
\texttt{source2:power2:alc:mode:receiver:ratio "a1/a3,3"}
\texttt{source:power:alc:mode:receiver:ratio "R1/R3,3","Port 1 Src2"}

See ReceiverLeveling example

**Query Syntax**

\texttt{SOURce<ch>:POWer<port>:ALC[MODE]:RECeiver:RATio? [src]}

Returned is two VNA physical receivers that are controlled by different sources and the port that is 'paired' with \texttt{<port>}. The receivers and paired port are separated by a comma.

**Return Type**\hspace{1cm} String

**Default**\hspace{1cm} "a1/a3,3"

---

\texttt{SOURce<ch>:POWer<port>:ALC[MODE]:RECeiver:REFerence <rec>, [src]}

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets and returns the reference receiver to be used with Receiver Leveling.

Learn more about Receiver Leveling

**Parameters**

- \texttt{<ch>}\hspace{1cm} Channel number to be receiver leveled. If unspecified, value is set to 1
**<port>**

Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

**<rec>**

(String) VNA receiver. Choose the VNA physical receiver that works with the source port <port>.

For example: `<port 1> = "R1"

**[src]**


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
SOUR:POW:ALC:REC:REF 'r1'
source2:power2:alc:mode:receiver:reference 'r2'
source:power:alc:mode:receiver:reference "r1","Port 1 Src2"
'Read the last setting back
source:power:alc:mode:receiver:reference? "Port 1 Src2"
'Returns:
"R1,1"
```

*See ReceiverLeveling example*

**Query Syntax**

`SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:REFerence? [src]`

**Return Type**

String - Name of the reference receiver.

**Default**

Not Applicable

---

**SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:SAFE[:STATe] <bool>, [src]**

Applicable Models: N522xB, N523xB, N524xB

*(Read-Write)* Sets and returns the state of Safe Mode.

*Learn more about Receiver Leveling*

**Parameters**

**<ch>**

Channel number to be receiver leveled. If unspecified, value is set to 1

**<port>**

Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
### <bool>

Safe mode state.

**ON or 1 - Safe mode ON**

**OFF or 0 - Safe mode OFF**

### [src]


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
SOUR:POW:ALC:REC:SAFE 1
source2:power2:alc:mode:receiver:safe on
source:power:alc:mode:receiver:safe:state off,"Port 1 Src2"
```

*See Receiver Leveling example*

**Query Syntax**

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:SAFE[:STATe]? [src]

**Return Type**

Boolean

**Default**

OFF

---

**SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:SAFE:MAX <value>, [src]**

**Applicable Models:** N522xB, N523xB, N524xB

(Read-Write) Sets and returns the maximum power level for Safe Mode.

The MAX/MIN limit is always used regardless of the safe mode state. In addition, the MAX/MIN limit is for port power and related to power offset. If the power offset is not set correctly, the MAX/MIN limit is not correct and it may impact the leveling. Ensure that the power offset in the channel is the same as power offset during calibration. If the exact power offset is not known, choose a limit for source and then it will not be related to power offset.

*Learn more about Receiver Leveling*

**Parameters**

- **<ch>**
  
  Channel number to be receiver leveled. If unspecified, value is set to 1

- **<port>**
  
  Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
**Example**

```
SOUR:POW:ALC:REC:SWE:MAX 10
source2:power2:alc:mode:receiver:safe:max 20
source:power:alc:mode:receiver:safe:max 15,"Port 1 Src2"
```

See ReceiverLeveling example

---

### Parameters

- `<ch>`
  - Channel number to be receiver leveled. If unspecified, value is set to 1

- `<port>`
  - Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the `[src]` argument.

- `<value>`
  - Minimum power level in dB.
**SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:SAFE:MIN -50**
source2:power2:alc:mode:receiver:safe:min -80
source:power:alc:mode:receiver:safe:min -40,"Port 1 Src2"

See ReceiverLeveling example

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>SOURce&lt;ch&gt;:POWer&lt;port&gt;:ALC[:MODE]:RECeiver:SAFE:MIN? [src]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Numeric</td>
</tr>
<tr>
<td>Default</td>
<td>-95 dB</td>
</tr>
</tbody>
</table>

**SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:SAFE:STEP <value>, [src]**

Applicable Models: N522xB, N523xB, N524xB

*(Read-Write)* Sets and returns the maximum step power level for Safe Mode.

Learn more about Receiver Leveling

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Channel number to be receiver leveled. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;port&gt;</td>
<td>Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.</td>
</tr>
<tr>
<td>&lt;value&gt;</td>
<td>Maximum Step power level in dB.</td>
</tr>
</tbody>
</table>

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.
Examples

SOUR:POW:ALC:REC:SAFE:STEP 2
source2:power2:alc:mode:receiver:safe:step 1.5
source:power:alc:mode:receiver:safe:min 2, "Port 1 Src2"

See ReceiverLeveling example

Query Syntax
SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:SAFE:S

Return Type
Numeric

Default
1 dB

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver[:STATe] <bool>, [src]

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets and returns the state of Receiver Leveling for the specified source port.

Learn more about Receiver Leveling

Parameters

<ch>  
Channel number to be receiver leveled. If unspecified, value is set to 1

<port>  
Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<bool>  
Receiver Leveling state.
ON or 1 - Receiver Leveling ON
OFF or 0 - Receiver Leveling OFF

[src]  

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

SOUR:POW:ALC:REC 1
source2:power2:alc:mode:receiver on
source:power:alc:mode:receiver:state off, "Port 1 Src2"

See ReceiverLeveling example

Query Syntax
SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver[:STATe]? [src]
Return Type: Boolean
Default: OFF

**SOURce<ch>:POWer<port>:ALC[:MODE]:REceiver:TOLerance <value>, [src]**

Applicable Models: N522xB, N523xB, N524xB

(Read-Write) Sets and returns the tolerance value for leveling sweeps.

Learn more about Receiver Leveling

**Parameters**

<ch>  Channel number to be receiver leveled. If unspecified, value is set to 1

<port>  Source port being used for Receiver Leveling. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<value>  Tolerance level in dB.


While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, true mode balanced port, or one of the Source outputs on the 2-port PNA-X model with multiple sources such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```
SOUR:POW:ALC:REC:TOL .01
source2:power2:alc:mode:receiver:tolerance .5
source:power:alc:mode:receiver:tolerance .2,"Port 1 Src2"
```

See ReceiverLeveling example

**Query Syntax**

SOURce<ch>:POWer<port>:ALC[:MODE]:RECeiver:TOLerance? [src]

**Return Type** Numeric

**Default** .1 dB
Status Keywords SCPI
Status Command Keywords

The following keywords can be appended to the node or nodes that represent the Status register you want to control.

:CONDition?
:ENABLE
:ENABLE?
:EVENT?
:MAP
:NTRansition
:PTRansition

Learn about Status Registers
SCPI Command Tree

:CONDition?
Monitors the conditions as they occur REAL TIME. That is, a condition may occur, and then clear before the condition is read. Reading this register returns a 16-bit decimal weighted number.

:ENABLE <bit>
Enables register bits that will monitored using the service request (SRQ) method. (To use the direct read method, you do not have to enable the bit.)

Default value for STATus:QUEStionable:ENABLE and STATus:OPERation:ENABLE is 0: No bits enabled.

Default value for all other registers :ENABLE <bits> is 32767; ALL BITS ENABLED.

Therefore it is ONLY necessary to send the ENABLE keyword if you want to DISABLE some conditions. For example, to enable ONLY Trace1 (bit 2) of the LIMIT1 register (disable all other traces), send:
STATus:QUEStionable:LIMIT1:ENABLE 4

:ENABLE?
Read the enable register to verify the bits that you enabled. Returns a 16 bit weighted sum of the bits that are enabled.
[:EVENt]?

Query only - This is the Default keyword for most registers. Use it to determine if a condition has occurred. These bits remain set until they are read or otherwise cleared.

---

:MAP <bit>,<error>

 Associates a bit is the User register with an error number. For example

```
STATus:QUESTionable:DEFine:USER2:MAP 0,-113
```

0 is the bit that will be set

-113 is the error

When error -113 "Undefined Header" occurs, bit 0 in the USER2 register will be set to 1.

---

:NTRansition <bits>

Write-Read - Negative Transition register bits set the condition to be set on the Negative going (True to False) transition. Use this register if you are only interested in a condition changing from True to False.

:NTRansition?

queries the register to verify that you set a negative transition.

---

:PTRansition <bits>

Write-Read - Positive Transition register bits set the condition to be set on the False to True transition. Use this register if you are only interested in the change of a condition from False to True.

:PTRansition?

Queries the register to verify that you set a positive transition.
# TDR Display

**DISPlay:TDR Commands**

These commands control the TDR display setup.

<table>
<thead>
<tr>
<th>DISPlay:TDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>EYE</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMAGe</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>MEASure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MINimize:STATe</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>SCALe</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VIEW</th>
</tr>
</thead>
</table>

| X                    |
|                      |
|                     |
| | SCALe               |
Click on a red keyword to view the command details.

**see Also**

- Synchronizing the Analyzer and Controller
- SCPI Command Tree

---

**DISPlay:TDR:EYE:Y:SCALe:AUTO:STATe <bool>**

**Applicable Models:** All with TDR Options (S9x011A/B)

**(Read-Write)** Turns the continuous auto-scale mode for the eye y-axis ON or OFF.

**Parameters**

- `<bool>`
  - ON or 1 - Turns continuous auto-scale ON.
  - OFF or 0 - Turns continuous auto-scale OFF.
Examples

DISP:TDR:EYE:Y:SCAL:AUTO:STAT ON
display:tdr:eye:y:scale:auto:state off

Query Syntax

DISPLAY:TDR:EYE:SCALe:AUTO:STATe?

Return Type

Boolean

Default

ON

DISP:TDR:EYE:Y:SCAL:PDIVision <value>

Applicable Models: All with TDR Options (S9x011A/B)
(Read-Write) This command sets the value of the y-axis scale per division for eye diagram.

Parameters

<value>
Value of eye diagram y-axis scale per division. The range is 1E-18 to 5.

Examples

DISP:TDR:EYE:Y:SCAL:PDIV 300E-03
display:tdr:eye:y:scale:pdivision 300e-03

Query Syntax

DISPLAY:TDR:EYE:Y:SCALe:PDIVision?

Return Type

Double

Default

200m

DISP:TDR:EYE:Y:SCAL:RLEVel <value>

Applicable Models: All with TDR Options (S9x011A/B)
(Read-Write) This command sets the value of the eye diagram y-axis reference line.

Parameters

<value>
Value of eye diagram y-axis reference line. The range is -5 to +5.

Examples

DISP:TDR:EYE:Y:SCAL:RLEV 0.01
display:tdr:eye:y:scale:rlevel 0.01

Query Syntax

DISPLAY:TDR:EYE:Y:SCALe:RLEVel?

Return Type

Double

Default

0
**Display:TDR:Eye:Y:Scale:RPosition <value>**

Applicable Models: All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the value of the eye diagram y-axis reference position.

**Parameters**

*<value>* Value of eye diagram y-axis reference position. The range is 0 to 10.

**Examples**

```
 DISPLAY:TDR:EYE:Y:SCALE:RPOSITION 10
 display:tdr:eye:y:scale:rposition 10
```

**Query Syntax**

```
 DISPLAY:TDR:EYE:Y:SCALE:RPOSITION?
```

**Return Type** Integer

**Default** 0

---

**Display:TDR:Image <enum>**

Applicable Models: All with TDR Options (S9x011A/B)

*(Read-Write)* This command changes the background color of the screen.

**Parameters**

*<enum>* Screen background color. Choose from:

- **NORMAL** - Black background color.
- **INVert** - White background color.

**Examples**

```
 DISPLAY:TDR:IMAGE NORM
 display:tdr:image normal
```

**Query Syntax**

```
 DISPLAY:TDR:IMAGE?
```

**Return Type** String

**Default** NORMal

---

**Display:TDR:Measure[1-16]:DMemory:Type <enum>**

Applicable Models: All with TDR Options (S9x011A/B)

*(Read-Write)* This command sets the display to off, data type, memory type, or data and memory type.

**Parameters**

*<enum>* Data/memory display. Choose from:

- **OFF** - Nothing is displayed on the graph plot.
- **DATA** - Data only is displayed on the graph plot.

**Examples**

```
 DISPLAY:TDR:MEASURe:DMEMOrY:TYPE DATA
 display:tdr:measure:dmemory:type data
```
MEMory - Memory only is displayed on the graph plot.
DMEMory - Data and Memory are displayed on the graph plot.

Examples
DISP:TDR:MEAS1:DMEM:TYPE OFF
display:tdr:measure1:dmemory:off

Query Syntax
DISPlay:TDR:MEASure[1-16]:DMEMory:TYPE?

Return Type
String

Default
DATA

DISPlay:TDR:MEASure[1-16]:X:SCALe:AUTO

Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) This command executes x-axis auto scaling.

Examples
DISP:TDR:MEAS1:X:SCAL:AUTO
display:tdr:measure1:x:scale:auto

DISPlay:TDR:MEASure[1-16]:X:SCALe:PDIVision <value>

Applicable Models: All with TDR Options (S9x011A/B)
(Read-Write) This command sets the value of the x-axis scale per division.

Parameters

<value> Value of x-axis scale per division.

Examples
DISP:TDR:MEAS1:X:SCAL:PDIV 1E-9
display:tdr:measure1:scale:pdivision 1e-9

Query Syntax
DISPlay:TDR:MEASure[1-16]:X:SCALe:PDIVision?

Return Type
Double

Default
2n

DISPlay:TDR:MEASure[1-16]:X:SCALe:RLEVel <value>

Applicable Models: All with TDR Options (S9x011A/B)
(Read-Write) This command sets the value of the x-axis reference line.

Parameters

<value> Value of x-axis reference line.
Examples

| DISPlay:TDR:MEAS1:X:SCALe:RLEV 20E-9 |
| display:tdr:measure1:x:scale:rlevel 20e-9 |

Query Syntax

| DISPlay:TDR:MEASure[1-16]:X:SCALe:RLEVel? |

Return Type

| Double |

Default

| 10n |

---

**DISPlay:TDR:MINimize:STATe <bool>**

Applicable Models: All with TDR Options (S9x011A/B)

*(Read-Write)* Sets or gets the minimize state.

**Parameters**

- `<bool>`
  - ON or 1 - Turns minimize state ON.
  - OFF or 0 - Turns minimize state OFF.

**Examples**

| DISPlay:TDR:MIN:STAT ON |
| display:tdr:minimize:state off |

**Query Syntax**

| DISPlay:TDR:MINimize:STATe? |

**Return Type**

| Boolean |

**Default**

| OFF |

---

**DISPlay:TDR:SCALe:AUTO**

Applicable Models: All with TDR Options (S9x011A/B)

*(Write-only)* This command executes y-axis auto scaling.

**Examples**

| DISPlay:TDR:SCAL:AUTO |
| display:tdr:scale:auto |

---

**DISPlay:TDR:VIEW <enum>**

Applicable Models: All with TDR Options (S9x011A/B)

*(Read-Write)* This command selects the view point for waveform analysis either before or after the DUT.

**Parameters**

- `<enum>`
  - X-axis reference position. Choose from:
STIMulus - Stimulus view, observation point before DUT.
RESPonse - Response view, observation point after DUT.

Examples
DISP:TDR:VIEW STIM
display:tdr:view stimulus

Query Syntax
DISPlay:TDR:VIEW?

Return Type
String
Default
STIMulus

DISPlay:TDR:X:SCALe:RPOSition <enum>

Applicable Models: All with TDR Options (S9x011A/B)

(Read-Write) This command sets the x-axis reference position for the time domain measurement.

Parameters
<enum> X-axis reference position. Choose from:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>Reference position is the left edge.</td>
</tr>
</tbody>
</table>

Examples
DISP:TDR:SCAL:RPOS LEFT
display:tdr:scale:rposition left

Query Syntax
DISPlay:TDR:SCALe:RPOSition?

Return Type
String
Default
LEFT
## TDR Memory

### MMEMory:TDR Commands

These commands control the loading and storing of eye bit pattern and mask files.

<table>
<thead>
<tr>
<th>MMEMory:TDR</th>
<th>LOAD</th>
<th>STORE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EYE</td>
<td>EYE</td>
</tr>
<tr>
<td></td>
<td>BPATtern</td>
<td>BPATtern</td>
</tr>
<tr>
<td></td>
<td>MASK</td>
<td>MASK</td>
</tr>
<tr>
<td></td>
<td>STATE</td>
<td>STATE</td>
</tr>
</tbody>
</table>

Click on a red keyword to view the command details.

### See Also

- **Synchronizing the Analyzer and Controller**
- **SCPI Command Tree**

### MMEMory:TDR:LOAD:EYE:BPATtern <filename>

**Applicable Models:** All with TDR Options (S9x011A/B)

(Write-only) This command loads the specified user bit pattern file. The extension of file should be .txt. The bit pattern editing is not available through the command.
Parameters
<filename>

File name of the user bit pattern (.txt)

Examples
MMEM:TDR:LOAD:EYE:BPAT "C:\TDR\mybitpattern.txt"
memory:tdr:load:eye:bpattern "c:\tdr\mybitpattern.txt"

Default
Not Applicable

**MMEMory:TDR:LOAD:EYE[:MASK] <filename>**

Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) This command loads eye-mask file. The format of the eye mask file should be the same as the format of the Infiniium DCA (86100C). The extension of the file should be .msk. The MASK pattern editing is not available through the command.

Parameters
<filename>

File name of the eye mask (.msk)

Examples
MMEM:TDR:LOAD:EYE:MASK "C:\TDR\FC0133.msk"
memory:tdr:load:eye:mask "c:\tdr\FC0133.msk"

Default
Not Applicable

**MMEMory:TDR:LOAD:STATe <filename>**

Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) Loads the specified instrument state file (.sta).

Parameters
<filename>

String - Name of any valid file that does not already exist.

Examples
MMEM:TDR:LOAD:STAT 'myState'
memory:tdr:load:state 'c:\tdr\myState.sta'

Query Syntax
Not applicable

Default
Not applicable

**MMEMory:TDR:STORe:EYE:BPATtern <filename>**

Applicable Models: All with TDR Options (S9x011A/B)
(Write-only) This command stores the user bit pattern file. The extension of file should be .txt. The bit pattern editing is not available through the command.

Parameters

<filename>  
File name of the user bit pattern (.txt)

Examples

MMEM:TDR:STOR:EYE:BPAT "C:\TDR\mybitpattern.txt"
memory:tdr:store:eye:bpattern "c:\tdr\mybitpattern.txt"

Default  
Not Applicable

**MMEMory:TDR:STORe:EYE[:MASK] <filename>**

Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) This command stores the eye-mask file. The format of the eye mask file should be the same as the format of the Infinium DCA (86100C). The extension of the file should be .msk. The MASK pattern editing is not available through the command.

Parameters

<filename>  
File name of the eye mask (.msk)

Examples

MMEM:TDR:STOR:EYE:MASK "C:\TDR\mymask.msk"
memory:tdr:store:eye:mask "c:\tdr\mymask.msk"

Default  
Not Applicable

**MMEMory:TDR:STORe:FDATa <filename>**

Applicable Models: All with TDR Options (S9x011A/B)

(Write-only) Stores the specified measurement data file of the active trace.

Parameters

<filename>  
String - Name of any valid file that does not already exist.

Examples

MMEM:TDR:STOR:FDATa 'myFdata'
memory:tdr:store:fdata 'c:\tdr\myFdata'

Query Syntax  
Not applicable

Default  
Not applicable
### MMEMory:TDR:STORe:FDATa:ALL <filename>

**Applicable Models:** All with TDR Options (S9x011A/B)

**(Write-only)** Stores the specified measurement data file for all traces.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;filename&gt;</td>
<td>String - Name of any valid file that does not already exist.</td>
</tr>
</tbody>
</table>

**Examples**

```plaintext
MMEM:TDR:STOR:FDAT:ALL 'myFdata'  
memory:tdr:store:fdata:all 'c:\tdr\myFdata'
```

**Query Syntax**

Not applicable

**Default**

Not applicable

### MMEMory:TDR:STORe:SNP <n>

**Applicable Models:** All with TDR Options (S9x011A/B)

**(Write-only)** Stores the SnP measurement data. Learn more about SnP data.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;n&gt;</td>
<td>SnP measurement data to store. If unspecified, &lt;n&gt; is set to 2. The number you specify must be less than or equal to the number of available ports on the VNA. Choose from: 1 (S1P) stores 1-Port data for the active measurement if the active measurement is a reflection parameter such as S11 or S22. The behavior is UNDEFINED if the active measurement is a transmission parameter such as an S21. 2 (S2P) stores data for the four 2 port parameters associated with the current measurement. Default. Data that is not available is zero-filled. 3 (S3P) stores data for the nine 3 port parameters associated with the current measurement. Data that is not available is zero-filled. 4 (S4P) stores data for the sixteen 4 port parameters associated with the current measurement. Data that is not available is zero-filled.</td>
</tr>
</tbody>
</table>

**Examples**

```plaintext
MMEM:TDR:STOR:SNP 1  
memory:tdr:store:snp 1
```

**Query Syntax**

Not applicable

**Default**

Not applicable
MMEMory:TDR:STOR:STATe <filename>

Applicable Models: All with TDR Options (S9x011A/B)
(Write-only) Stores the specified instrument state file (.sta).

Parameters

<filename>  String - Name of any valid file that does not already exist.

Examples

- MMEM:TDR:STOR:STAT 'myState'
- mmemory:tdr:store:state 'c:\tdr\myState.sta'

Query Syntax

Not applicable

Default

Not applicable
TDR Source
SOURce:TDR Commands

SOURce:TDR
POWer
  | LEVel
  | IMMediate
  | AMPLitude

Click on a red keyword to view the command details.

See Also

Synchronizing the Analyzer and Controller
SCPI Command Tree

SOURce<cnum>:TDR:POWer[:LEVel][:IMMediate][:AMPLitude] <value>

Applicable Models: All with TDR Options (S9x011A/B)
(Read-Write) This command sets the source power level.

Parameters

<cnum> Channel number of the measurement. If unspecified, <cnum> is set to 1.

<value> Source power level in dBm.

Examples

SOUR:TDR:POW 20
source:tdr:power:level:immediate:amplitude 20

Query Syntax

SOURce:TDR:POWer[:LEVel][:IMMediate][:AMPLitude]?

Return Type

Double

Default

-8 dBm
Cal All for Mixer Channel

*CLS
SYST:FPReset
'------------------------------------------
' create a standard channel
'------------------------------------------
DISPlay:WINDow1:STATE ON
CALC1:PARameter:DEFine:EXT "MyMeas", S21
DISPlay:WINDow1:TRACe1:FEED "MyMeas"
SENSe1:BANDwidth 700
SENSe1:FREQuency:CENTer 1ghz
SENSe1:FREQuency:SPAN 1ghz
SENSe1:SWEep:POINts 11
'------------------------------------------
' create a Mixer channel
'------------------------------------------
DISPlay:WINDow2:STATE ON
CALC2:CUST:DEF 'My SC21', 'Scalar Mixer/Converter', 'SC21'
DISP:WIND2:TRAC:FEED 'My SC21'
SENS2:SWEep:POINts 11
SENS2:BANDwidth 1e3
SENS2:MIX:INPUT:FREQ:MODE SWEPt
SENS2:MIX:INPUT:FREQ:STAR 3.6e9
SENS2:MIX:INPUT:FREQ:STOP 3.9e9
SENS2:MIX:LO:FREQ:MODE FIXED
SENS2:MIX:LO:FREQ:FIX 1.0e9
SENS2:MIX:LO:POW 10
SENS2:MIX:OUTP:FREQ:SID LOW
SENS2:MIX:CALC Output
SENS2:MIX:LO:NAME 'Port 3'
SENS2:MIX:APPLY
'------------------------------------------
' configure cal all
' select channels 1 and 2.
' for channel 2 select all the ports of the mixer
' explicitly because we want to add the LO (port 3)
'------------------------------------------
SYST:CAL:ALL:RESet
SYST:CAL:ALL:SEL 1,2
SYST:CAL:ALL:CHAN2:PORT:SEL 1,2,3
SYST:ERR?
SYST:CAL:ALL:CHAN2:PORT:SEL?
SYST:CAL:ALL:IFBW 1e3
SYST:CAL:ALL:PORT1:SOUR:POWer -10
SYST:CAL:ALL:CSET:PREFix 'MyCalAllExample'

' query for the available mixer cal properties to set
' this is an info only query


' Enable phase correction using a cal mixer with known delay

SYST:CAL:ALL:MCL:PROP:VAL 'Enable Phase Correction','true'
SYST:CAL:ALL:MCL:PROP:VAL 'Phase Correction Method','Use Mixer Delay'
SYST:CAL:ALL:MCL:PROP:VAL 'Mixer Delay', '10e-9'

' configure power sensor

SYST:COMM:PSEN USB, "Agilent Technologies,U8485A,my53470003"

' Perform calibration
' Note: the channel number used in the following SENS
' commands MUST BE the channel number returned
' from the preceding query SYST:CAL:ALL:GUIDed:CHAN?

SYST:CAL:ALL:GUIDed:CHAN?
SENS200:corr:coll:guid:conn:port1 "APC 3.5 male"
SENS200:corr:coll:guid:conn:port2 "APC 3.5 female"
SENS200:corr:coll:guid:conn:port3 "Not used"
SENS200:corr:coll:guid:conn:port4 "Not used"
SENS200:corr:coll:guid:ckit:port1 "N4691-60004 ECal"
SENS200:corr:coll:guid:ckit:port2 "N4691-60004 ECal"
SENS200:corr:coll:guid:init; *OPC?
The channel number used for the SENSe header is determined by the \texttt{SYST:CAL:ALL:GUID:CHAN?} command. You must query this channel number – do not assume that it will always be a particular value.
Cal All Independent Calibration Channels

Set pna = CreateObject("AgilentPNA835x.Application")
Set scpi = pna.ScpiStringParser
scpi.Parse "SYST:PRESET"
scpi.Parse "SYST:CAL:ALL:RES"
scpi.Parse "SYST:CAL:ALL:GUID:CHAN?"
scpi.Parse "SYST:CAL:ALL:SEL?"
'The following SCPI will request Channel 1 to do its own calibration:
scpi.Parse "SYST:CAL:ALL:CHAN1:PORT 1,2"
scpi.Parse "SYST:CAL:ALL:CHAN1:PORT:SEL?"
scpi.Parse "SENS200:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 male'"
scpi.Parse "SENS200:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 male'"
scpi.Parse "SYST:CAL:ALL:GUID:PORT?"
'Even though Ch1 is performing its own calibration, interface through CalAll is the same:"
scpi.Parse "SYST:CAL:ALL:GUID:CHAN?"
scpi.Parse "SENS200:CORR:COLL:GUID:INIT"
scpi.Parse "SENS200:CORR:COLL:GUID:STEP?"
scpi.Parse "SENS200:CORR:COLL:GUID:ACQ STAN1;"
scpi.Parse "SENS200:CORR:COLL:GUID:ACQ STAN3;"
scpi.Parse "SENS200:CORR:COLL:GUID:ACQ STAN4;"
scpi.Parse "SENS200:CORR:COLL:GUID:ACQ STAN5;"
scpi.Parse "SENS200:CORR:COLL:GUID:ACQ STAN7;"
scpi.Parse "SENS200:CORR:COLL:GUID:ACQ STAN8;"

The channel number used for the SENSe header is determined by the SYST:CAL:ALL:GUID:CHAN? command. You must query this channel number – do not assume that it will always be a particular value.
Cal All Multi-Channel Independent Calibration

Channels

Set pna = CreateObject("AgilentPNA835x.Application")
Set scpi = pna.ScpiStringParser
scpi.Parse "SYST:PRESET"
scpi.Parse "calc:par:del:all"
scpi.Parse "calc1:cust:def 'CompS11','Gain Compression Converters','CompS11'"
scpi.Parse "calc2:cust:def 'S11','Standard','S11'"
scpi.Parse "calc3:cust:def 'b1','Spectrum Analyzer','b1'"
scpi.Parse "calc4:cust:def 'S11_1','Standard','S11'"
scpi.Parse "calc5:cust:def 'S22','Standard','S22'"
scpi.Parse "calc6:cust:def 'S44','Standard','S44'"
scpi.Parse "display:wind1:trace1:feed 'CompS11'"
scpi.Parse "display:wind1:trace2:feed 'S11'"
scpi.Parse "display:wind1:trace3:feed 'b1'"
scpi.Parse "display:wind1:trace4:feed 'S11_1'"
scpi.Parse "display:wind1:trace5:feed 'S22'"
scpi.Parse "display:wind1:trace6:feed 'S44'"
scpi.Parse "sens2:sweep:lfex:state ON"
scpi.Parse "sens4:sweep:lfex:state ON"
scpi.Parse "sens6:sweep:lfex:state ON"
scpi.Parse "+OPC?"
scpi.Parse "SYST:CAL:ALL:RES"
scpi.Parse "SYST:CAL:ALL:GUID:CHAN?"
scpi.Parse "SYST:CAL:ALL:SEL?"
'Identifying independent power calibration channels:
scpi.Parse "SYST:CAL:ALL:CHAN1:PORT 1,2"
scpi.Parse "SYST:CAL:ALL:CHAN2:PORT 1,2"
scpi.Parse "SYST:CAL:ALL:CHAN3:PORT 1,2"
scpi.Parse "SYST:CAL:ALL:CHAN4:PORT 1,2,4"
scpi.Parse "SENS200:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 male'"
scpi.Parse "SENS200:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 male'"
scpi.Parse "SENS200:CORR:COLL:GUID:CONN:PORT4 'APC 3.5 male'"
The channel number used for the SENSe header is determined by the `SYST:CAL:ALL:GUID:CHAN?` command. You must query this channel number – do not assume that it will always be a particular value.
The channel number used for the SENSe header is determined by the `SYST:CAL:ALL:GUID:CHAN?` command. You must query this channel number – do not assume that it will always be a particular value.
CalAll_2-Port_1-Chan_ECal

```plaintext
, ' calibrate 2 ports, no power, ECal ,

system:preset;
SYST:CAL:ALL:RESet
SYST:CAL:ALL:SEL 1
SYST:CAL:ALL:CHAN1:PORT:SEL 1,2
SYST:CAL:ALL:MCL:PROP:VAL "Include Power Calibration","false"
SYST:CAL:ALL:GUID:CHAN?
SENS200:corr:coll:guid:conn:port1 "APC 3.5 female"
SENS200:corr:coll:guid:conn:port2 "APC 3.5 male"
SENS200:corr:coll:guid:ckit:port1 "N4691-60004 ECal"
SENS200:corr:coll:guid:ckit:port2 "N4691-60004 ECal"
SENS200:CORR:COLL:GUID:INIT
SENS200:CORR:COLL:GUID:STEPS?
SENS200:corr:coll:guid:acq stan1
SENS200:corr:coll:guid:save
```

The channel number used for the SENSe header is determined by the `SYST:CAL:ALL:GUID:CHAN?` command. You must query this channel number – do not assume that it will always be a particular value.
CalAll_2-Port_1-Chan_PwrCal_ECal
Cal All - 2-Port, 1-Channel, Power Cal, with ECal

' calibrate 2 ports, power cal, ECal
'
system:preset;
CALC:PAR:DEF "S11", S11
CALC2:PAR:DEF "S22", S22
SYST:CAL:ALL:RESet
SYST:CAL:ALL:SEL 1,2
SYST:CAL:ALL:CHAN1:PORT:SEL 1,2
SYST:CAL:ALL:MCL:PROP:VAL "Include Power Calibration","true"

SYST:CAL:ALL:GUID:CHAN?
SENS200:CORR:COLL:GUID:PSEN1:CONN 'Ignored'
SENS200:CORR:COLL:GUID:PSEN1:CKIT 'Not used'
SENS200:corr:coll:guid:conn:port1 "APC 3.5 male"
SENS200:corr:coll:guid:conn:port2 "APC 3.5 female"
SENS200:corr:coll:guid:ckit:port1 "N4691-60004 ECal"
SENS200:corr:coll:guid:ckit:port2 "N4691-60004 ECal"
SENS200:CORR:COLL:GUID:INIT
SENS200:CORR:COLL:GUID:STEPS?
SENS200:corr:coll:guid:acq stan1
SENS200:corr:coll:guid:acq stan2
SENS200:corr:coll:guid:save

The channel number used for the SENSe header is determined by the SYST:CAL:ALL:GUID:CHAN? command. You must query this channel number – do not assume that it will always be a particular value.
CalAll_2-Port_2-Chan_PwrCal_ECal
Cal All - 2-Port, 2-Channel, Power Cal, with ECal

' calibrate 2 ports, 2 channels, power cal, ECal
'
system:preset;
SYST:COMM:PSEN USB, "Agilent Technologies,U8485A,my53470003"
CALC:PAR:DEF "S11", S11
CALC2:PAR:DEF "S22", S22
SYST:CAL:ALL:RESet
SYST:CAL:ALL:SEL 1,2
SYST:CAL:ALL:CHAN1:PORT:SEL 1
SYST:CAL:ALL:CHAN2:PORT:SEL 1,2
SYST:CAL:ALL:MCL:PROP:VAL "Include Power Calibration", "true"
SYST:CAL:ALL:GUID:CHAN?
SENS200:CORR:COLL:GUID:PSEN1:CONN 'Ignored'
SENS200:CORR:COLL:GUID:PSEN1:CKIT 'Not used'
SENS200:corr:coll:guid:conn:port1 "APC 3.5 male"
SENS200:corr:coll:guid:conn:port2 "APC 3.5 female"
SENS200:corr:coll:guid:ckit:port1 "N4691-60004 ECal"
SENS200:corr:coll:guid:ckit:port2 "N4691-60004 ECal"
SENS200:CORR:COLL:GUID:INIT
SENS200:CORR:COLL:GUID:STEPs?
SENS200:corr:coll:guid:acq stan1
SENS200:corr:coll:guid:acq stan2
SENS200:corr:coll:guid:save

The channel number used for the SENSe header is determined by the SYST:CAL:ALL:GUID:CHAN? command. You must query this channel number – do not assume that it will always be a particular value.
CalAll_IMD

'----------------------------------------
' CHANNEL 1: IMD
'----------------------------------------
disp:wind:state on
CALC:CUST:DEF 'PwrMain1', "Swept IMD", "PwrMain"
DISP:WIND:TRAC1:FEED 'PwrMain1'
SENS:SWE:POIN 11
SENS:FREQ:START 4E9
SENS:FREQ:STOP 8E9
SENS:IMD:FREQ:DFR:CW 2E6
CALC:PAR:CAT?
'----------------------------------------
' CAHNNEL 2:  imd
'----------------------------------------
DISP:WIND2:STATE ON
CALC2:CUST:DEF 'PwrMain2', "Swept IMD", "PwrMain"
SYST:ERR?
DISP:WIND2:TRAC1:FEED 'PwrMain2'
SENS2:SWE:POIN 21
SENS2:FREQ:START 18E9
SENS2:FREQ:STOP 20E9
SENS2:IMD:FREQ:DFR:CW 100E3
SYST:ERR?
'----------------------------------------
' configure the power sensor for cal
'----------------------------------------
SYST:COMM:PSEN USB, "Agilent Technologies,U8485A,my53470003"
'----------------------------------------
' configure calibrate all
'----------------------------------------
SYST:CAL:ALL:CSET:PREFIX "imdcalall"
SYST:CAL:ALL:RESET
SYST:CAL:ALL:SEL 1,2
SYST:CAL:ALL:CHAN1:PORTS:SEL 1,2
SYST:CAL:ALL:CHAN2:PORTS:SEL 1,2
'----------------------------------------
' we want to calibrate for 2nd order products on channel 1
' but not on channel 2. Here's how to do it.
'----------------------------------------
The channel number used for the SENSe header is determined by the SYST:CAL:ALL:GUID:CHAN? command. You must query this channel number – do not assume that it will always be a particular value.
CalAll_Noise Figure

'--------------------------------------------------------
' create NF channel
'--------------------------------------------------------
SYST:FPRESET
DISP:WIND:STATE ON
CALC1:MEAS1:DEF "S11:Noise Figure Cold Source"
DISP:MEAS1:FEED 1
CALC1:MEAS2:DEF "NF:Noise Figure Cold Source"
DISP:MEAS2:FEED 1

'--------------------------------------------------------
' configure power sensor
'--------------------------------------------------------

system:communicate:psensor usb, "Agilent Technologies,U8485A,MY53470003"

'--------------------------------------------------------
' configure calibrate all
'--------------------------------------------------------
SYST:CAL:ALL:RESET
SYST:CAL:ALL:CSET:PREFIX "Example"
SYST:CAL:ALL:SEL 1
SYST:CAL:ALL:CHAN1:PORTS:SEL 1,2

'--------------------------------------------------------
' use this query to see what cal all properties are
' relevant to the noise figure channel
'--------------------------------------------------------
SYST:CAL:ALL:MCLASS:PROP:NAME:CAT? "Noise Figure Cold Source"

'--------------------------------------------------------
' set scalar noise cal, using power meter
'--------------------------------------------------------

'--------------------------------------------------------
' retrieve the guided cal channel number
'--------------------------------------------------------
SYST:CAL:ALL:GUIDED:CHAN?

' configure the sensor
' set ignored unless you want to calibrate
' an adapter used for the power sensor.
SENS200:CORR:COLL:GUID:PSEN1:CONN 'Ignored'
SENS200:CORR:COLL:GUID:PSEN1:CKIT 'Not used'

' configure basic cal properties: connectors, kits
' NOTE: always fully specify ecals when Noise figure
' is in the channel list. IE: include the ecal serial
' number!

SENS200:CORR:COLL:GUID:CONN:PORT1:SEL "APC 3.5 male"
SENS200:CORR:COLL:GUID:CONN:PORT2:SEL "APC 3.5 female"
SENS200:CORR:COLL:GUID:ckit:port1:SEL "N4691-60004 ECal 02297"
SENS200:CORR:COLL:GUID:INIT

' acquire the calibration
SENS200:CORR:COLL:GUID:STEPS?
SENS200:CORR:COLL:GUID:DESC? 1
SENS200:CORR:COLL:GUID:ACQ STAN1
SENS200:CORR:COLL:GUID:ACQ STAN2
SENS200:CORR:COLL:GUID:ACQ STAN3
SENS200:CORR:COLL:GUID:SAVE

The channel number used for the SENSE header is determined by the SYST:CAL:ALL:GUID:CHAN? command. You must query this channel number – do not assume that it will always be a particular value.
'------------------------------------------
' create a standard channel
'------------------------------------------
DISPlay:WINDow1:STATE ON
CALC1:PARameter:DEFine:EXT "MyMeas", S21
DISPlay:WINDow1:TRACe1:FEED "MyMeas"
SENSe1:BANDwidth 700
SENSe1:FREQuency:CENTer 1ghz
SENSe1:FREQuency:SPAN 1ghz
SENSe1:SWEep:POINts 11
'------------------------------------------
' create a Mixer channel
'------------------------------------------
DISPlay:WINDow2:STATE ON
CALC2:CUST:DEF 'My SC21', 'Scalar Mixer/Converter', 'SC21'
DISP:WIND2:TRAC:FEED 'My SC21'
SENSe2:SWEep:POINts 11
SENSe2:BANDwidth 1e3
SENSe2:MIX:INPut:FREQ:MODE SWEPt
SENSe2:MIX:INPut:FREQ:STAR 3.6e9
SENSe2:MIX:INPut:FREQ:STOP 3.9e9
SENSe2:MIX:LO:FREQ:MODE FIXED
SENSe2:MIX:LO:FREQ:FIX 1.0e9
SENSe2:MIX:LO:POW 10
SENSe2:MIX:OUTP:FREQ:SID LOW
SENSe2:MIX:CALC Output
SENSe2:MIX:LO:NAME 'Port 3'
SENSe2:MIX:APPLY
'------------------------------------------
' configure cal all
'------------------------------------------
SYST:CAL:ALL:RESet
SYST:CAL:ALL:SEL 1,2
SYST:CAL:ALL:IFBW 1e3
SYST:CAL:ALL:PORT1:SOUR:POWer -10
SYST:CAL:ALL:CSET:PREFix 'MyCalAllExample'
'------------------------------------------
' query for the available mixer cal properties to set
' this is an info only query

' Enable phase correction using a cal mixer with known delay
SYST:CAL:ALL:MCL:PROP:VAL 'Enable Phase Correction','true'
SYST:CAL:ALL:MCL:PROP:VAL 'Phase Correction Method','Use Mixer Delay'
SYST:CAL:ALL:MCL:PROP:VAL 'Mixer Delay', '10e-9'

' configure power sensor
SYST:COMM:PSEN USB, "Agilent Technologies,U8485A,my53470003"

' Perform calibration
SYST:CAL:ALL:GUID:CHAN?
SENS200:corr:coll:guid:conn:port1 "APC 3.5 male"
SENS200:corr:coll:guid:conn:port2 "APC 3.5 female"
SENS200:corr:coll:guid:conn:port3 "Not used"
SENS200:corr:coll:guid:conn:port4 "Not used"
SENS200:corr:coll:guid:ckit:port1 "N4691-60004 ECal"
SENS200:corr:coll:guid:ckit:port2 "N4691-60004 ECal"
SENS200:corr:coll:guid:init
SENS200:corr:coll:guid:steps?
SENS200:corr:coll:guid:acq stan1
SENS200:corr:coll:guid:acq stan2
SENS200:corr:coll:guid:acq stan3
SENS200:CORR:COLL:GUID:SAVE

The channel number used for the SENSe header is determined by the SYST:CAL:ALL:GUID:CHAN? command. You must query this channel number – do not assume that it will always be a particular value.
Set pna = CreateObject("AgilentPNA835x.Application")
Set scpi = pna.ScpiStringParser
scpi.Parse "SYST:PRESET"
scpi.Parse "calc:par:del:all"
scpi.Parse "calc:cust:def 'sc12','Scalar Mixer/Converter','SC12'"
scpi.Parse "disp:wind:trac1:feed 'sc12'"
scpi.Parse "*OPC?"
scpi.Parse "SYST:CAL:ALL:RESet"
scpi.Parse "syst:cal:all:sel 1"
scpi.Parse "SYST:CAL:ALL:CSET:PREFix 'smcSplit'"
scpi.Parse "syst:cal:all:chan1:port:sel 1,2"
'Split Cal attribute MUST be sent before setting connectors & kits
scpi.Parse "SENS200:CORR:COLL:GUID:CHAN?"
scpi.Parse "SENS200:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 male'"
scpi.Parse "SENS200:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 male'"
scpi.Parse "SENS200:CORR:COLL:GUID:INIT"
scpi.Parse "SENS200:CORR:COLL:GUID:STEP?"
'Note that Step 1 is to connect Power Sensor to Port 1
'And Step 2 is to connect Power Sensor to Port 2
'Performing two 1-Port calibrations instead of a 2-port calibration
scpi.Parse "SENS200:CORR:COLL:GUID:ACQ STAN1;"
scpi.Parse "SENS200:CORR:COLL:GUID:ACQ STAN3;"
scpi.Parse "SENS200:CORR:COLL:GUID:ACQ STAN4;"
scpi.Parse "SENS200:CORR:COLL:GUID:ACQ STAN5;"
scpi.Parse "SENS200:CORR:COLL:GUID:ACQ STAN7;"
The channel number used for the SENSe header is determined by the `SYST:CAL:ALL:GUID:CHAN?` command. You must query this channel number – do not assume that it will always be a particular value.
Dim app
Set app = CreateObject("Agilentpna835x.application","hostname")
Set scpi = app.ScpiStringParser
scpi.Parse("SYST:PRESET")
scpi.Parse("calc:par:del:all")
scpi.Parse("calc:cust:def "ENR","Noise Figure Cold Source","ENR"")
scpi.Parse("disp:wind:trac1:feed "ENR""")
scpi.Parse("SENSe:FREQuency:STOP 8.5E+9")
scpi.Parse("SYST:CAL:ALL:RESet")
scpi.Parse("syst:cal:all:sel 1")
scpi.Parse("syst:cal:all:chan1:port:sel 1,2")
scpi.Parse("SENS200:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 male'")
scpi.Parse("SENS200:CORR:COLL:GUID:INIT")
scpi.Parse("SENS200:CORR:COLL:GUID:ACQ STAN1")
scpi.Parse("SENS200:CORR:COLL:GUID:ACQ STAN2")
scpi.Parse("SENS200:CORR:COLL:GUID:ACQ STAN3")
scpi.Parse("SENS200:CORR:COLL:GUID:ACQ STAN4")
scpi.Parse("SENS200:CORR:COLL:GUID:ACQ STAN5")
scpi.Parse("SENS200:CORR:COLL:GUID:ACQ STAN6")
scpi.Parse("SENS200:CORR:COLL:GUID:ACQ STAN7")
scpi.Parse("SENS200:CORR:COLL:GUID:ACQ STAN8")
Configure an External Source

This VB Script program configures an External Source.

Learn more about External Source Configuration

These programs can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as ExtSource.vbs. Learn how to setup and run the macro.

See all External Device Configuration commands

```
' Get the VNA application, and
' start the scpi parser, and preset the VNA
dim app
Set app = CreateObject("AgilentPNA835x.Application")
set scpi = app.ScpiStringParser
scpi.parse "*rst"
'Configure the external source
scpi.parse "Syst:conf:edev:add 'newSource'"
scpi.parse "Syst:conf:edev:dtype 'newSource', 'Source'"
scpi.parse "Syst:conf:edev:driver 'newSource', 'AGMXG'"
scpi.parse "Syst:conf:edev:ioconfig 'newSource', 'gpib0::16::instr'"
'Activate and enable the external source
scpi.parse "Syst:conf:edev:ioen 'newSource', 1"
'State activates and talks to the device if "Syst:conf:edev:ioen" is enabled
scpi.parse "Syst:conf:edev:stat 'newSource', 1"
```
Create a Pulse Profile Measurement

The following SCPI example demonstrates how to create a Pulse Profile measurement using the Integrated Pulse Application on the PNA-X and N522xB.

Four measurements are created:

S11 (default)
S21
B receiver
R1 receiver - used to verify pulse gen settings.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as PulseProfile.vbs. Learn how to setup and run the macro.

See Also

Integrated Pulse Application
SCPI commands used in the program.
SCPI commands to control the internal pulse generators
Other Pulse SCPI examples

```
' VNA application object
Dim oPNA
' Channel 1 object
Dim chan1

' Create / Get the VNA application.
Set oPNA = CreateObject("AgilentPNA835x.Application")
Dim scpi
Set scpi = oPNA.ScpiStringParser
scpi.parse "syst:preset"
scpi.parse "CALCulate:PARameter:DEFine:EXT 'MyMeas1',S21"
scpi.parse "DISPlay:WINDow1:TRACe2:FEED 'MyMeas1'"
scpi.parse "CALCulate:PARameter:DEFine:EXT 'MyMeas2',R_1"
scpi.parse "DISPlay:WINDow1:TRACe3:FEED 'MyMeas2'"
scpi.parse "CALCulate:PARameter:DEFine:EXT 'MyMeas3',B_1"
scpi.parse "DISPlay:WINDow1:TRACe4:FEED 'MyMeas3'"

'Pulse settings
scpi.parse "SENSe:SWEep:PULSe:MODE PROFILE"
scpi.parse "SENSe:SWEep:PULSe:DRIVE 1"
scpi.parse "SENSe:SWEep:PULSe:PRF 1"
scpi.parse "SENSe:SWEep:PULSe:TIMing 1"
```
scpi.parse "SENSe:SWEep:PULSe:DETectmode 1"
scpi.parse "SENSe:SWEep:PULSe:IFBW 1"
scpi.parse "SENSe:SWEep:PULSe:MASTer:WIDth 1e-6"
scpi.parse "SENSe:SWEep:PULSe:MASTer:PERiod 1e-3"
Create_and_Cal_a_Diff_IQ_Measurement

Create a Differential I/Q Measurement

This example program creates a Differential I/Q measurement setup.

This VBScript program can be run as a macro in the VNA. To do this, copy the code into a text editor file such as Notepad and save on the VNA hard drive as DIQ.vbs. Learn how to setup and run the macro.

See the Differential I/Q commands.

See Other SCPI Example Programs

' This section gets the VNA application
' starts the scpi parser, and presets the VNA
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi= app.ScpiStringParser
scpi.Parse "SYST:FPR"
scpi.Parse "CALC:CUST:DEF 'mytrace','Differential I/Q','IPwrF1'"
scpi.Parse "DISP:WIND ON"
scpi.Parse "DISP:WIND:TRAC:FEED 'mytrace'"
scpi.Parse "SENS:DIQ:FREQ:RANG:ADD"
scpi.Parse "SENS:DIQ:FREQ:RANG1:STARt 1e9"
scpi.Parse "SENS:DIQ:FREQ:RANG1:STOP 2e9"
scpi.Parse "SENS:DIQ:FREQ:RANG2:COUP:STATe ON"
scpi.Parse "SENS:DIQ:FREQ:RANG2:COUP:ID 1"
scpi.Parse "SENS:DIQ:FREQ:RANG2:COUP:MULT 2"
scpi.Parse "SENS:DIQ:PORT1:RANG 1"
scpi.Parse "SENS:DIQ:PORT1:STATe ON"
scpi.Parse "SENS:DIQ:PORT1:MATC:CORR:STATe ON"
scpi.Parse "SENS:DIQ:PORT1:MATC:CORR:RANGe 'F1'"
scpi.Parse "SENS:DIQ:PORT1:MATC:CORR:TREC 'b3'"
scpi.Parse "SENS:DIQ:PORT1:MATC:CORR:RREC 'a3'"
scpi.Parse "SENS:DIQ:PORT1:POW:SWE:STATe OFF"
scpi.Parse "SENS:DIQ:PORT1:POW:STARt -5"
scpi.Parse "SENS:DIQ:PORT:POW:ATT 5"
scpi.Parse "SENS:DIQ:PORT:POW:ATT:AUTO ON"
Create and Cal a VMC Measurement

Note: This example program uses SCPI commands that were used BEFORE VNA release A.09.33. While these commands still work, we encourage you to use the newer VMC commands that were introduced with the A.09.33 release. See newer commands and example program.

This VB Script example creates and calibrates a Vector mixer measurement. To run this example without modification you need the following:

A Mixer setup file saved on the VNA: C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Mixer\MyMixer.mxr.

If the mixer file uses an external LO source, it must also be attached and configured.

An ECal module that covers the frequency range of the measurement.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example. However, some modification is necessary to make the program run on a traditional GPIB Interface.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as VMC.vbs. Learn how to setup and run the macro.

```vbs
Dim app
Dim scpi
' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
'---Create a Vector Mixer Measurement
'First, delete all measurements on the channel
scpi.Parse "CALC:PAR:DEL:ALL"
'Create a forward scalar mixer measurement and configure
'it in channel 1.
'The first parameter is a unique identifying string
'(specified by the user) to allow subsequent
'commands to be directed at this specific measurement.
scpi.Parse "CALC:CUST:DEF 'My VC21', 'Vector Mixer/Converter', 'VC21'"
'Setup the new measurement as the 2nd trace in the active window
scpi.Parse "DISP:WIND:TRAC2:FEED 'My VC21'"
'Make the new trace the active measurement
scpi.Parse "CALC:PAR:SEL 'My VC21'"
'The parameters of the mixer measurement can now be configured.
'This can be done by either using the SENS:MIX commands
'for each of the parameters or by loading a mixer setup file.
'This example loads a mixer setup file. The path name
'for the mixer file may be loaded from other mapped drives.
scpi.Parse "SENS:MIXer:Load 'c:\\users\\public\\network analyzer\\documents\\Mixer\\MyMixer.mxr'"

'---------------------Perform A Vector Mixer Calibration---------------------
'Initialize an VMC guided calibration for session number 6
scpi.Parse "SENS:CORR:COLL:SESS6:INIT ""VMC"""
'This sets the VMC operation to full system cal as opposed to
'performing a mixer characterization only.
```
This example uses ECal for the 2-port cal portion of the procedure. To use a mechanical kit you will have to use the following command:
```
```
If you select the mechanical method then you also have to specify the connector types and the cal kits for each of the ports.

The comments below show an example of how that is done:
```
' scpi.Parse "SENS:CORR:COLL:SESS6:CONN:PORT1:SEL "APC 3.5 male"
```

Choose the between ECal or Mechanical calibration for the Mixer Characterization portion of the VMC cal.
```
```
Specify the ECal module and the ECal characterization for the two port calibration portion of this session. FCA calibrations currently only support ECal module number 1. In this example the factory characterization is used by specifying 0 for the characterization number.
```
scpi.Parse "SENS:CORR:COLL:SESS6:VMC:TWOP:ECAL:CHAR 1,0"
```
Specify the thru measurement method. This applies to both ECal and mechanical calibrations. For ECal 'DEFAULT' will use the ECal thru. Other choices may be used depending on the genders and types of the connectors on the test interface.
```
```
Omit the isolation part of the 2-port cal.
```
```
Tell the wizard to generate and report the number of steps in this cal session.
```
Dim steps
Dim desc
'Determine the number of steps required to complete the calibration.
' First send the write command, then the query.
scpi.Parse "SENS:CORR:COLL:SESS6:STEP"
For i = 1 To steps
'Display the prompt for each step
MsgBox (desc)
'Perform the measurement for each step
scpi.Parse "SENS:CORR:COLL:SESS6:ACQ " & CStr(i)
Next
'Finish the cal and save the calset

Msgbox("VMC Cal Complete!")
Create and Cal an SMC Measurement

Note: This example program uses SCPI commands that were used BEFORE VNA release A.09.33. While these commands still work, we encourage you to use the newer SMC commands that were introduced with the A.09.33 release. See newer commands and example program.

This Visual Basic example creates and calibrates a scalar mixer measurement.

To run this example without modification you need the following:

A Mixer setup file saved on the VNA: C:\Program Files(86)\Keysight\Network Analyzer\Documents\Mixer\MyMixer.mxr.

If the mixer file uses an external LO source, it must also be attached and configured.

An ECAL module that covers the frequency range of the measurement.

A power meter must be attached to the VNA. If this example is run in the VNA, the power meter does not need to be attached using a GPIB/USB interface.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example. However, some modification is necessary to make the program run on a traditional GPIB Interface. For example, during the power meter portion of this calibration, scpi.Parse will not process a command until the power meter routine has completed. Traditional GPIB would require a serial polling technique to ensure the routine has completed before proceeding.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as SMC.vbs. Learn how to setup and run the macro.

```
Dim app
Dim scpi
' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
'---Create a Scalar Mixer Forward Measurement
'First, delete all measurements on the channel
scpi.Parse "CALC:PAR:DEL:ALL"
'Create a forward scalar mixer measurement and configure it in
'channel 1. The first parameter is a unique
'identifying string (specified by the user) to allow subsequent
'commands to be directed at this specific measurement.
scpi.Parse "CALC:CUST:DEF 'My SC21', 'Scalar Mixer/Converter', 'SC21'"

'Setup the new measurement as the 2nd trace in the active window
scpi.Parse "DISP:WIND:TRAC2:FEED 'My SC21'
'Make the new trace the active measurement
scpi.Parse "CALC:PAR:SEL 'My SC21'"
'The parameters of the mixer measurement can now be configured.
'This can be done by either using the individual SENS:MIX commands
'for each of the parameters or by loading a mixer setup file. This
'exmaple loads a mixer setup file. The path name
'for the mixer file may be loaded from other mapped drives
scpi.Parse "SENS:MIXer:Load ""C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Mixer\MyMixer.mx"
'------------------Perform A Scalar Mixer Calibration------------------
'Initialize an SMC guided calibration for session number 6
```
Select to use an ECal for the 2-port cal portion of the procedure.
To use a mechanical kit you will have to use the following command:

If you select the mechanical method then you also have to specify the connector types and the cal kits for each of the ports. The comments below show an example of how that is done:
'scrii.Parse "SENS:CORR:COLL:SESS6:CONN:PORT1:SEL ""APC 3.5 male"

Specify the ECal module and the ECal characterization for this session. FCA calibrations currently only support ECal module number 1. In this example the factory characterization is used by specifying 0 for the characterization number.
'scrii.Parse "SENS:CORR:COLL:SESS6:SMC:ECAL:CHAR 1,0"

Specify the thru measurement method. This applies to both ECal and mechanical calibrations. For ECal 'DEFAULT' will use the ECal thru. Other choices may be used depending on the genders and types of the connectors on the test interface.

Omit the isolation part of the 2-port cal

Turn on auto orientation for the ECal

Tell the wizard to generate and report the number of steps in this cal session
Dim steps
Dim desc
'Determine the number of steps required to complete the calibration.
'First send the write command, then the query.
scrii.Parse "SENS:CORR:COLL:SESS6:STEP"
steps = scrii.Parse ("SENS:CORR:COLL:SESS6:STEP?"
For i = 1 To steps
'Display the prompt for each step
MsgBox (desc)
'Perform the measurement for each step
scrii.Parse "SENS:CORR:COLL:SESS6:ACQ " & CStr(i)
Next

Dim calset
'Finish the cal and save the calset
Msgbox ("SMC Cal Complete!")
Create SMC Embedded LO Measurement

This VB Script example creates a SC21 SMC measurement for a mixer with an Embedded LO.

To make an embedded LO measurement:

Set the LO frequency for each stage to the nominal frequency of the embedded LO. Set the LO source to "Not Controlled".

Calibrate as usual at the end of the measurement setup. To do this, copy the Calibration section from Create and Cal an SMC Measurement.

The following are the mixer settings:

Single-stage Fixed LO's - Uncomment the **Blue** lines to measure DUAL stage mixers.

Swept Input and Output

Set Input and LO frequencies - Calculate the Output frequencies

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as SMC.vbs. Learn how to setup and run the macro.

```vbnet
Dim app
Dim scpi
' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Parse "SYSTem:PRESet"
' Create a Scalar Mixer Forward Measurement
' First, delete all measurements on the channel
scpi.Parse "CALC:PAR:DEL:ALL"
' Create a forward scalar mixer measurement and configure it in
' channel 1. The first parameter is a unique
' identifying string (specified by the user) to allow subsequent
' commands to be directed at this specific measurement.
scpi.Parse "CALC:CUST:DEF 'My SC21', 'Scalar Mixer/Converter', 'SC21'"
' Setup the new measurement as the 2nd trace in the active window
scpi.Parse "DISP:WIND:TRAC2:FEED 'My SC21'"
' Make the new trace the active measurement
scpi.Parse "CALC:PAR:SEL 'My SC21'"

' Setup Stimulus - Points and IFBW are channel settings
scpi.Parse "SENS:SWEep:POINts 21"
```
scpi.Parse "SENS:BANDwidth 1e3"

'Perform single sweep

scpi.Parse "SENS:SWE:MODE SING:*OPC?"

'Mixer settings

' Input settings (swept)

scpi.Parse "SENS:MIX:INPut:FREQ:STAR 3.6e9"
scpi.Parse "SENS:MIX:INPut:FREQ:STOP 3.9e9"

' LO1 settings (fixed)

scpi.Parse "SENS:MIX:LO1:FREQ:MODE FIXED"
scpi.Parse "SENS:MIX:LO1:FREQ:FIX 500e6"
scpi.Parse "SENS:MIX:LO1:NAME 'Not controlled'"

'Dual-stage - LO2 settings (fixed)

' Uncomment these lines for dual stage mixer

'scpi.Parse "SENS:MIX:STAGE 2"
'scpi.Parse "SENS:MIX:LO2:FREQ:MODE FIXED"
'scpi.Parse "SENS:MIX:LO2:FREQ:FIX 500e6"
'scpi.Parse "SENS:MIX:LO2:NAME 'Not controlled'"

' Output settings (calculated)

scpi.Parse "SENS:MIX:OUTP:FREQ:SID LOW"
scpi.Parse "SENS:MIX:CALC Output"
scpi.Parse "SENS:MIX:APPLY"

' Changing the following default settings is usually not necessary

scpi.Parse "SENS:MIX:ELO:TUN:MODE BROadband"
scpi.Parse "SENS:MIX:ELO:TUN:IFBW 30e3"
scpi.Parse "SENS:MIX:ELO:TUN:INT 1"
scpi.Parse "SENS:MIX:ELO:TUN:ITER 5"
scpi.Parse "SENS:MIX:ELO:TUN:SPAN 3e6"
scpi.Parse "SENS:MIX:ELO:TUN:TOL 1"

' Enable embedded LO
scpi.Parse "SENS:MIX:ELO:STAT 1"

'Single sweep does the same as "Find Now"

' from the ELO dialog
scpi.Parse "SENS:SWE:MODE SING:*OPC?"

'Perform Calibration here as usual
Dynamic Uncertainty

This example does the following in separate subprograms:

- Setup Uncertainty Manager Workspace
- Perform a standard 2-port Guided Cal
- Perform a Cable Characterization
- Perform a Noise Characterization
- Perform a 2-Port Uncertainty Cal
- Make Uncertainty Trace Settings

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as *.vbs.

See Also

- Learn how to setup and run the macro.
- See all Dynamic Uncertainty commands
- Learn about Dynamic Uncertainty (Option S93015A/B)

### See Other SCPI Example Programs

```vbscript
Dim app
Dim scpi
' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
'Preset the analyzer
scpi.Parse "SYST:PRESet"
SetupWorkspace
TwoPortGuidedCal
PerformCableChar
PerformNoiseChar
TwoPortUncertCal
MakeTraceSettings
'*****************************************************************************
' Setup Uncertainty Workspace
',
```
Sub SetupWorkspace

cableCatalog = scpi.Parse("SYSTem:UNCertainty:CABLe:CATalog?"")
' Strip the leading and trailing quotation
' marks from the catalog string

cableCatalog = Mid(cableCatalog, 2, Len(cableCatalog) - 3)
' Tokenize the comma-delimited list string
' into an array of the individual substrings

cables = Split(cableCatalog, ",")
' Re-assemble the cable list into one string delimited by linefeed characters

cableCatalog = ""

For i = 0 To UBound(cables)
    cableCatalog = cableCatalog & cables(i) & Chr(10)
    ' Uncomment the following line to reset (clear) repeatability data for the cable
    ' scpi.Parse "SYSTem:UNCertainty:CABLe:REPeat:RESet " & cables(i) & ""

Next

MsgBox "Cables:" & Chr(10) & Chr(10) & cableCatalog

portCatalog = scpi.Parse("SOURce:CATalog?"")
' Strip the leading and trailing quotation
' marks from the catalog string

portCatalog = Mid(portCatalog, 2, Len(portCatalog) - 3)
' Tokenize the comma-delimited list string
' into an array of the individual substrings

ports = Split(portCatalog, ",")

portCatalog = ""

For i = 0 To UBound(ports)
    portNum = i + 1
    ' Ensuring we only count the actual testports and not any source-only ports like external sources
    If (ports(i) = "Port " & CStr(portNum)) Then
        cable = scpi.Parse("SYSTem:UNCertainty:PORT" & CStr(portNum) & ":CABLe:SELeCt?"")
        portCatalog = portCatalog & ports(i) & ": " & cable & Chr(10)
        ' Uncomment the following line to reset (clear) noise data for the port
        ' scpi.Parse "SYSTem:UNCertainty:PORT" & CStr(portNum) & ":NOISe:RESet"
    End If

Next

MsgBox "Port Cables:" & Chr(10) & Chr(10) & portCatalog
' Uncomment the following line to select the particular cable for all ports
' scpi.Parse "SYSTem:UNCertainty:PORT:CABLe:ALL:SELeCt " & cables(0) & ""
' Uncomment the following line to copy the noise data for Port 1 to be used for all ports
' scpi.Parse "SYSTem:UNCertainty:PORT:NOISe:ALL:COpy 1"
' Uncomment the following line to reset (clear) the noise data for all ports
' scpi.Parse "SYSTem:UNCertainty:PORT:NOISe:ALL:RESeT"
' The next lines toggle uncertainty manager properties on
scpi.Parse "SYSTem:UNCertainty:ETERm:NOISE:ENABLE ON"
MsgBox "Noise enabled for cals = " & scpi.Parse("SYSTem:UNCertainty:ETERm:NOISE:ENABLE?")
scpi.Parse "SYSTem:UNCertainty:ETERm:REPeat:ENABLE ON"
MsgBox "Cable repeatability enabled for cals = " &
scpi.Parse "SYSTem:UNCertainty:SDEFinitions:ENABLE ON"
MsgBox "Standards definition uncertainties enabled for cals = " &
' Uncomment the following line to change the max number of uncertainty points for cals to
' be 500
's CPI.Parse "SYSTem:UNCertainty:POINts:MAXimum 500"
MsgBox "Max number of uncertainty points for cals = " &
' Perform a Save of the currently loaded uncertainty workspace ('.ml4') file
scpi.Parse "SYSTem:UNCertainty:STORe"
' Uncomment the following line to save the currently loaded uncertainty workspace to a
' specific ('.ml4') filename
's CPI.Parse "SYSTem:UNCertainty:STORe 'myUncertaintyWorkspace.ml4'
' Uncomment the following line to recall the uncertainty workspace ('.ml4') file that we
' just saved
's CPI.Parse "SYSTem:UNCertainty:LOAD 'myUncertaintyWorkspace.ml4"
End Sub

'*********************************************
' Measure the steps of a cable characterization,
' noise characterization, or a calibration.
Sub MeasureStepsOfCharacterizationOrCal(isNoiseCharacterization, numSteps)
Dim retVal
Dim opc
For i = 1 To numSteps
   step = "Step " + CStr(i) + " of " + CStr(numSteps)
   ' A "connection adjustment" means the user has to change something about the
   ' connection,
   ' such as wiggling cable and re-connecting standard during a repeatability
   ' characterization,
   ' or to move the slide position of a sliding load if a connection step in a
   ' calibration
   ' were to involve a sliding load.
   If isNoiseCharacterization = False Then
CStr(i))
   Else ' noise characterization
      ' in this case, "iterations" are just repetitive measurements, not re-adjust/re-
      measure the standard
      numConnectionAdjustmentsPerStep = 1
   End If
   ' ...
End If
For j = 1 To numConnectionAdjustmentsPerStep
  retVal = MsgBox(strPrompt, vbOKCancel, step)
  If retVal = vbCancel Then WScript.Quit
  opc = scpi.Parse("sens1:corr:coll:guid:acq STAN" + CStr(i) + ";*OPC?"")
Next

scpi.Parse "sens1:corr:coll:guid:save"
End Sub

Function FormatList(list)
Dim tokens
  ' Strip the leading and trailing quotation
  ' marks from the list string
  list = Mid(list, 2, Len(list) - 3)
  ' Tokenize the comma-delimited list string
  ' into an array of the individual substrings
tokens = Split(list, ",")
  ' Using Trim to remove leading and trailing spaces.
  For i = 0 To UBound(tokens)
    tokens(i) = Trim(tokens(i))
  Next
  FormatList = tokens
End Function

'*********************************************
' Performing a NON-UNCERTAINTY Guided 2-port cal (Ports 1 and 2)
Sub TwoPortGuidedCal
Dim connList
Dim connIndex1, connIndex2
Dim kitList
Dim kitIndex
Dim message
Dim formattedConnListStr
message = "A non-uncertainty 2-port calibration will now be performed, "
message = message & "because a calibration is needed for characterizing the cables"
MsgBox message
  ' Query the list of connectors that the VNA system recognizes
  ' Turn the list from comma-delimited string into an array of strings
connList = FormatList(connList)
message = "Enter the index number (from this list) of your DUT connector for Port 1:" & Chr(10) & Chr(10)
' Note: the array's indexing is 0-based but the numbering is being presented in the
InputBox as 1-based
For i = 0 To UBound(connList)
formattedConnListStr = formattedConnListStr & Chr(9) & CStr(i+1) + ") " & connList(i)
& Chr(10)
Next
connIndex1 = InputBox(message & formattedConnListStr)
If connIndex1 = "" Then WScript.Quit
' Convert number from String to Long and from 1-based index to 0-based
connIndex1 = CLng(connIndex1) - 1
scpi.Execute "sens:corr:coll:guid:conn:port1 '" & connList(connIndex1) & "'"
message = "Enter the index number (from this same list again) of your DUT connector for
Port 2:" & Chr(10) & Chr(10)
connIndex2 = InputBox(message & formattedConnListStr)
If connIndex2 = "" Then WScript.Quit
' Convert number from String to Long and from 1-based index to 0-based
connIndex2 = CLng(connIndex2) - 1
scpi.Execute "sens:corr:coll:guid:conn:port2 '" & connList(connIndex2) & "'"
' Note: If your VNA has more than 2 ports, then uncomment
' one or both of these next two lines.
'scpi.Execute "sens:corr:coll:guid:conn:port3 ""Not used"" "
'scpi.Execute "sens:corr:coll:guid:conn:port4 ""Not used"" "
' This next block of commented code demonstrates how to specify an adapter
' and it's electrical delay, in situations where you are performing an
' Unknown Thru or Adapter Removal calibration.

In most situations, the

' VNA is able to correctly determine an adapter's electrical length
' at the end of the calibration.

However, there are scenarios where

' the VNA cannot correctly calculate the length -- such as when the channel
' has a relatively small number of measurement points (for example, 201 or less)
' and the adapter is significantly long (for example, a cable that is several feet).
' In these cases, the ADAP commands (below) enable you to explicitly specify
' the adapter you are using.
' Send these commands prior to the "sens:corr:coll:guid:init" command.
' Create adapter and return the adapter number
"','"& connList(connIndex2) & "'")
' The adapterNum string contains a '+' character.
' Here we convert to integer to remove that.
'adapterNum = CStr( CInt(adapterNum) )
' Specify that this adapter has 10 nanoseconds electrical delay (coaxial).
'scpi.Execute "sens:corr:coll:guid:adap" & adapterNum & ":del 10E-9"
' Text description of adapter

8322


'scpi.Execute "sens:corr:coll:guid:adap" & adapterNum & ":desc 'My adapter'"
' Select to use this adapter specifically between ports 1 and 2
'scpi.Execute "sens:corr:coll:guid:adap" & adapterNum & ":path 1,2"
' End of adapter block

' Query the list of acceptable cal kits and
' ECal module characterizations for Port 1.
' Turn the list from a comma-delimited string into an array
kitList = FormatList(kitList)
' Select the Cal Kit or ECal module characterization to use for Port 1.
message = "Enter the index number (from this list) of your cal kit or ECal module
characterization for Port 1:"
message = message & Chr(10) & Chr(10)
For i = 0 To UBound(kitList)
    message = message & Chr(9) & CStr(i+1) + ") " & kitList(i) & Chr(10)
Next
kitIndex = InputBox(message)
If kitIndex = "" Then WScript.Quit
' Convert number from String to Long and from 1-based index to 0-based
kitIndex = CLng(kitIndex) - 1
scpi.Execute "sens:corr:coll:guid:ckit:port1 " & kitList(kitIndex) & ""
' Query the list of acceptable cal kits
' and ECal module characterizations for Port 2.
' Turn the list from a comma-delimited string into an array
kitList = FormatList(kitList)
' Select the Cal Kit or ECal module characterization to use for Port 2.
message = "Enter the index number (from this list) of your cal kit or ECal module
characterization for Port 2:"
message = message & Chr(10) & Chr(10)
For i = 0 To UBound(kitList)
    message = message & Chr(9) & CStr(i+1) + ") " & kitList(i) & Chr(10)
Next
kitIndex = InputBox(message)
If kitIndex = "" Then WScript.Quit
' Convert number from String to Long and from 1-based index to 0-based
kitIndex = CLng(kitIndex) - 1
scpi.Execute "sens:corr:coll:guid:ckit:port2 " & kitList(kitIndex) & ""
' This determines whether the cal will be a "Guided Power Cal"
' or just a traditional S-parameter cal.
message = "On which port number shall power be measured? "
message = message & "For a traditional guided cal without power cal, enter 0"

Dim powerPort
powerPort = CInt( InputBox(message) )
If powerPort > 0 Then
    scpi.Execute("sens:corr:coll:guid:psen" & CStr(powerPort) & ":stat on")
End If
Dim retVal
retVal = MsgBox("Is the power sensor's connector type or gender different from the DUT connector for that port?", vbYesNo)
If retVal = vbYes Then
    message = "Enter the index number of your power sensor's connector. Choose from this list:" & Chr(10) & Chr(10)
    ' Select the sensor's connector.
    connIndex1 = InputBox(message & formattedConnListStr)
    If connIndex1 = "" Then WScript.Quit
    connIndex1 = CLng(connIndex1) - 1
    scpi.Execute "sens:corr:coll:guid:psen" & CStr(powerPort) & ":conn " & connList(connIndex1) & ":" & ":
    ' Query the list of acceptable cal kits and ECal module characterizations 
    ' that are applicable for the sensor's connector.
    ' Turn the list from comma-delimited string into an array of strings
    kitList = FormatList(kitList)
    ' Select the Cal Kit or ECal module characterization to use for de-embed of the sensor's connector.
    message = "Enter index number of your cal kit or ECal module characterization to use for de-embed of the sensor's connector. 
    message = message & "Choose from this list:" & Chr(10) & Chr(10)
    For i = 0 To UBound(kitList)
        message = message & Chr(9) & CStr(i+1) + ") " & kitList(i) & Chr(10)
    Next
    kitIndex = InputBox(message)
    If kitIndex = "" Then WScript.Quit
    kitIndex = CLng(kitIndex) - 1
    scpi.Execute "sens:corr:coll:guid:psen" & CStr(powerPort) & ":ckit " & kitList(kitIndex) & ":
    Else
    scpi.Execute("sens:corr:coll:guid:psen" & CStr(powerPort) & ":conn 'Ignored'")
End If ' End of block that considers the sensor's connector

End If ' End of block that considers the sensor's connector

' Ask for the power level to perform the power cal at 
' (if this command is omitted, the default is 0 dBm).
Dim powerLevel
powerLevel = InputBox("Enter the power level for the power cal to be performed at")
If powerLevel = "" Then Exit Sub
scpi.Execute "sens:corr:coll:guid:psen" & CStr(powerPort) & ":pow:lev " & powerLevel
Else
End If ' End of block that considers if the cal will include power calibration

' This next block of commented code
' shows optional functions when using ECal.
' Send these "sens:corr:pref" commands prior to the
' Read ECAL information from ECal module #1 on the USB bus
' about the Keysight factory characterization data
'module1Info = scpi.Execute("sens:corr:coll:ckit:inf? ECAL1,CHAR0")
'MsgBox "Description of ECal Module #1:" & Chr(10) & Chr(10) & module1Info

' The following command enables auto orientation of
' the ECal module (The VNA senses which port of the
' module is connected to which port of the VNA).
'scpi.Execute "sens:corr:pref:ecal:ori ON"
' However, if you are measuring at very low power levels where
' the VNA may fail to sense the module's orientation, then turn auto
' orientation OFF and specify how the module is connected.
' "A1,B2" indicates Port A of the module is connected
' to VNA Port 1 and Port B is connected to VNA Port 2).
'scpi.Execute "sens:corr:pref:ecal:ori OFF"
' End of optional ECal setup

' Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"
numCalSteps = CLng(scpi.Execute("sens:corr:coll:guid:steps?"))
.MsgBox "Number of steps is " + CStr(numCalSteps)
' Measure the standards and conclude the cal
MeasureStepsOfCharacterizationOrCal False, numCalSteps
.MsgBox "The cal needed for the cable repeatability characterizations is done"
End Sub

'*********************************************
' Perform Cable Characterization
Sub PerformCableChar
' Select/activate the default (S11) measurement that should still
' be present since a Preset preceded this subroutine
scpi.Parse "CALC1:PAR:SEL 'CH1_S11_1'"
' Ensure the previously-performed calibration is turned on for that selected measurement
' Initiate repeatability characterizations on Channel 1 for Ports 1 and 2, with 3
' iterative measurements of each standard (normally users should specify more than 3
' iterations per standard but it just makes this example speedier)
Dim numRequestedIterations
numRequestedIterations = 3
' Begin characterization for port 1
scpi.Parse "sens1:corr:coll:guid:unc:char:cable 1, " + CStr(numRequestedIterations)
Dim numCharSteps
' Query the number of connection steps
numCharSteps = CLng(scpi.Parse("sens1:corr:coll:guid:steps?"))
message = "Now to characterize the cable currently associated with port 1. "
message = message & "Number of standard steps for characterizing port 1 cable = " & numCharSteps
MsgBox message
MeasureStepsOfCharacterizationOrCal False, numCharSteps
MsgBox "Cable characterization for port 1 is done"
' Begin characterization for port 2
' Query the number of connection steps
numCharSteps = CLng(scpi.Parse("sens1:corr:coll:guid:steps?"))
message = "Now to characterize the cable currently associated with port 2. "
message = message & "Number of standard steps for characterizing port 2 cable = " & numCharSteps
MsgBox message
MeasureStepsOfCharacterizationOrCal False, numCharSteps
MsgBox "Cable characterization for port 2 is done"
End Sub

'**************************************************
'Perform Noise Characterization
Sub PerformNoiseChar
' Initiate noise characterization on Channel 1 for Ports 1 and 2, with 3 iterative
' measurements of each standard (normally users should specify more than 3 iterations
' per standard but it just makes this example speedier)
scpi.Parse "sens1:corr:coll:guid:unc:char:noise 1, 2, 3"
Dim numCharSteps
' Query the number of connection steps
numCharSteps = CLng(scpi.Parse("sens1:corr:coll:guid:steps?"))
message = "Now to characterize the system noise for ports 1 and 2. "
message = message & "Number of connection steps to be made = " & numCharSteps
MsgBox message
MeasureStepsOfCharacterizationOrCal True, numCharSteps
MsgBox "Noise characterization is done"
End Sub

'*********************************************
'Perform 2-port Uncertainty Calibration (Ports 1 and 2) on Channel 1
Sub TwoPortUncertCal
Dim kitList
Dim kitIndex
Dim message
' Specify this calibration is to be performed with uncertainties.
scpi.Parse "sens1:corr:coll:guid:UNCertainty:ENABle ON"
' Query the list of uncertainty cal kits and
' ECal module factory characterizations applicable for the port cable connector.
' Turn the list from comma-delimited string into an array of strings
kitList = FormatList(kitList)
message = "Now to perform an uncertainty calibration for ports 1 and 2. "
message = message & "Enter the index number (from this list) of your cal kit or ECal module to use:"
message = message & Chr(10) & Chr(10)
' Note: the array's indexing is 0-based but the numbering is being presented in the InputBox as 1-based
For i = 0 To UBound(kitList)
    message = message & Chr(9) & CStr(i+1) + "") " & kitList(i) & Chr(10)
Next
' Select the Cal Kit or ECal module to use for both ports.
' If the selected kit does not have a suitable set of standards
' that can mate the connector of the cable attached to Port 2,
' then the attempt to INIT the cal should yield an applicable error message.
kitIndex = InputBox(message)
If kitIndex = "" Then WScript.Quit
' Convert from String to Long and from 1-based index to 0-based
kitIndex = CLng(kitIndex) - 1
scpi.Parse "sens:corr:coll:guid:ckit:port1 " & kitList(kitIndex) & ""
scpi.Parse "sens:corr:coll:guid:ckit:port2 " & kitList(kitIndex) & ""
' Note: If your VNA has more than 2 ports, then uncomment
' one or both of these next two lines just in case this channel
' may have had a remote non-Uncert cal performed prior that
' involving port 3 and/or 4.
' This next block of commented code
' shows optional functions when using ECal.
' Send these "sens:corr:pref" commands prior to the
' Read ECAL information from ECal module #1 on the USB bus
' about the factory characterization data
'module1Info = scpi.Parse("sens:corr:coll:ckit:inf? ECAL1,CHAR0")
'MsgBox "Description of ECal Module #1:" & Chr(10) & Chr(10) & module1Info

' The following command enables auto orientation of
' the ECal module (The VNA senses which port of the
' module is connected to which port of the VNA).
'sci.Parse "sens:corr:pref:ecal:ori ON"
' However, if you are measuring at very low power levels where
' the VNA may fail to sense the module's orientation, then turn auto
' orientation OFF and specify how the module is connected.
' "A1,B2" indicates Port A of the module is connected
' to VNA Port 1 and Port B is connected to VNA Port 2).
'sci.Parse "sens:corr:pref:ecal:ori OFF"
' End of optional ECal setup
' Initiate the calibration and query the number of steps
'sci.Parse "sens:corr:coll:guid:init"
MsgBox "Number of steps is " + CStr(numCalSteps)
' Measure the standards and conclude the cal
MeasureStepsOfCharacterizationOrCal False, numCalSteps
MsgBox "Cal is done"
End Sub

'*****************************************************

'Make Uncertainty Trace Settings
Sub MakeTraceSettings
' Note: normally you would first ensure your active measurement is an S-parameter channel
' measurement before this point, and that the channel has an uncertainty cal to be
' able to turn on uncertainty display, but as the TwoPortUncertCal subroutine has
' executed right before this one, that is already ensured.
chanNum = CInt(scpi.Parse("SYSTem:ACTive:CHANnel?"))
If (chanNum = 0) Then
   MsgBox "No active channel"
   WScript.Quit
End If
activeMeasName = scpi.Parse("SYSTem:ACTive:MEASurement?")
scpi.Parse "CALCulate" & CStr(chanNum) & ":PARameter:SELect " & activeMeasName
MsgBox "Uncertainty Display Type = " & scpi.Parse("CALCulate" & CStr(chanNum) & ":UNCertainty:DISPlay:TYPE?")
On Error Resume Next
scpi.Parse "CALCulate" & CStr(chanNum) & ":UNCertainty:DISPlay:TYPE SHADe"
If Err.Number <> 0 Then
    MsgBox Err.Description
    Err.Clear
Else
    MsgBox "Uncertainty Display Type = " & scpi.Parse("CALCulate" & CStr(chanNum) & ":UNCertainty:DISPlay:TYPE?")
End If
On Error GoTo 0
MsgBox "CoverageFactor = " & scpi.Parse("CALCulate" & CStr(chanNum) & ":UNCertainty:DISPlay:CFACtor?")
' Uncomment the following line to try setting the CoverageFactor = 3
'scpi.Parse "CALCulate" & CStr(chanNum) & ":UNCertainty:DISPlay:CFACtor 3"
scpi.Parse "CALCulate" & CStr(chanNum) &":UNCertainty:MODE:CABLE:REPeat OFF"
MsgBox "Cable Repeatability Uncertainty state = " & scpi.Parse("CALCulate" & CStr(chanNum) & ":UNCertainty:MODE:CABLE:REPeat?")
scpi.Parse "CALCulate" & CStr(chanNum) &":UNCertainty:MODE:CABLE:REPeat ON"
MsgBox "Cable Repeatability Uncertainty state = " & scpi.Parse("CALCulate" & CStr(chanNum) & ":UNCertainty:MODE:CABLE:REPeat?")
scpi.Parse "CALCulate" & CStr(chanNum) &":UNCertainty:MODE:NOISe OFF"
MsgBox "Measurement Noise Uncertainty state = " & scpi.Parse("CALCulate" & CStr(chanNum) & ":UNCertainty:MODE:NOISe?")
scpi.Parse "CALCulate" & CStr(chanNum) &":UNCertainty:MODE:NOISe ON"
MsgBox "Measurement Noise Uncertainty state = " & scpi.Parse("CALCulate" & CStr(chanNum) & ":UNCertainty:MODE:NOISe?")
scpi.Parse "CALCulate" & CStr(chanNum) &":UNCertainty:MODE:ETERm OFF"
MsgBox "Error Term Uncertainty state = " & scpi.Parse("CALCulate" & CStr(chanNum) & ":UNCertainty:MODE:ETERm?")
scpi.Parse "CALCulate" & CStr(chanNum) &":UNCertainty:MODE:ETERm ON"
MsgBox "Error Term Uncertainty state = " & scpi.Parse("CALCulate" & CStr(chanNum) & ":UNCertainty:MODE:ETERm?")
End Sub
Getting and Putting Data

This Rocky Mountain Basic example does the following:

- Takes a sweep, and reads the formatted data trace into an array. The trace is read as a definite length block.
- Instructs you to remove DUT
- Downloads the trace back to the analyzer as an definite length block.

---

**See Other SCPI Example Programs**

Sub SampleGetPutData()
'*** The variables of the resource manager and the instrument I/O are declared.
Dim ioMgr As VisaComLib.ResourceManager
Dim GPIB As VisaComLib.FormattedIO488
'*** The memory area of the resource manager and the instrument I/O are acquired.
Set ioMgr = New VisaComLib.ResourceManager
Set GPIB = New VisaComLib.FormattedIO488
'*** Open the instrument.
Set GPIB.IO = ioMgr.Open("GPIB0::16::INSTR")
GPIB.IO.timeout = 10000

Dim Numpts As Long
Dim Datam As Variant

'Select the measurement
GPIB.WriteString "CALCulate1:MEASure1:PARameter 'S21'", True
'Read the number of data points
GPIB.WriteString "SENSe1:SWEep:POINts?", True
Numpts = GPIB.ReadNumber
'Turn continuous sweep off
GPIB.WriteString "INITiate:CONTinuous OFF", True
'Take a sweep
GPIB.WriteString "INITiate1:IMMediate;*WAI", True
'Ask for the Data
'PICK ONE OF THESE LOCATIONS TO READ
GPIB.WriteString "CALCulate1:MEASure1:DATA:FDATA?", True
'Formated Meas
'GPIB.WriteString "CALCulate1:MEASure1:DATA:FMEM?", True
'Formated Memory
'GPIB.WriteString "CALCulate1:MEASure1:DATA:SDATA?", True
'Corrected, Complex Meas
'GPIB.WriteString "CALCulate1:MEASure1:DATA:SMEM?", True
'Corrected, Complex Memory
'GPIB.WriteString "SENSe1:CORRection:CSET:ETERm:DATA? 'Directivity(1,1)'", True
>Error-Term Directivity

'Parse the data
Datam = GPIB.ReadList(ASCIIType_R8, ",")

'PUT THE DATA BACK IN
GPIB.WriteString "CALCulate1:MEASure1:DATA:FDATA ", False
' Formatted Meas
'GPIB.WriteString "CALCulate1:MEASure1:DATA:FMEM ", False
' Formatted Memory
'GPIB.WriteString "CALCulate1:MEASure1:DATA:SDATA ", False
' Corrected, Complex Meas
'GPIB.WriteString "CALCulate1:MEASure1:DATA:SMEM ", False
' Corrected, Complex Memory
'GPIB.WriteString "SENSe1:CORRection:CSET:ETERm:DATA 'Directivity(1,1)',", False
Error-Term Directivity
GPIB.WriteList Datam, ASCIIType_R8, ",", True

'*** End procedure
GPIB.IO.Close
End Sub

100 DIM A$[10],Data1(1:51)
110 INTEGER Digits,Bytes
120 !
130 COM /Sys_state/ @Hp87xx,Scode
140 ! Identify I/O Port
150 CALL Iden_port
160 !
170 !
180 OUTPUT @Hp87xx;"SYST:PRES"
190 !
200 OUTPUT @Hp87xx;"CALC:PAR:SEL 'CH1_S11_1'"
210 !
220 ! Set up the analyzer to measure 51 data points.
230 OUTPUT @Hp87xx;"SENS1:SWE:POIN 51:*OPC?"
240 ENTER @Hp87xx;Opc
250 !
260 ! Take a single sweep, leaving the analyzer
270 ! in trigger hold mode.
280 OUTPUT @Hp87xx;"ABOR;:INIT1:CONT OFF;:INIT1:*WAI"
290 !
300 ! Select binary block transfer
310 OUTPUT @Hp87xx;"FORM:DATA REAL,64"
320 !
330 ! Request the channel 1 formatted data array
340 ! from the analyzer.
350 OUTPUT @Hp87xx;"CALC:DATA? FDATA"
360 !
370 ! Turn on ASCII formatting on the I/O path.
380 ! It is needed for reading the header
390 ! information.
400 ASSIGN @Hp87xx;FORMAT ON
410 !
420 ! Get the data header. "A$" will contain the
430 ! 
440 ! "Dig" character indicating a block data transfer.
450 ! for the number of bytes value which follows.
ENTER @Hp87xx USING "%,A,D";A$,Digits

! Get the rest of the header. The number of
! bytes to capture in the data array will be
! placed in "Bytes". Note the use of "Digits"
! in the IMAGE string.

ENTER @Hp87xx USING "%,"&VAL$(Digits)&"D";Bytes
PRINT "HEADER",A$,Digits,Bytes

! Turn off ASCII formatting on the I/O path;
! it is not needed for transferring binary
! formatted data.
ASSIGN @Hp87xx;FORMAT OFF

! Get the data.
ENTER @Hp87xx;Data1(*)

! Turn on ASCII formatting again.
ASSIGN @Hp87xx;FORMAT ON

! Get the "end of data" character.
ENTER @Hp87xx;A$

! Display the first three numbers in the array.
DISP "Trace: ";Data1(1);Data1(2);Data1(3);"...

! Use this time to visually compare the
! numbers to the visible data trace.
WAIT 5

! Prompt the operator to disconnect the test
! device and how to continue the program.
DISP "Disconnect the test device -- Press Continue"
PAUSE

! Update the display line.
DISP "Taking a new sweep...";

! Take a sweep so the display shows new data.
OUTPUT @Hp87xx;":INIT1;*WAI"
DISP " Done."
WAIT 5

! Send the header for an indefinite block length
! data transfer.
DISP "Downloading saved trace...";

! The first byte '3' indicates the next three digits equal number of transfer bytes
! The number of transfer bytes equals 8x the number of tracepoints.
OUTPUT @Hp87xx;"CALC:DATA FDATA, #3408";

! Turn off ASCII formatting.
ASSIGN @Hp87xx;FORMAT OFF

! Send the data array back to the analyzer.
OUTPUT @Hp87xx;Data1(*),END
1000 ! Turn on ASCII formatting again.
1010 ASSIGN @Hp87xx;FORMAT ON
1020 DISP " Done!"
1030 END
1040 !
1050 !*************************************************************
1060 ! Iden_port: Identify io port to use
1070 ! Description: This routines sets up the I/O port address for
1080 ! the SCPI interface. For "HP 87xx" instruments,
1090 ! the address assigned to @Hp87xx = 800 otherwise,
1100 ! 716.
1110 !*************************************************************
1120 SUB Iden_port
1130 COM /Sys_state/ @Hp87xx,Scode
1140 !
1150 IF POS(SYSTEM$("SYSTEM ID"),"HP 87")<>0 THEN
1160 ASSIGN @Hp87xx TO 800
1170 Scode=8
1180 ELSE
1190 ASSIGN @Hp87xx TO 716
1200 Scode=7
1210 END IF
1220 !
1230 SUBEND !Iden_port
1240 !
8333
Independent Power Calibration

The following program creates an independent power calibration over a specified frequency span when performing a Cal All.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as BalancedCOM.vbs. Learn how to setup and run the macro.

```
SYST:PRESET
sens:freq:start 1e9
sens:freq:stop 2e9
calc2:par:def "S22Ch2",S22
disp:wind:trac2:feed "S22Ch2"
sens2:freq:start 5e9
sens2:freq:stop 7e9
SYST:CAL:ALL:RESet
syst:cal:all:sel 1,2
syst:cal:all:chan1:port:sel 1,2
syst:cal:all:mcl:prop:val "Include Power Calibration","true"
syst:cal:all:mcl:prop:val "Enable Extra Power Cals","Port 2,Port 3"
SYST:CAL:ALL:GUID:CHAN?
syst:cal:all:ind:sour:cal:cat?
syst:cal:all:ind:sour3:cal:rang1:start 3e9
syst:cal:all:ind:sour3:cal:rang1:stop 4e9
syst:cal:all:ind:sour3:cal:rang1:points 21
syst:cal:all:ind:sour3:cal:range2:start 20e9
syst:cal:all:ind:sour3:cal:range2:stop 21e9
syst:cal:all:ind:sour3:cal:range2:points 7
syst:cal:all:ind:sour2:cal:rang1:start 3e9
syst:cal:all:ind:sour2:cal:rang1:stop 4e9
syst:cal:all:ind:sour2:cal:rang1:points 21
syst:cal:all:ind:sour2:cal:range:count?
syst:cal:all:ind:sour3:cal:range1:start?
```
The channel number used for the SENSe header is determined by the **SYST:CAL:ALL:GUID:CHAN?** command. You must query this channel number – do not assume that it will always be a particular value.
**Multiple Instances of Calibrate All Channels**

**Note:** You can assign a single client channel to multiple cal all instances. However, care should be taken here. If you create user calsets, all calsets will be created but only the last one will be applied. If you are only using a cal register, only the last cal all will be written to the cal register (only supports one calibration). The order that the cal alls are created should be the order in which they are saved (SENS200:CORR:COLL:GUID:SAVE). This ensures that the client channel imports the proper ETerms.

```
# create 4 total channels:
# does one calall on channels 1/2 with ports 1/2 selected and power cal on P1
# does another calall on channels 3/4 with ports 3/4 selected and no power cal

syst:preset
calc2:par:ext 'ch2_s11','S11'
DISPLAY:WNDow1:TRACe2:FEED "ch2_s11"
calc3:par:ext 'ch3_s11','S11'
DISPLAY:WNDow1:TRACe3:FEED "ch3_s11"
calc4:par:ext 'ch4_s11','S11'
DISPLAY:WNDow1:TRACe4:FEED "ch4_s11"
sYST:CAL:ALL:RESet
sYST:CAL:ALL:CHAN1:PORT:SEL 1,2
sYST:CAL:ALL:CHAN2:PORT:SEL 1,2
sYST:CAL:ALL:CSET:PREFIX 'firstCal'
sYST:CAL:ALL:GUID:CHAN?
# returns 200
sYST:CAL:ALL2:RESet
sYST:CAL:ALL2:SEL 3,4
sYST:CAL:ALL2:CSET:PREFIX 'secondCal'
sYST:CAL:ALL2:CHAN3:PORT:SEL 3,4
sYST:CAL:ALL2:CHAN4:PORT:SEL 3,4
sYST:CAL:ALL2:GUID:CHAN?
# returns 500

# do power cal for first
sYST:CAL:ALL:MCL:PROP:VAL 'Include Power Calibration', 'True'
# but not on the second
sYST:CAL:ALL2:MCL:PROP:VAL 'Include Power Calibration', 'False'
SENS200:corr:coll:guid:conn:port1 "APC 3.5 male"
SENS200:corr:coll:guid:conn:port2 "APC 3.5 male"
SENS200:CORR:COLL:GUID:CKIT:PORT1 '85052D'
SENS200:CORR:COLL:GUID:CKIT:PORT2 '85052D'
SENS200:CORR:COLL:GUID:INIT
SENS200:CORR:COLL:GUID:STEP?
SENS500:corr:coll:guid:conn:port3 "APC 3.5 male"
SENS500:corr:coll:guid:conn:port4 "APC 3.5 male"
SENS500:CORR:COLL:GUID:CKIT:PORT3 '85052D'
SENS500:CORR:COLL:GUID:CKIT:PORT4 '85052D'
```
SENS500:CORR:COLL:GUID:INIT
SENS500:CORR:COLL:GUID:STEP?

#in this case, cals are totally different (different ports)
#just do one and then the other
#first step for first cal is the power step:
SENS200:CORR:COLL:GUID:DESC? 1
SENS200:CORR:COLL:GUID:ACQ STAN1; *OPC?
SENS200:CORR:COLL:GUID:ACQ STAN2; *OPC?
SENS200:CORR:COLL:GUID:ACQ STAN3; *OPC?
SENS200:CORR:COLL:GUID:DESC? 4
SENS200:CORR:COLL:GUID:ACQ STAN4; *OPC?
SENS200:CORR:COLL:GUID:ACQ STAN5; *OPC?
SENS200:CORR:COLL:GUID:ACQ STAN6; *OPC?
SENS200:CORR:COLL:GUID:ACQ STAN7; *OPC?
SENS200:CORR:COLL:GUID:DESC? 8
SENS200:CORR:COLL:GUID:ACQ STAN8; *OPC?

#now do second cal:
SENS500:CORR:COLL:GUID:DESC? 1
SENS500:CORR:COLL:GUID:ACQ STAN1; *OPC?
SENS500:CORR:COLL:GUID:DESC? 2
SENS500:CORR:COLL:GUID:ACQ STAN2; *OPC?
SENS500:CORR:COLL:GUID:DESC? 3
SENS500:CORR:COLL:GUID:ACQ STAN3; *OPC?
SENS500:CORR:COLL:GUID:DESC? 4
SENS500:CORR:COLL:GUID:ACQ STAN4; *OPC?
SENS500:CORR:COLL:GUID:ACQ STAN5; *OPC?
SENS500:CORR:COLL:GUID:ACQ STAN6; *OPC?
SENS500:CORR:COLL:GUID:ACQ STAN7; *OPC?

#SAVE!!
SENS200:CORR:COLL:GUID:SAVE
SENS500:CORR:COLL:GUID:SAVE

The channel number used for the SENSe header is determined by the SYST:CAL:ALL:GUID:CHAN? command. You must query this channel number – do not assume that it will always be a particular value.
Noise Power Ratio (NPR) Correction Example

The following example uses port 4 and port 1, and a MXG named "MyMXG". The MXG is connected to port 4's rear panel. It uses a NPR modulation file "D:\ModulationFile\npr.csv" and saves the corrected modulation file into "D:\ModulationFile\npr.mdx".

See Other SCPI Example Programs

*IDN?
SYST:FPR

DISP:WIND1:STAT ON
CALC:CUST:DEF 'CH1_A','Spectrum Analyzer','A'
DISP:WIND1:TRAC1:FEED 'CH1_A'

SENS:PATH:CONF:ELEM 'Port4Bypass', 'RearPanel'
SYST:CONF:EDEV:STATe 'MyMxg', ON
SOUR:MOD6:LOAD 'D:\ModulationFile\npr.csv'
SENS:SA:COH:DIST:MOD:STAT ON
SOUR:POW6:MODE ON

SOUR:MOD6:CORR:COLL:POW:ENAB 1
SOUR:MOD6:CORR:COLL:FLAT:ENAB 1
SOUR:MOD6:CORR:COLL:FLAT:REC "R4"
SOUR:MOD6:CORR:COLL:NOTCh:ENAB 1
SOUR:MOD6:CORR:COLL:NOTCh:REC "R4"
SOUR:MOD6:CORR:COLL:ACP:ENAB 1
SOUR:MOD6:CORR:COLL:ACP:REC "R4"
SOUR:MOD6:CORR:COLL:FAST:ENAB 1
SOUR:MOD6:CORR:COLL:ACQ
*OPC?
SOUR:MOD6:CORR:COLL:ACQ:STAT?
SOUR:MOD6:CORR:STAT ON
Perform a Guided Cal using Multiple Power Sensors

This example performs a Guided Calibration using TWO power sensors that are setup as PMAR devices. Use of two power sensors is necessary when the frequency range of the measurement is greater than can be attained with a single sensor.

A measurement must first be set up with desired frequency range, power, and so forth, ready to be calibrated.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do not need to control the VNA via GPIB to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as mps.vbs.

Learn how to setup and run the macro.

See Guided Power Sensor SCPI commands

See Other SCPI Example Programs
next
s.parse "SENS:CORR:COLL:GUID:SAVE"
Perform a Guided Cal with Sliding Load

This example sets the sliding load behavior, then performs a Guided Calibration that uses a sliding load. A measurement must first be set up with desired frequency range, power, and so forth, ready to be calibrated.

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do not need to control the VNA via GPIB to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as guided.vbs.

Learn how to setup and run the macro.

See Guided Cal commands.

```
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' Specify that any sliding loads should be measured using the
' remote iterative method rather than launching sliding load dialog.
scpi.Execute "sens:corr:coll:guid:conn:port1 ""APC 3.5 female"" "
scpi.Execute "sens:corr:coll:guid:conn:port2 ""APC 3.5 male"" "
' 85052B cal kit uses sliding loads
scpi.Execute "sens:corr:coll:guid:ckit:port1 ""85052B"" "
scpi.Execute "sens:corr:coll:guid:ckit:port2 ""85052B"" "
scpi.Execute "sens:corr:coll:guid:init"


' Measure the standards
For i = 1 To numSteps
    step = "Step " + CStr(i) + " of " + CStr(numSteps)
    MsgBox strPrompt, vbOKOnly, step
    minIterations = scpi.Execute("sens:corr:coll:guid:iter:min? " + CStr(i))
    For j = 1 To minIterations
        If minIterations > 1 Then MsgBox "Adjust/position the standard for measurement " + CStr(j) + " of " + CStr(minIterations), vbOKOnly
        scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
    Next
    If iterationCount <> minIterations Then
        MsgBox "Unexpected error!", vbOKOnly, step
        scpi.Execute "sens:corr:coll:guid:iter:res " + CStr(i)
    End If
```
Next

' Conclude the calibration

scpi.Execute "sens:corr:coll:guid:save"
Perform a Guided QSOLT Cal

This example performs a Guided QSOLT calibration on a 4-port VNA.

Because the DUT port 1 is female and the other ports are male, a 'Zero Thru' can be used between port 1 and the other ports. If this were NOT the case, a "Defined Thru" would be needed in the listed Cal Kits for those ports. Learn more about Thru methods.

Although no standards are used for ports 2, 3, and 4, a Cal Kit must be defined for these ports.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do not need to control the VNA via GPIB to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as qsolt.vbs.

See Also
Learn more about QSOLT.
Learn how to setup and run the macro.
See Guided Cal commands.

```vbnet
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Execute "Calc1:Par:Mnum 1"
scpi.Execute "Sens1:CORR:COLL:GUID:THRU:PORT 1,2,1,3,1,4"
scpi.Execute "Sens1:CORR:COLL:GUID:PATH:TMET 1,2,'Zero Thru'"
scpi.Execute "Sens1:CORR:COLL:GUID:PATH:TMET 1,3,'Zero Thru'"
scpi.Execute "Sens1:CORR:COLL:GUID:PATH:TMET 1,4,'Zero Thru'"
scpi.Execute "Sens1:CORR:COLL:GUID:PATH:CMET 1,2,'QSOLT1'"
scpi.Execute "Sens1:CORR:COLL:GUID:PATH:CMET 1,3,'QSOLT1'"
scpi.Execute "Sens1:CORR:COLL:GUID:PATH:CMET 1,4,'QSOLT1'"
' Initiate the calibration and query the number of steps
scpi.Execute "sens1:corr:coll:guid:init"
MsgBox "Number of steps is " + CStr(numSteps)
```
' Measure the standards
For i = 1 To numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
MsgBox strPrompt, vbOKOnly, step
scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
Next
' Conclude the calibration
scpi.Execute "SENS1:CORR:COLL:GUID:SAVE"
Perform a VMC Mixer Characterization

This example performs a VMC Mixer Characterization ONLY.
To run this example program without error:

Replace the ECal module model and serial number with that of your own, or a mechanical cal kit model.

Store a 'default.csa' instrument state file on the VNA with the setup information for your mixer. Or add mixer setup information to this program. See an example.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as *.vbs.

Learn how to setup and run the macro.

See Other SCPI Example Programs

Dim app
Dim scpi

' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Parse "MMEM:LOAD 'default.csa'"

' Perform Cal

' Define the connector and calkit for port 1 of the VNA
scpi.Parse "sens:corr:coll:guid:conn:port1 'APC 3.5 female'"

' Define the connector and calkit for the mixer output port
scpi.Parse "sens:corr:coll:guid:conn:port3 'APC 3.5 female'"
' Select a characterization only calibration. This produces an s2p file of the mixer - so the mixer can be used as a calibration mixer for the VMC calibration
' The outcome of the calibration is an S2P file.
scpi.Parse "SENS:CORR:COLL:GUID:VMC:MIXer:CHAR:CAL:FIL 'C:\Program Files(x86)\Keysight\Network Analyzer\Documents\MyMixer.s2p''
' Set ECal Auto orientation ON
scpi.Parse "SENS:CORR:PREF:ECAL:ORI ON"
' For the mixer char step ONLY,
' Auto orientation is turned OFF by the VNA.
' Otherwise it would fail because of the loss of the mixer.
' Manually set the ECal orientation for that step.
' the main calibration loop
' a description for the connection instructions is read
' and then the standard is acquired
dim steps, strPrompt
scpi.Parse "sens:corr:coll:guid:init"
steps=scpi.Parse ("sens:corr:coll:guid:steps?")
wscript.echo "Number of Steps = " + cstr(steps)
if (steps > 0) then ' otherwise an error condition occurred
for i = 1 to steps
MsgBox strPrompt, vbOKOnly, step
scpi.Parse ("sens:corr:coll:guid:acq STAN" + CStr(i))
next
MsgBox ("Cal is done!")
end if
Perform an SMC Phase Ref Cal

This example sets Phase Reference Cal properties, then performs a Phase Reference calibration. It is NOT necessary to create an SMC measurement before performing a **remote** Phase Reference Cal. It is necessary when performed from the user interface.

The SCPI commands in this example are sent over a COM interface using the **SCPIStringParser** object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as *.vbs.

Learn how to setup and run the macro.

See Phase Reference Cal SCPI commands

Learn about SMC with Phase Ref Cal

```
Set app = CreateObject("agilentpna835x.application")
SET s = app.scpistringparser
s.parse "SYST:PRES"
S.parse "SENS:SWE:MODE HOLD"
'Set cal params
s.parse "SYST:CAL:PHAS:RESet"
S.parse "SYST:CAL:PHAS:FREQ:STAR 1e9"
s.parse "SYST:CAL:PHAS:FREQ:STOP 10E9"
' Read then set the phase ref ID
' Change this to your phase ref name
s.parse "SYST:CAL:PHAS:REF 'MYPILOT44'"
s.parse "SYST:CAL:PHAS:POW:ATT 10"
s.parse "SYST:CAL:PHAS:CONN 'APC 3.5 female'
S.Parse "SYST:CAL:PHAS:CKIT '85052D'"
' turn on port 4 as well. Port 1 and Port 2 are always on
s.parse "SYST:CAL:PHAS:PORT4 on"
' Uncomment the following line to use
' an unknown mixer to measure below 55 Mhz.
' If the unknown mixer cal is ON, then the start frequency of the
' entire calibration is always 10 Mhz
' s.parse "SYST:CAL:PHAS:UNKN:INCLude ON"
' Read chan num then begin guided cal
```
chan = s.parse("SYST:CAL:PHAS:GUID:CHAN?")
header = "SENS" & CInt(chan) & ":CORR:COLL:GUID:"
S.parse header & "INIT"
steps = s.parse(header & "STEPS?")
wscript.echo steps
' Acquire stds
for i = 1 to steps
    MsgBox s.parse(header & "DESC? " & i)
    s.parse header & "ACQ STAN" & i
next
s.parse header & "SAVE:CSET 'scpi_phase_reference'"
msgbox "done"
Perform Unguided THRU Response Cal

This example program performs Thru Response cals in both the forward and reverse directions. It does this by selecting the appropriate measurement right before acquiring the standard. The cal infers the direction from the measurement.

This program also demonstrates the use of the SENSE:CORR:PREF:CSET:SAVE command. The details are in the comments.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save it on the VNA hard drive as Unguided.vbs. Learn how to setup and run the macro.

```
Dim App
Set App = CreateObject("AgilentPNA835x.Application")
Dim Parser
Set Parser = App.SCPIStringParser
'Preset and delete measurement
Parser.Parse "SYSTem:FPReset"
'The following commands determine how the cal set is saved.
'Pick one of the following preferences, comment the other
'Save cals to separate new USER CalSets
'Parser.Parse "SENS:CORR:PREF:CSET:SAVE USER"
'Save both cals to a single cal register
'Parser.Parse "SENS:CORR:PREF:CSET:SAVE CALR"
'Save both cals to a single currently selected CalSet or register
Parser.Parse "SENS:CORR:PREF:CSET:SAVE REUSE"
'
'Create a new S21 Measurement
Parser.Parse "CALCulate:PARameter:DEFine:EXT 'MyS21Meas',S21"
Parser.Parse "DISPlay:WINDow1 ON"
Parser.Parse "DISPlay:WINDow1:TRACe1:FEED 'MyS21Meas'"
'Create a new S12 Measurement
Parser.Parse "CALCulate:PARameter:DEFine:EXT 'MyS12Meas',S12"
Parser.Parse "DISPlay:WINDow1:TRACe2:FEED 'MyS12Meas'"
'Turn off continuous sweep
Parser.Parse "INITiate:CONTinuous OFF"
'Begin cals
'Select a cal kit
```
Parser.Parse "SENSe:CORRection:COLLect:CKIT:SE lect 1"
'Perform a forward thru response cal
'Select the S21 Meas
Parser.Parse "CALCulate1:PARameter:SE lect 'MyS21Meas'"
'Set the calibration method to Thru Response
Parser.Parse "SENSe1:CORRection:COLLect:METHod TRAN1"
MsgBox("Connect Thru between ports Then press OK")
Parser.Parse "SENSe1:CORRection:COLLect:ACQuire STAN4"
Parser.Parse "SENSe1:CORRection:COLLect:SAVE"
'Then perform a reverse thru response cal
'Change measurement to S12
Parser.Parse "CALCulate1:PARameter:SE lect 'MyS12Meas'"
'Set the calibration method to Thru Response
Parser.Parse "SENSe1:CORRection:COLLect:METHod TRAN1"
'Ensure the thru connection is still in place
'Acquire Thru std in reverse direction
Parser.Parse "SENSe1:CORRection:COLLect:ACQuire STAN4"
'All standards have been measured.
Parser.Parse "SENSe1:CORRection:COLLect:SAVE"
'Turn ON continuous sweep
Parser.Parse "INITiate:CONTinuous ON"
MsgBox("The calibration has been completed")
Set ECal States

This example cycles through the state settings on the first ECal module it finds on the USB bus.

The state settings include all of the ECal states on Port A, Port B and the AB thru path. The first state on a port-pair path such as AB is the thru state that is used during calibrations. The second state on that path is the "confidence state" which is the equivalent of an attenuator that is used by the ECal Confidence Check feature.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as *.vbs.

Learn how to setup and run the macro.

See ECal State commands

---

### Option Explicit

Dim app
Set app = CreateObject("AgilentPNA835x.Application")
Dim scpi
Set scpi = app.ScpiStringParser
Dim moduleIndexList
' These are 1-based indices as opposed to 0-based,
' so if this query returns 0 it indicates there appear
' to be no ECal modules connected.
If CInt(moduleIndexList(0)) = 0 Then
    MsgBox "No ECal module was found"
    WScript.Quit(0)
End If
SetStates("A")
SetStates("B")
SetStates("AB")
MsgBox "Done"
Sub SetStates(path)
    Dim pathNumStates
    pathNumStates = CInt( scpi.Parse("CONT:ECAL:MOD1:PATH:COUN? " + path) )
End Sub
Dim stateNum
For stateNum = 1 To pathNumStates
    Dim stateNumStr
    stateNumStr = CStr(stateNum)
    Dim pathDescr
    If Len(path) = 1 Then
        pathDescr = "port " + path
    Else
        pathDescr = "path " + path
    End If
    Dim isOK
    isOK = MsgBox("Click OK to switch to state number " + stateNumStr + " of " + pathDescr, vbOKCancel)
    If isOK = vbCancel Then WScript.Quit(0)
    scpi.Parse "CONT:ECAL:MOD1:PATH:STAT " + path + "," + stateNumStr
Next
End Sub
Setup an FCA Segment Sweep

This example program shows how to setup a segment sweep in FCA.

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the VNA hard drive as FCASeg.vbs.

Learn how to setup and run the macro.

```
option explicit
dim App, scpi
set App = CreateObject("Agilentpna835x.application")
App.Reset
' Create FCA Scalar Mixer/Converter channel with an SC21 measurement:
App.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter", "SC21", 1
' Access the COM SCPI string parser
set scpi = App.ScpiStringParser
' Delete all existing segments, and create three new ones
scpi.Parse("sens:mix:segm:del:all")
scpi.Parse("sens:mix:segm:add 3")
' Turn #1 ON
scpi.Parse("sens:mix:segm1:stat on")

' Set segment sweep
' The following 'type:segm' command discards the changes made to the scratch mixer
' Therefore, precede with Apply
' Also, always do this before setting the LO port
scpi.Parse("sens:mix:apply")
scpi.Parse("sens:swe:type segm")
' Setup segment #1
' Input is swept: [1.1GHz, 1.39GHz]
' LO1 is fixed: 2.2 GHz
' Output is low-side mixing swept and calculated
' from input and LO1.
' Number of points is 21
' Input power is -10 dBm
```
' LO1 power is 10.0 dBm
scpi.Parse("sens:mix:segm1:inp:freq:star 1.10E9")
scpi.Parse("sens:mix:segm1:inp:pow -10.0")
scpi.Parse("sens:mix:segm1:lol:freq:fix 2.2E9")
scpi.Parse("sens:mix:segm1:lol:freq:mode fixed")
scpi.Parse("sens:mix:segm1:lol:pow 10.0")
scpi.Parse("sens:mix:segm1:poin 21")
scpi.Parse("sens:mix:segm1:stat on")
scpi.Parse("sens:mix:segm1:calc outp")
' Setup segment #2:
scpi.Parse("sens:mix:segm2:inp:pow -10.0")
scpi.Parse("sens:mix:segm2:lol:freq:fix 2.2E9")
scpi.Parse("sens:mix:segm2:lol:pow 10.0")
scpi.Parse("sens:mix:segm2:poin 21")
scpi.Parse("sens:mix:segm2:stat on")
scpi.Parse("sens:mix:segm2:calc outp")
' Setup segment #3:
scpi.Parse("sens:mix:segm3:inp:freq:star 1.50E9")
scpi.Parse("sens:mix:segm3:inp:freq:stop 1.6E9")
scpi.Parse("sens:mix:segm3:inp:pow -10.0")
scpi.Parse("sens:mix:segm3:lol:freq:fix 2.2E9")
scpi.Parse("sens:mix:segm3:lol:pow 10.0")
scpi.Parse("sens:mix:segm3:poin 21")
scpi.Parse("sens:mix:segm3:stat off")
scpi.Parse("sens:mix:segm3:calc outp")
' Mixer Input to be port 1
' Mixer output to Port 2
' Mixer LO to Port 3
scpi.Parse("sens:mix:lol:name ""Port 3"""")
' Apply the scratch mixer
scpi.Parse("sens:mix:apply")
Use an Existing Power Cal During an SMC Cal

This example shows how to use an existing Source Power Cal instead of the power cal that is performed during an SMC calibration. To run this program without modification, you need the following:

- A Mixer setup file saved on the VNA: C:\Program Files(x86)\Keysight\Network Analyzer\Documents\Mixer\MyMixer.mxr.
- If the mixer file uses an external LO source, it must also be attached and configured.
- An ECAL module that covers the frequency range of the measurement.
- An SMC cal set named "SMC_CAL". This is the cal set that source power correction data will be imported from. The input and output frequency ranges of the cal set must cover the corresponding ranges used during calibration, or guided cal initialization will fail.

**Error Messages**

If you attempt to import power cal data from an SMC calset that uses different ports than the ones currently in use, the message “The necessary calibration standards were not found.” will appear.

If the imported Cal Set does not cover the frequency range of the current cal, the message “Interpolation target is out of range. Cannot interpolate.” will appear.

**See Also**

Sens:Corr:Coll:Guid:SMC commands

This VBScript (*.vbs) program can be run as a macro in the VNA. To do this, copy the following code into a text editor file such as Notepad and save on the VNA hard drive as SMC.vbs. Learn how to setup and run the macro.

```vbs
Dim app
Dim scpi
' Create / Get the VNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
'---Create a Scalar Mixer Forward Measurement
'First, delete all measurements on the channel
scpi.Parse "CALC:PAR:DEL:ALL"
'Create a forward scalar mixer measurement and configure it in channel 1. The first parameter is a unique identifying string (specified by the user) to allow subsequent commands to be directed at this specific measurement.
scpi.Parse "CALC:cust:DEF 'My SC21', 'Scalar Mixer/Converter', 'SC21'
'Setup the new measurement as the 2nd trace in the active window
scpi.Parse "DISP:WIND:TRAC2:FEED 'My SC21'
'Make the new trace the active measurement
```
scpi.Parse "CALC:PAR:SEL 'My SC21'"

'The parameters of the mixer measurement can now be configured.
'This can be done by either using the individual SENS:MIX commands
'for each of the parameters or by loading a mixer setup file. This
'exemple loads a mixer setup file. The path name
'for the mixer file may be loaded from other mapped drives
scpi.Parse "SENS:MIxer:Load ""C:\Program Files(x86)\Keysight\Network
Analyzer\Documents\Mixer\MyMixer.mxr"

'-------------Perform A Scalar Mixer Calibration-----------------------
'Specify the connector types and the cal kits for each of the ports.
scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT1:SEL ""APC 3.5 male""

'Import power cal data from the existing SMC calset "MySMC"

'Specify the thru measurement method. This applies to both ECal
'and mechanical calibrations.
'Always send the init command before the Thru method command
scpi.Parse "SENS:CORR:COLL:GUID:INIT"
scpi.Parse "SENS:CORR:COLL:GUID:PATH:TMET 1,2,""DEFINED THRU"
'Omit the isolation part of the 2-port cal (default behavior).
scpi.Parse "SENS:CORR:COLL:GUID:ISOL NONE"
'Turn on auto orientation for the ECal (default behavior).
scpi.Parse "SENS:CORR:PREF:ECAL:ORI ON"
'Initialize an SMC guided calibration.
scpi.Parse "SENS:CORR:COLL:GUID:INIT"
'Tell the wizard to generate and report the number of steps in this cal.
Dim steps
Dim desc
'Determine the number of steps required to complete the calibration.
steps = scpi.Parse ("SENS:CORR:COLL:GUID:STEP?")
    For i = 1 To steps
    'Display the prompt for each step
    MsgBox (desc)
    'Perform the measurement for each step
    scpi.Parse "SENS:CORR:COLL:GUID:ACQ STANSTAN" & CStr(i)
    Next
Dim calset
'Finish the cal and save the calset
calset = scpi.Parse("SENS:CORR:COLL:GUID:SAVE OFF")
Msgbox("SMC cal saved to cal register")
Learning about COM

The following topics can help you learn more about controlling the VNA using COM.

- COM versus SCPI
- Configure for COM-DCOM Programming
- COM Fundamentals
- Getting a Handle to an Object
- COM Collections in the VNA
- VNA Automation Interfaces
- COM Data Types
- Working with VNA Events
- Read and Write Calibration Data using COM
- C++ and the COM Interface
- Using COM with .NET

See more VNA programming information and examples at: http://na.support.keysight.com/pna/programming/.
Learning about GPIB

The following topics can help you learn more about controlling the VNA using SCPI and the GPIB.

- COM versus SCPI
- GPIB Fundamentals
- The Rules and Syntax of SCPI Commands
- Configure for SCPI / SICL over GPIB
- Configure for VISA / SICL over LAN
- Calibrating the VNA Using SCPI
- Getting Data from the VNA using SCPI
- The VNA as a USB Device
- Remote Control of SCPI USB Devices Connected to a VNA
- Reading the VNA Status Registers
- New Referring to Traces, Measurements, Channels, and Windows Using SCPI
- Synchronizing the VNA and Controller
- Shut Down the VNA Remotely

See more VNA programming information and examples at: [http://na.support.keysight.com/pna/programming/](http://na.support.keysight.com/pna/programming/)
CFMixer Topic

File | Instrument | Response | Stimulus | Utility | Cal | Apps | Remote ONLY

FCA (Opt S9x082A / S9x083A/B): Configure | Calibrate | Embedded LO | Segment Sweep
Noise Figure App (Opt 028, 029, S93029A/B): Setup | Cal | SNP | ENR files | NFX
GCA (Opt S9x086A): Setup | Cal | Analysis | GCX
Swept IMD (Opt S93087A/B): Setup | Cal | Swept IMDx
IM Spectrum (Opt S93087A/B): Setup | IMx Spectrum
Diff IQ (Opt S9x089A/B): Meas | Setup | Range Settings | Parameters | X-Axis
SA (Opt S9x090xA): SA Setup | Source: Setup | Coherence | Advanced Settings | IF Setup | Trig. & Pulse
Setup | Processing Setup | ADC & LO Setup | Data | SA Markers | IQMod
VSA (Opt 095): VSA Setup | Source: Setup | IF Setup | Advanced Settings | Misc. VSA Commands
iTMSA (Opt S93460A/B): Setup | Phase Sweep
Common Mixer Setup commands
Pulse (PNA-X): Setup | Generators | Trigger
Frequency Offset (Opt S9x080A)
Enhanced Time Domain Analysis [GUI Command Finder] (TDR, Option S9x011A/B)
Active (Hot) Parameters (Opt S93110A/B)
Modulation Distortion (Opt S9x070xB)
Phase Noise (Opt. S93031xB)

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create an FCA measurement</td>
<td>CALCulate:MEASure:DEFine</td>
<td>App.CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Change an FCA measurement</td>
<td>CALCulate:MEASure:PARameter</td>
<td>Meas.ChangeParameter</td>
</tr>
<tr>
<td>Sweep type</td>
<td>SENSE:SWEep:TYPE</td>
<td>Sweep Type Property</td>
</tr>
<tr>
<td>Map DUT ports to PNA ports</td>
<td>SENSE:MIXer:PMAP</td>
<td>SetDutPorts</td>
</tr>
<tr>
<td>Read Input port map</td>
<td>SENSE:MIXer:PMAP:INPut</td>
<td>DeviceInputPort</td>
</tr>
</tbody>
</table>

FCA Configure Measurements

Use Mixer commands to setup the mixer.
Use FCA Segment commands to setup segment sweep.
Use the following commands to setup and calibrate SMC and VMC measurements.
<table>
<thead>
<tr>
<th>Read Output port map</th>
<th>SENSe:MIXer:PMAP:OUTPut</th>
<th>DeviceOutputPort</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avoid Spurs</td>
<td>SENSe:MIXer:AVoidspurs</td>
<td>AvoidSpurs</td>
</tr>
</tbody>
</table>

### Minimum required FCA (SMC and VMC) Cal commands

**SCPI and COM examples.**

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Cal object</td>
<td>None</td>
<td>CreateCustomCal</td>
</tr>
<tr>
<td>Begin Cal</td>
<td>SENSe:CORRection:COLLect:GUIDed:INITiate</td>
<td>Initialize</td>
</tr>
<tr>
<td>Set connector type for each port</td>
<td>SENSe:CORRection:COLLect:GUIDed:CONNECTor:PORT:SELeCT</td>
<td>ConnectorType</td>
</tr>
<tr>
<td>Set Cal Kit for each port</td>
<td>SENSe:CORRection:COLLect:GUIDed:CKIT:PORT:SELeCT</td>
<td>CalKitType</td>
</tr>
<tr>
<td>Returns the number of steps required by the Calibration</td>
<td>SENSe:CORRection:COLLect:GUIDed:STEPS?</td>
<td>GenerateSteps</td>
</tr>
<tr>
<td>Returns the step description.</td>
<td>SENSe:CORRection:COLLect:GUIDed:DESCRIPTION?</td>
<td>GetStepDescription</td>
</tr>
<tr>
<td>Measure a standard</td>
<td>SENSe:CORRection:COLLect:GUIDed:ACQuire</td>
<td>AcquireStep</td>
</tr>
<tr>
<td>Generate Error Terms</td>
<td>SENSe:CORRection:COLLect:GUIDed:SAVE</td>
<td>GenerateErrorTerms</td>
</tr>
</tbody>
</table>

### Optional FCA (SMC and VMC) commands

See **SMC** and **VMC** specific optional commands

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets ECAL auto-orientation ON or OFF</td>
<td>SENSe:CORRection:PREFERENCE:ECAL:ORIentation</td>
<td>AutoOrient</td>
</tr>
<tr>
<td>Sets ECAL port map when orientation is off.</td>
<td>SENSe:CORRection:PREFERENCE:ECAL:PMAP</td>
<td>EcalOrientation</td>
</tr>
<tr>
<td>Perform or Omit Isolation</td>
<td>SENSe:CORRection:COLLect:GUIDed:ISOLation</td>
<td>OmitIsolation</td>
</tr>
<tr>
<td>SMC Optional commands</td>
<td>Command Description</td>
<td>Property</td>
</tr>
<tr>
<td>------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Import existing Phase Ref Cal or Source Power Cal</td>
<td><code>SENSe:CORRection:COLLect:GUIDed:SMC:IMPort</code></td>
<td><code>ImportDataSet</code></td>
</tr>
<tr>
<td>Use Thru or perform separate power cals</td>
<td><code>SENSe:CORRection:COLLect:GUIDed:SMC:PWRCal:SEParate</code></td>
<td><code>SeparatePowerCal</code></td>
</tr>
<tr>
<td>Include Reverse SC12 sweep</td>
<td><code>SENSe:MIXer:REVerse</code></td>
<td><code>IncludeReverseSweep</code></td>
</tr>
<tr>
<td>Include input/output match correction</td>
<td><code>CALCulate:MEASure:CORRection:TYPE</code></td>
<td><code>meas.CalibrationTypeID</code></td>
</tr>
<tr>
<td>Use Nominal Incident Power</td>
<td><code>SENSe:MIXer:INPut:POWer:USENominal</code></td>
<td><code>NominalIncidentPowerSt</code> e</td>
</tr>
<tr>
<td>Save to s2p and s2px file</td>
<td><code>MMEMory:STORe</code></td>
<td>None</td>
</tr>
<tr>
<td>Load Power Table (used with mmWave)</td>
<td><code>SENSe:CORRection:COLLect:GUIDed:PSENsor:POWTable</code></td>
<td>None00</td>
</tr>
<tr>
<td>SMC Include Phase</td>
<td>SCPI</td>
<td>COM</td>
</tr>
<tr>
<td>-------------------</td>
<td>------</td>
<td>-----</td>
</tr>
<tr>
<td>Enable Phase</td>
<td>SENSe:MIXer:PHASe</td>
<td>EnablePhase</td>
</tr>
<tr>
<td>Set normalize point</td>
<td>SENSe:MIXer:NORMalize:POINt</td>
<td>NormalizePoint</td>
</tr>
<tr>
<td>Choose known delay or S2P</td>
<td>SENSe:CORRection:COLLect:GUIDed:SMC:PHASe:METHod</td>
<td>DelayCalculationMethod</td>
</tr>
<tr>
<td>Set known delay</td>
<td>SENSe:CORRection:COLLect:GUIDed:SMC:PHASe:DELay</td>
<td>FixedDelay</td>
</tr>
<tr>
<td>Set Cal Mixer Char S2P filename</td>
<td>SENSe:CORRection:COLLect:GUIDed:SMC:PHASe:MIXer</td>
<td>MixerCharacterizationFile</td>
</tr>
<tr>
<td>Use Phase Ref Cal Set</td>
<td>SENSe:CORRection:COLLect:GUIDed:SMC:IMPort</td>
<td>ImportDataSet0000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Perform SMC Phase Reference Cal</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>SYSTem:CALibrate:PHASe:RESet</td>
<td>Reset</td>
</tr>
<tr>
<td>Start Freq</td>
<td>SYSTem:CALibrate:PHASe:FREQuency:STARt</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop Freq</td>
<td>SYSTem:CALibrate:PHASe:FREQuency:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>Read channel number</td>
<td>SYSTem:CALibrate:PHASe:GUIDed:CHANnel?</td>
<td>N/A</td>
</tr>
<tr>
<td>Read all Phase Reference IDs</td>
<td>SYSTem:CALibrate:PHASe:REFerence:CATalog?</td>
<td>GetConnectedPhaseReferences</td>
</tr>
<tr>
<td>Set Phase Reference</td>
<td>SYSTem:CALibrate:PHASe:REFerence</td>
<td>PhaseReference</td>
</tr>
<tr>
<td>Set Source Attenuator</td>
<td>SYSTem:CALibrate:PHASe:POWer:ATTenuator</td>
<td>SourceAttenuator</td>
</tr>
<tr>
<td>Set Cal Set name</td>
<td>None</td>
<td>CalSet</td>
</tr>
<tr>
<td>Perform Cal</td>
<td>Guided Cal commands</td>
<td>Guided Cal commands</td>
</tr>
<tr>
<td>Use Phase Ref Cal Set</td>
<td>SENSe:CORRection:COLLect:GUIDed:SMC:IMPort</td>
<td>ImportDataSet</td>
</tr>
<tr>
<td>Unknown Mixer calibration state</td>
<td>SYSTem:CALibrate:PHASe:UNKNown:INCLude</td>
<td>IncludeUnknownMixer</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Property</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>Unknown mixer LO frequency</td>
<td>SYSTem:CALibrate:PHASe:UNKown:LO:FREQuency</td>
<td>UnknownMixerLOFrequency</td>
</tr>
<tr>
<td>Unknown mixer LO power</td>
<td>SYSTem:CALibrate:PHASe:UNKown:LO:POWer</td>
<td>UnknownMixerLOPower</td>
</tr>
<tr>
<td>Input power level to the unknown mixer</td>
<td>SYSTem:CALibrate:PHASe:UNKown:INPut:POWer</td>
<td>UnknownMixerInputPower</td>
</tr>
<tr>
<td>Connector type and gender of your Cal Kit.</td>
<td>SYSTem:CALibrate:PHASe:CONNector</td>
<td>ConnectorType PhaseRef</td>
</tr>
<tr>
<td>Cal Kit to be used to perform the S-parameter Cal</td>
<td>SYSTem:CALibrate:PHASe:CKIT</td>
<td>CalKitType PhaseRef</td>
</tr>
<tr>
<td>Set the ports to be calibrated.</td>
<td>SYSTem:CALibrate:PHASe:PORT[X]</td>
<td>IncludePort</td>
</tr>
<tr>
<td>Reverse Port2 Coupler</td>
<td>SYSTem:CALibrate:PHASe:DEEMbed</td>
<td>DeembedCoupler</td>
</tr>
</tbody>
</table>

**VMC Optional commands**


**Perform VMC Mixer characterization ONLY**

<p>| Perform full calibration or mixer characterization only                     | SENSE:CORRection:COLLect:GUIDed:VMC:OPERation                           | CharacterizeMixerOnly         |</p>
<table>
<thead>
<tr>
<th>Mixer Characterization</th>
<th>SCPI Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets ECAL auto-orientation ON or OFF</td>
<td>SENSE:CORRection:PREFerence:ECAL:ORientation</td>
<td>AutoOrient Property</td>
</tr>
<tr>
<td>Sets the port mapping for the mixer characterization.</td>
<td>SENSE:CORRection:COLLect:GUIDed:VMC:MIXer:ECAL:PORTmap</td>
<td>EcalOrientation1Port</td>
</tr>
</tbody>
</table>

**FCA Segment Sweep**

**SCPI and COM examples.**

<table>
<thead>
<tr>
<th>Operation</th>
<th>SCPI Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recalculate</td>
<td>SENSE:MIXer:RECalculate</td>
<td>ReCalculate</td>
</tr>
<tr>
<td>Segment Calculate</td>
<td>SENSE:MIXer:SEGment:CALCulate</td>
<td>SegmentCalculate</td>
</tr>
<tr>
<td>Query Count</td>
<td>SENSE:MIXer:SEGment:COUNt?</td>
<td>SegmentCount</td>
</tr>
<tr>
<td>Add Segments</td>
<td>SENSE:MIXer:SEGment:ADD</td>
<td>AddSegment</td>
</tr>
<tr>
<td>Delete Segments</td>
<td>SENSE:MIXer:SEGment:DEL</td>
<td>DeleteSegment</td>
</tr>
<tr>
<td>Remove All Segments</td>
<td>SENSE:MIXer:SEGment:DELeTe:ALL</td>
<td>DeleteAllSegments</td>
</tr>
<tr>
<td>State</td>
<td>SENSE:MIXer:SEGment[:STATe]</td>
<td>SegmentState</td>
</tr>
<tr>
<td>Number Of Points</td>
<td>SENSE:MIXer:SEGment:POINts</td>
<td>SegmentPoints</td>
</tr>
<tr>
<td>IFBW</td>
<td>SENSE:MIXer:SEGment:BWIDth</td>
<td>SegmentIFBandwidth</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Explanation</td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Input Fixed Freq</td>
<td>SENSE: MIXer: SEGment: INPut: FREQuency: FIXed</td>
<td>SegmentFixedFrequency</td>
</tr>
<tr>
<td>Input Start Freq</td>
<td>SENSE: MIXer: SEGment: INPut: FREQuency: START</td>
<td>SegmentStartFrequency</td>
</tr>
<tr>
<td>Input Stop Freq</td>
<td>SENSE: MIXer: SEGment: INPut: FREQuency: STOP</td>
<td>SegmentStopFrequency</td>
</tr>
<tr>
<td>Input Power</td>
<td>SENSE: MIXer: SEGment: INPut: POWER</td>
<td>SegmentFixedPower</td>
</tr>
<tr>
<td>Output Fixed Freq</td>
<td>SENSE: MIXer: SEGment: OUTPut: FREQuency: FIXed</td>
<td>SegmentFixedFrequency</td>
</tr>
<tr>
<td>Output Start Freq</td>
<td>SENSE: MIXer: SEGment: OUTPut: FREQuency: START</td>
<td>SegmentStartFrequency</td>
</tr>
<tr>
<td>Output Stop Freq</td>
<td>SENSE: MIXer: SEGment: OUTPut: FREQuency: STOP</td>
<td>SegmentStopFrequency</td>
</tr>
<tr>
<td>Output (+/-)</td>
<td>SENSE: MIXer: SEGment: OUTPut: FREQuency: SIDeband</td>
<td>SegmentMixingMode</td>
</tr>
<tr>
<td>Output Power</td>
<td>SENSE: MIXer: SEGment: OUTPut: POWER</td>
<td>SegmentFixedPower</td>
</tr>
<tr>
<td>LO Fixed Freq</td>
<td>SENSE: MIXer: SEGment: LO&lt;x&gt;: FREQuency: FIXed</td>
<td>SegmentFixedFrequency</td>
</tr>
<tr>
<td>LO Start Freq</td>
<td>SENSE: MIXer: SEGment: LO&lt;x&gt;: FREQuency: START</td>
<td>SegmentStartFrequency</td>
</tr>
<tr>
<td>LO Stop Freq</td>
<td>SENSE: MIXer: SEGment: LO&lt;x&gt;: FREQuency: STOP</td>
<td>SegmentStopFrequency</td>
</tr>
<tr>
<td>LO Fixed/Swept</td>
<td>SENSE: MIXer: SEGment: LO&lt;x&gt;: FREQuency: MODE</td>
<td>SegmentRangeMode</td>
</tr>
<tr>
<td>Input &gt;LO</td>
<td>SENSE: MIXer: SEGment: LO&lt;x&gt;: FREQuency: I&lt;LT</td>
<td>SegmentIsInputGreaterThanLO</td>
</tr>
<tr>
<td>LO Power</td>
<td>SENSE: MIXer: SEGment: LO&lt;x&gt;: POWER</td>
<td>SegmentFixedPower</td>
</tr>
<tr>
<td>IF (+/-)</td>
<td>SENSE: MIXer: SEGment: IF: FREQuency: SIDeband</td>
<td>SegmentMixingMode</td>
</tr>
</tbody>
</table>

**Embedded LO (used with FCA, NFX, GCX, MODX, and Swept IMDx)**

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedded LO ON</td>
<td>OFF</td>
<td>SENSE: MIXer:ELO: STATE</td>
</tr>
<tr>
<td>Select tuning point</td>
<td>SENSE: MIXer:ELO: NORMALize: POINT</td>
<td>NormalizePoint</td>
</tr>
<tr>
<td>Set tuning mode</td>
<td>SENSE: MIXer:ELO: TUNing: MODE</td>
<td>TuningMode</td>
</tr>
<tr>
<td>Feature Description</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>Set broadband sweep span</td>
<td><code>SENSe:MIXer:ELO:TUNing:SPAN</code></td>
<td>BroadbandTuningSpan</td>
</tr>
<tr>
<td>Set precise tuning tolerance</td>
<td><code>SENSe:MIXer:ELO:TUNing:TOLerance</code></td>
<td>PreciseTuningTolerance</td>
</tr>
<tr>
<td>Set precise tuning iterations</td>
<td><code>SENSe:MIXer:ELO:TUNing:ITERations</code></td>
<td>MaxPreciseTuningIterations</td>
</tr>
<tr>
<td>LO delta frequency</td>
<td><code>SENSe:MIXer:ELO:LO:DELTA</code></td>
<td>LOFrequencyDelta</td>
</tr>
<tr>
<td>Resets tuning parameters</td>
<td><code>SENSe:MIXer:ELO:TUNing:RESet</code></td>
<td>ResetTuningParameters</td>
</tr>
<tr>
<td>Reset LO Delta frequency</td>
<td><code>SENSe:MIXer:ELO:LO:RESET</code></td>
<td>ResetLOFrequency</td>
</tr>
<tr>
<td>Reset LO Frequency Delta and Tuning parameters to their default settings.</td>
<td><code>SENSe:MIXer:ELO:RESET</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the Noise Bandwidth for Broadband and Precise tuning sweeps.</td>
<td><code>SENSe:MIXer:ELO:TUNing:NBW</code></td>
<td>None</td>
</tr>
<tr>
<td>Embedded LO Diagnostics (Available with Swept IMDx using COM ONLY)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clear current diagnostic information</td>
<td><code>SENSe:MIXer:ELO:DIAGnostic:CLEAr</code></td>
<td>Clear</td>
</tr>
<tr>
<td>Get result of the last tuning sweeps.</td>
<td><code>SENSe:MIXer:ELO:DIAGnostic:STATus?</code></td>
<td>StatusAsString</td>
</tr>
<tr>
<td>Get number of tuning sweeps.</td>
<td><code>SENSe:MIXer:ELO:DIAGnostic:SWEep:COUNt?</code></td>
<td>NumberOfSweeps</td>
</tr>
<tr>
<td>Was a marker was used for a tuning sweep?</td>
<td><code>SENSe:MIXer:ELO:DIAGnostic:SWEep:MARKer:STATe?</code></td>
<td>IsMarkerOn</td>
</tr>
<tr>
<td>Get the marker X-axis position.</td>
<td>SENSe:MIXer:ELO:DIAGnostic:SWEep:MARKer:POSi</td>
<td>MarkerPosition</td>
</tr>
<tr>
<td>Get the marker annotation.</td>
<td>SENSe:MIXer:ELO:DIAGnostic:SWEep:MARKer:ANNotation?</td>
<td>MarkerAnnotation</td>
</tr>
<tr>
<td>Get the tuning sweep X axis annotation.</td>
<td>SENSe:MIXer:ELO:DIAGnostic:SWEep:X:ANNotation?</td>
<td>XAxisAnnotation</td>
</tr>
<tr>
<td>Get the tuning sweep Y axis annotation.</td>
<td>SENSe:MIXer:ELO:DIAGnostic:SWEep:Y:ANNotation?</td>
<td>YAxisAnnotation</td>
</tr>
<tr>
<td>Get the Start sweep value.</td>
<td>SENSe:MIXer:ELO:DIAGnostic:SWEep:X:STARt?</td>
<td>XAxisStart</td>
</tr>
<tr>
<td>Get the Stop sweep value.</td>
<td>SENSe:MIXer:ELO:DIAGnostic:SWEep:X:STOP?</td>
<td>XAxisStop</td>
</tr>
<tr>
<td>Returns the tuning sweep parameter name.</td>
<td>SENSe:MIXer:ELO:DIAGnostic:SWEep:PARameter?</td>
<td>Parameter</td>
</tr>
<tr>
<td>Returns the tuning sweep title.</td>
<td>SENSe:MIXer:ELO:DIAGnostic:SWEep:TITLe?</td>
<td>StepTitle</td>
</tr>
<tr>
<td>Returns the LO frequency delta from this tuning sweep</td>
<td>SENSe:MIXer:ELO:DIAGnostic:SWEep:LO:DELTa?</td>
<td>LODeltaFound</td>
</tr>
</tbody>
</table>

**Gain Compression Setup**

<p>| Create a GCA Meas | CALCulate:MEASure:DEFine | App.CreateCustomMeasurementEx |
| Change a GCA Parameter | CALCulate:MEASure:PARameter | Meas.ChangeParameter |
| Sweep Type | SENSe:SWEep:TYPE | SweepType |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Command</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of frequency points</td>
<td>SENSe:GCSetup:SWEep:FREQuency:POINts</td>
<td>NumberOfFrequencyPoints</td>
</tr>
<tr>
<td>Number of power points</td>
<td>SENSe:GCSetup:SWEep:POWer:POINts</td>
<td>NumberOfPowerPoints</td>
</tr>
<tr>
<td>Maximum number of points</td>
<td>None</td>
<td>MaximumNumberOfPoints</td>
</tr>
<tr>
<td>Total number of points</td>
<td>None</td>
<td>TotalNumberOfPoints</td>
</tr>
<tr>
<td>Acquisition mode</td>
<td>SENSe:GCSetup:AMODe</td>
<td>AcquisitionMode</td>
</tr>
<tr>
<td>Smart tolerance</td>
<td>SENSe:GCSetup:SMARt:TOLerance</td>
<td>SmartSweepTolerance</td>
</tr>
<tr>
<td>Smart Iterations</td>
<td>SENSe:GCSetup:SMARt:MITerations</td>
<td>SmartSweepMaximumIterations</td>
</tr>
<tr>
<td>Smart settling time</td>
<td>SENSe:GCSetup:SMARt:STIMe</td>
<td>SmartSweepSettlingTime</td>
</tr>
<tr>
<td>Smart show iterations</td>
<td>SENSe:GCSetup:SMARt:SITerations</td>
<td>SmartSweepShowIterations</td>
</tr>
<tr>
<td>Read DC at compression point</td>
<td>SENSe:GCSetup:SMARt:CDC</td>
<td>ReadDCAtCompression</td>
</tr>
<tr>
<td>Read compression failures</td>
<td>SENSe:GCSetup:SFAilures?</td>
<td>SearchFailures</td>
</tr>
<tr>
<td>Write port map</td>
<td>SENSe:GCSetup:PMAP</td>
<td>SetPortMap</td>
</tr>
<tr>
<td>Read Port Map (Input)</td>
<td>SENSe:GCSetup:PMAP:INPut</td>
<td>DeviceInputPort</td>
</tr>
<tr>
<td>Read Port Map (Output)</td>
<td>SENSe:GCSetup:PMAP:OUTPut</td>
<td>DeviceOutputPort</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>Remaps the source port to Port 1.</td>
<td><code>SENSe:GCSetup:PMAP:SOURce:OVERride</code></td>
<td>None</td>
</tr>
<tr>
<td>End of Sweep</td>
<td><code>SENSe:GCSetup:EOSoperation</code></td>
<td><code>EndOfSweepOperation</code></td>
</tr>
<tr>
<td>Linear input power</td>
<td><code>SENSe:GCSetup:POWer:LINear:INPut:LEVel</code></td>
<td><code>InputLinearPowerLevel</code></td>
</tr>
<tr>
<td>Reverse Power</td>
<td><code>SENSe:GCSetup:POWer:REVerse:LEVel</code></td>
<td><code>ReverseLinearPowerLevel</code></td>
</tr>
<tr>
<td>Start power</td>
<td><code>SENSe:GCSetup:POWer:STARt:LEVel</code></td>
<td><code>chan.Start Power</code></td>
</tr>
<tr>
<td>Stop power</td>
<td><code>SENSe:GCSetup:POWer:STOP:LEVel</code></td>
<td><code>chan.Stop Power</code></td>
</tr>
<tr>
<td>Compression algorithm</td>
<td><code>SENSe:GCSetup:COMPression:ALGorithm</code></td>
<td><code>CompressionAlgorithm</code></td>
</tr>
<tr>
<td>Compression Level</td>
<td><code>SENSe:GCSetup:COMPression:LEVel</code></td>
<td><code>CompressionLevel</code></td>
</tr>
<tr>
<td>Backoff Level</td>
<td><code>SENSe:GCSetup:COMPression:BACK:LEVel</code></td>
<td><code>CompressionBackoff</code></td>
</tr>
<tr>
<td>X Delta</td>
<td><code>SENSe:GCSetup:COMPression:DELTa:X</code></td>
<td><code>CompressionDeltaX</code></td>
</tr>
<tr>
<td>Y Delta</td>
<td><code>SENSe:GCSetup:COMPression:DELTa:Y</code></td>
<td><code>CompressionDeltaY</code></td>
</tr>
<tr>
<td>Saturation level</td>
<td><code>SENSe:GCSetup:COMPression:SATuration:LEVel</code></td>
<td><code>SaturationLevel</code></td>
</tr>
<tr>
<td>Interpolation</td>
<td><code>SENSe:GCSetup:COMPression:INTerpolate[:STATe]</code></td>
<td><code>CompressionInterpolation</code></td>
</tr>
<tr>
<td>Set and read the desired phase to measure compression.</td>
<td><code>SENSe:GCSetup:COMPression:PHASe:LEVel</code></td>
<td>None</td>
</tr>
<tr>
<td>Set and read compression format to be either magnitude, phase, or</td>
<td><code>SENSe:GCSetup:COMPression:PHASe:MODE</code></td>
<td>None</td>
</tr>
<tr>
<td>Safe Sweep enable</td>
<td>SENSe:GCSetup:SAFE:ENABLE</td>
<td>SafeSweepEnable</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Safe Sweep coarse</td>
<td>SENSe:GCSetup:SAFE:CPADjustment</td>
<td>SafeSweepCoarsePowerAdjustment</td>
</tr>
<tr>
<td>Safe Sweep fine</td>
<td>SENSe:GCSetup:SAFE:FPADjustment</td>
<td>SafeSweepFinePowerAdjustment</td>
</tr>
<tr>
<td>Safe Sweep threshold</td>
<td>SENSe:GCSetup:SAFE:FTHReshold</td>
<td>SafeSweepFineThreshold</td>
</tr>
<tr>
<td>Safe Sweep max power</td>
<td>SENSe:GCSetup:SAFE:MLIMit</td>
<td>SafeSweepMaximumLimit</td>
</tr>
<tr>
<td>Set and read the name of the external DC device.</td>
<td>SENSe:GCSetup:SAFE:DC:PARameter</td>
<td>SafeSweepDCParameter</td>
</tr>
<tr>
<td>Set and read the maximum limit of the external DC device.</td>
<td>SENSe:GCSetup:SAFE:DC:MLimit</td>
<td>SafeSweepMaximumDCLimit</td>
</tr>
<tr>
<td>Set and read the DC readings at the compression point in the last iteration of a smart sweep</td>
<td>SENSe:GCSetup:SMARt:CDC</td>
<td>None</td>
</tr>
</tbody>
</table>

**GCA/GCX Phase Commands**

| Set and read the state of the mixer reference. | SENSe:GCSetup:MIXer:REFerence | None |

8374
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set and read the aperture to use when computing the linear input power.</td>
<td>SENSE:GCSetup:POWer:LINear:INPut:COMPute:APERture</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the state of the power smoothing.</td>
<td>SENSE:GCSetup:SWEep:POWer:SMOoth</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the power smoothing aperture in percent.</td>
<td>SENSE:GCSetup:SWEep:POWer:SMOoth:APERture</td>
<td>None</td>
</tr>
<tr>
<td>Read GCA Data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read all GCA data</td>
<td>CALCulate:MEASure:GCData:DATA?</td>
<td>GetRaw2DData</td>
</tr>
<tr>
<td>Read real GCA data</td>
<td>CALCulate:MEASure:GCData:REAL?</td>
<td>GetDataRe</td>
</tr>
<tr>
<td>Read imaginary GCA data</td>
<td>CALCulate:MEASure:GCData:IMAG?</td>
<td>GetDataIm</td>
</tr>
<tr>
<td>Read number of iterations</td>
<td>CALCulate:MEASure:GCD:ITERations?</td>
<td>TotalIterations</td>
</tr>
</tbody>
</table>

| Gain Compression Analysis                                                   |                                                                          |           |
| Enable a compression analysis trace                                         | CALCulate:MEASure:GCMeas:ANALysis:ENABle                                 | AnalysisEnable |
| Set CW frequency                                                            | CALCulate:MEASure:GCMeas:ANALysis:CWFRequency                           | AnalysisCWFreq |
| Set to discrete or interpolated CW frequencies                              | CALCulate:MEASure:GCMeas:ANALysis:DISCrete                               | AnalysisIsDiscreteFreq |
### Sets X-axis display

**CALCulate:MEASure:GCMes:ANALysis:XAXis**  
**AnalysisXAxis**

### GCA Cal

<table>
<thead>
<tr>
<th>Activity</th>
<th>Command</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read about GCA Cal</td>
<td>Gain Compression Cal</td>
<td>GainCompressionCal</td>
</tr>
<tr>
<td>Set power for source power cal</td>
<td>SENSe:CORRection:GCSetup:POWer</td>
<td>PowerLevel</td>
</tr>
<tr>
<td>Set connector type for power sensor.</td>
<td>SENSe:CORRection:GCSetup:SENSor:CONNector</td>
<td>PowerSensorConnectorType</td>
</tr>
<tr>
<td>Set cal kit for power sensor connector.</td>
<td>SENSe:CORRection:GCSetup:SENSor:CKIT</td>
<td>PowerSensorCalKitType</td>
</tr>
<tr>
<td>Guided Cal commands</td>
<td>SENSe:CORRection:COLLect:GUIDed</td>
<td>Guided Calibration</td>
</tr>
</tbody>
</table>

### GCx- Gain Compression for Converters

Use [Gain Compression commands](#) to setup the GCA measurement.

Use [Mixer commands](#) to setup the mixer.

See [SCPI](#) and [COM](#) examples

### Noise Figure Setup

<table>
<thead>
<tr>
<th>Activity</th>
<th>Command</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Noise Figure meas</td>
<td>CALCulate:CUSTOM:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Change Noise Figure meas</td>
<td>CALCulate:CUSTOM:MODify</td>
<td>meas.Change Parameter</td>
</tr>
<tr>
<td>Sets the number of impedance states to use</td>
<td>SENSe:NOISE:IMPedance:COUNt</td>
<td>ImpedanceStates</td>
</tr>
<tr>
<td>Select Noise Receiver</td>
<td>SENSe:NOIsE:REceiver</td>
<td>NoiseReceiver</td>
</tr>
<tr>
<td>Noise averaging ON and OFF</td>
<td>SENSe:NOIsE:AVERage:STATE</td>
<td>NoiseAverageState</td>
</tr>
<tr>
<td>Set averaging of noise receiver.</td>
<td>SENSe:NOIsE:AVERage</td>
<td>NoiseAverageFactor</td>
</tr>
<tr>
<td>Set narrowband state</td>
<td>SENSe:NOIsE:NARRowband</td>
<td>NarrowBand</td>
</tr>
<tr>
<td>Function Description</td>
<td>Command</td>
<td>Result</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------------</td>
<td>-------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Set bandwidth of noise receiver.</td>
<td>SENSE:NOISe:BWIDth</td>
<td>NoiseBandwidth</td>
</tr>
<tr>
<td>Set gain state of noise receiver.</td>
<td>SENSE:NOISe:GAIN</td>
<td>NoiseGain</td>
</tr>
<tr>
<td>Sets the port identifier of the ECa1 noise tuner that is connected to the PNA Source.</td>
<td>SENSE:NOISe:TUNer:INPut</td>
<td>NoiseTunerIn</td>
</tr>
<tr>
<td>Sets the port identifier of the ECa1 noise tuner that is connected to the DUT.</td>
<td>SENSE:NOISe:TUNer:OUTPut</td>
<td>NoiseTunerOut</td>
</tr>
<tr>
<td>Sets the state of the custom noise tuner file.</td>
<td>SENSE:NOISe:TUNer:FILE:[STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Set and read a custom noise tuner file to be used instead of the one generated automatically based on the state.</td>
<td>SENSE:NOISe:TUNer:FILE:NAME</td>
<td>None</td>
</tr>
<tr>
<td>Sets the state of the custom noise tuner file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set the excess noise source ON or OFF.</td>
<td>CONTROL:NOISe:SOURce or OUTPUT:MANual:NOISe:[STATE]</td>
<td>NoiseSourceState</td>
</tr>
<tr>
<td>Set mechanical switches</td>
<td>SENSE:PATH:CONFig:ELEMent</td>
<td></td>
</tr>
</tbody>
</table>

**Port Mapping - Noise Figure Opt 028**

<table>
<thead>
<tr>
<th>Function Description</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write port mapping</td>
<td>SENSE:NOISe:PMAP</td>
<td>SetPortMap</td>
</tr>
<tr>
<td>Read input port mapping</td>
<td>SENSE:NOISe:PMAP:INPut?</td>
<td>DeviceInputPort</td>
</tr>
<tr>
<td>Read output port mapping</td>
<td>SENSE:NOISe:PMAP:OUTPut?</td>
<td>DeviceOutputPort</td>
</tr>
</tbody>
</table>

**Noise Figure Cal - all other cal settings use standard commands.**

<table>
<thead>
<tr>
<th>Function Description</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Noise Cal object</td>
<td>N/A</td>
<td>CreateCustomCalEx</td>
</tr>
<tr>
<td>Set Noise Calibration method</td>
<td>SENSE:NOISe:CALibration:METHOD</td>
<td>CalMethod</td>
</tr>
<tr>
<td>Set Noise Tuner identifier</td>
<td>SENSE:NOISe:TUNer:ID</td>
<td>NoiseTuner</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Set Noise Recvr Method</td>
<td>SENSE:NOISe:CALibration:RMEthod</td>
<td>RcvCharMethod</td>
</tr>
<tr>
<td>Noise source ENR filename</td>
<td>SENSE:NOISe:ENR:FIleName</td>
<td>ENRFilename</td>
</tr>
<tr>
<td>Set noise source Cal Kit type</td>
<td>SENSE:NOISe:SOURce:CKIT</td>
<td>NoiseSourceCalKitType</td>
</tr>
<tr>
<td>Set ambient temperature</td>
<td>SENSE:NOISe:TEMPerature:AMBient</td>
<td>AmbientTemperature</td>
</tr>
<tr>
<td>Sets noise source connector type</td>
<td>SENSE:NOISe:SOURce:CONNector</td>
<td>NoiseSourceConnectorType</td>
</tr>
<tr>
<td>Set noise source connector temp</td>
<td>SENSE:CORRection:TCOLd:USER:VALue</td>
<td>NoiseSourceCold</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Get/Save Noise Parameters SNP Data</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read noise parameter SNP data</td>
<td>SENSE:NOISe:SNP?</td>
<td>GetSnPData</td>
</tr>
<tr>
<td>Write noise parameters to SNP file</td>
<td>SENSE:NOISe:SNP:SAVE</td>
<td>WriteSnPData</td>
</tr>
</tbody>
</table>

**NFX - Noise Figure on Mixers/Converters**

Use Noise Figure commands to setup Noise Figure measurement.

Use Mixer commands to setup the mixer.

Use the following commands to calibrate.

See SCPI and COM examples.

<table>
<thead>
<tr>
<th>Sets the power sensor connector</th>
<th>SENSE:CORRection:COLLect:GUIDed:PSENsor:CONNector</th>
<th>PowerSensorConnectorType</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets the power sensor calkit</td>
<td>SENSE:CORRection:COLLect:GUIDed:PSENsor:CKIT</td>
<td>PowerSensorCalKitType</td>
</tr>
</tbody>
</table>

8378
<table>
<thead>
<tr>
<th>source power cal</th>
<th>SENSE:NOISE:TUNer:ORIent</th>
<th>AutoOrientTuner</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets auto orientation state for noise tuner</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets LO power calibration state</td>
<td>SENSE:CORRection:COLLect:NOISE:LO:PCAL:STATE</td>
<td>EnableLOPowerCal</td>
</tr>
<tr>
<td>Sets the source pull technique to compute DUT S-parameters</td>
<td>SENSE:NOISE:PULL</td>
<td>SourcePullForSParameters</td>
</tr>
<tr>
<td>Sets the state of ENR adapter de-embedding.</td>
<td>SENSE:CORRection:COLLect:NOISE:ENR:ADAP:DEEmbed</td>
<td>ForceDeEmbedENRAdapter</td>
</tr>
<tr>
<td>Set and read the state of the thru adapter de-embedding.</td>
<td>SENSE:CORRection:COLLect:NOISE:THRU:ADAPter:DEEmbed[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Sets the state of Power Sensor adapter de-embedding.</td>
<td>SENSE:CORRection:COLLect:NOISE:PSEN:ADAP:DEEmbed</td>
<td>ForceDeEmbedSensorAdapter</td>
</tr>
</tbody>
</table>

### Noise Figure ENR File Data Management

<table>
<thead>
<tr>
<th>Set ENR calibration data.</th>
<th>SENSE:CORRection:ENR:CALibration:TABLE:DATA</th>
<th>PutENRData</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>Command</td>
<td>Function</td>
</tr>
<tr>
<td>---------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>Read ENR calibration data.</td>
<td>SENSE:CORRection:ENR:CALibration:TABLE:DATA?</td>
<td>GetENRData</td>
</tr>
<tr>
<td>Get/set ID of ENR table.</td>
<td>SENSE:CORRection:ENR:CALibration:TABLE:ID:DATA</td>
<td>ENRID</td>
</tr>
<tr>
<td>Get/set serial number of noise source.</td>
<td>SENSE:CORRection:ENR:CALibration:TABLE:SERial:DATA</td>
<td>ENRSN</td>
</tr>
<tr>
<td>Load ENR table from file.</td>
<td>MMEMory:LOAD:ENR</td>
<td>LoadENRFile</td>
</tr>
<tr>
<td>Save ENR table to file.</td>
<td>MMEMory:STORe:ENR</td>
<td>SaveENRFile</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Swept IMD</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a measurement</td>
<td>CALCulate:CUSTom:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Change a measurement</td>
<td>CALCulate:CUSTom:MODify</td>
<td>meas.Change Parameter</td>
</tr>
<tr>
<td>Frequency Tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set sweep type</td>
<td>SENSE:IMD:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Set DeltaF</td>
<td>SENSE:IMD:FREQuency:DFRequency</td>
<td>DeltaFrequency</td>
</tr>
<tr>
<td>Set center freq</td>
<td>SENSE:IMD:FREQuency:FCENter</td>
<td>FrequencyCenter</td>
</tr>
<tr>
<td>Start for center freq sweep</td>
<td>SENSE:IMD:FREQuency:FCENter:STARt</td>
<td>FrequencyCenterStart</td>
</tr>
<tr>
<td>Stop for center freq sweep</td>
<td>SENSE:IMD:FREQuency:FCENter:STOP</td>
<td>FrequencyCenterStop</td>
</tr>
<tr>
<td>Center for center freq sweep</td>
<td>SENSE:IMD:FREQuency:FCENter:CENTer</td>
<td>FrequencyCenterCenter</td>
</tr>
<tr>
<td>Span for center freq sweep</td>
<td>SENSE:IMD:FREQuency:FCENter:SPAN</td>
<td>FrequencyCenterSpan</td>
</tr>
<tr>
<td>Start for DeltaF sweep</td>
<td>SENSE:IMD:FREQuency:DFRequency:STARTt</td>
<td>DeltaFrequencyStart</td>
</tr>
<tr>
<td>Stop for DeltaF sweep</td>
<td>SENSE:IMD:FREQuency:DFRequency:STOP</td>
<td>DeltaFrequencyStop</td>
</tr>
<tr>
<td>Set F1 for CW and Power sweep</td>
<td>SENSE:IMD:FREQuency:F1</td>
<td>F1Frequency</td>
</tr>
<tr>
<td>Set F2 for CW and Power sweep</td>
<td>SENSE:IMD:FREQuency:F2</td>
<td>F2Frequency</td>
</tr>
<tr>
<td>Set main tone IFBW</td>
<td>SENSE:IMD:IFBWidth:MAIN</td>
<td>MainToneIFBandwidth</td>
</tr>
<tr>
<td>Set product tones IFBW</td>
<td>SENSE:IMD:IFBWidth:IMTone</td>
<td>IMToneIFBandwidth</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Power Tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enables power coupling for F1 and F2</td>
<td>SENSE:IMD:TPOWER:COUP</td>
<td>CoupleTonePower</td>
</tr>
<tr>
<td>Set power level for F1 tone</td>
<td>SENSE:IMD:TPOWER:F1</td>
<td>TonePower</td>
</tr>
<tr>
<td>Set power level for F2 tone</td>
<td>SENSE:IMD:TPOWER:F2</td>
<td>TonePower</td>
</tr>
<tr>
<td>F1 start for power sweep</td>
<td>SENSE:IMD:TPOWER:F1:STARt</td>
<td>TonePowerStart</td>
</tr>
<tr>
<td>F1 stop for power sweep</td>
<td>SENSE:IMD:TPOWER:F1:STOP</td>
<td>TonePowerStop</td>
</tr>
<tr>
<td>F2 start for power sweep</td>
<td>SENSE:IMD:TPOWER:F2:STARt</td>
<td>TonePowerStart</td>
</tr>
<tr>
<td>F2 stop for power sweep</td>
<td>SENSE:IMD:TPOWER:F2:STOP</td>
<td>TonePowerStop</td>
</tr>
<tr>
<td>Set power leveling mode</td>
<td>SENSE:IMD:TPOWER:LEV</td>
<td>LevelingMethod</td>
</tr>
<tr>
<td>Set source attenuation</td>
<td>SOURce:POWer:ATTenuation</td>
<td>Attenuator</td>
</tr>
<tr>
<td>Set receiver attenuation</td>
<td>SENSE:POWer:ATTenuator</td>
<td>ReceiverAttenuator</td>
</tr>
<tr>
<td>Configure Tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set external source for f2</td>
<td>SENSE:ROLE:DEVice “RF2”</td>
<td>RoleDevice “RF2”</td>
</tr>
<tr>
<td>For CTB, CSO, and XMod parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Normalization Mode</td>
<td>SENSE:IMD:NORMalized:MODE</td>
<td>CompositeNormalizationMode</td>
</tr>
<tr>
<td>Normalized CSO power</td>
<td>SENSE:IMD:CSO:NORMalized:POWer</td>
<td>CompositeNormalizedCSOPower</td>
</tr>
<tr>
<td>CSO Offset</td>
<td>SENSE:IMD:CSO:OFFSet</td>
<td>CSOOffset</td>
</tr>
<tr>
<td>CSO Number of Distortion products</td>
<td>SENSE:IMD:CSO:NDPRoducts</td>
<td>CSONumDistortionProducts</td>
</tr>
<tr>
<td>Normalized CTB power</td>
<td>SENSE:IMD:CTB:NORMalized:POWer</td>
<td>CompositeNormalizedCTBPower</td>
</tr>
<tr>
<td>CTB and XMod Number of carriers</td>
<td>SENSE:IMD:CTB:NCARRriers</td>
<td>CTBXMODNumCarriers</td>
</tr>
<tr>
<td>CTB Offset</td>
<td>SENSE:IMD:CTB:OFFSet</td>
<td>CTBOffset</td>
</tr>
</tbody>
</table>
Swept IMD Calibration

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>SCPI Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Cal Mode</td>
<td>Sensing Correction: IMD: Calibration: Method</td>
<td>SENSe:CORRection:IMD:CALibration:METHOD</td>
</tr>
<tr>
<td>Set Cal frequencies</td>
<td>Sensing Correction: IMD: Calibration: Frequencies</td>
<td>SENSe:CORRection:IMD:CALibration:FREQuencies</td>
</tr>
<tr>
<td>Max Products</td>
<td>Sensing Correction: IMD: MPRoduct</td>
<td>SENSe:CORRection:IMD:MPRoduct</td>
</tr>
<tr>
<td>Set power</td>
<td>Sensing Correction: IMD: Power</td>
<td>SENSe:CORRection:IMD:POWer</td>
</tr>
<tr>
<td>Sensor Cal Kit</td>
<td>Sensing Correction: IMD: SENSOR: CKIT</td>
<td>SENSe:CORRection:IMD:SENSor:CKIT</td>
</tr>
<tr>
<td>Sensor connector</td>
<td>Sensing Correction: IMD: SENSOR: CONNECTor</td>
<td>SENSe:CORRection:IMD:SENSor:CONNector</td>
</tr>
<tr>
<td>Include 2nd order</td>
<td>Sensing Correction: IMD: SORDer: INCLude</td>
<td>SENSe:CORRection:IMD:SORDer:INCLude</td>
</tr>
<tr>
<td>IMDx Cal</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Swept IMDx (IMD on Mixers/Converters)

Use the Swept IMD commands to configure the IMD measurement.

Use the Mixer Setup commands to configure the mixer.

Use the IMD Cal commands to calibrate.

See SCPI and COM examples

IM Spectrum

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>SCPI Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a measurement</td>
<td>Calculate: CUSTOM: DEFINE</td>
<td>CALCulate:CUSTom:DEFine</td>
</tr>
<tr>
<td>Change a measurement</td>
<td>Calculate: CUSTOM: MODIFY</td>
<td>CALCulate:CUSTom:MODify</td>
</tr>
<tr>
<td>Enable tracking with IMD</td>
<td>Sensing: IMS: TRACKing: STATE</td>
<td>SENSe:IMS:TRACking:STATe</td>
</tr>
<tr>
<td>Set IMD channel to track</td>
<td>Sensing: IMS: TRACKing: CHANNEL</td>
<td>SENSe:IMS:TRACking:CHANnel</td>
</tr>
<tr>
<td>Set sweep type</td>
<td>Sensing: IMS: SWEep: TYPE</td>
<td>SENSe:IMS:SWEep:TYPE</td>
</tr>
</tbody>
</table>
### IMxSpectrum (IM Spectrum on Mixers/Converters)

Use the [Mixer Setup commands](#) to configure the mixer.

Use the [IMSpectrum commands](#) to configure the measurement.

Learn about [IMxSpectrum calibration](#).

See [SCPI](#) and COM examples.

<table>
<thead>
<tr>
<th>Setting</th>
<th>SCPI Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set product to view</td>
<td><code>SENSe:IMS:SWEep:ORDer</code></td>
<td>SweepOrder</td>
</tr>
<tr>
<td>Set step sweep</td>
<td><code>SENSe:IMS:TRACking:MSENable</code></td>
<td>TrackingManualStepEnable</td>
</tr>
<tr>
<td>Set step index</td>
<td><code>SENSe:IMS:TRACking:SINDex</code></td>
<td>TrackingStepIndex</td>
</tr>
<tr>
<td>Set Res Bandwidth</td>
<td><code>SENSe:IMS:RBW</code></td>
<td>ResolutionBW</td>
</tr>
<tr>
<td>Stimulus settings</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set DeltaF</td>
<td><code>SENSe:IMS:STIMulus:DFRequency</code></td>
<td>DeltaFrequency</td>
</tr>
<tr>
<td>Set center freq</td>
<td><code>SENSe:IMS:STIMulus:FCEnter</code></td>
<td>FrequencyCenter</td>
</tr>
<tr>
<td>Set F1 Freq</td>
<td><code>SENSe:IMS:STIMulus:F1FRequency</code></td>
<td>F1Frequency</td>
</tr>
<tr>
<td>Set F2 Freq</td>
<td><code>SENSe:IMS:STIMulus:F2FRequency</code></td>
<td>F2Frequency</td>
</tr>
<tr>
<td>Set F1 Power</td>
<td><code>SENSe:IMS:STIMulus:TPOWer:F1</code></td>
<td>TonePower</td>
</tr>
<tr>
<td>Set F2 Power</td>
<td><code>SENSe:IMS:STIMulus:TPOWer:F2</code></td>
<td>TonePower</td>
</tr>
<tr>
<td>Set F1/F2 power coupling</td>
<td><code>SENSe:IMS:TPOWer:COUPle</code></td>
<td>CoupleTonePower</td>
</tr>
<tr>
<td>Set power leveling mode</td>
<td><code>SENSe:IMS:TPOWer:LEV</code></td>
<td>LevelingMethod</td>
</tr>
<tr>
<td>Response (receiver) settings</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set receiver start frequency</td>
<td><code>SENSe:IMS:RESPonse:STARt</code></td>
<td>SpectrumStartFrequency</td>
</tr>
<tr>
<td>Set receiver stop frequency</td>
<td><code>SENSe:IMS:RESPonse:STOP</code></td>
<td>SpectrumStopFrequency</td>
</tr>
<tr>
<td>Set receiver center frequency</td>
<td><code>SENSe:IMS:RESPonse:CENT</code></td>
<td>SpectrumCenterFrequency</td>
</tr>
<tr>
<td>Set receiver frequency span</td>
<td><code>SENSe:IMS:RESPonse:SPAN</code></td>
<td>SpectrumSpanFrequency</td>
</tr>
</tbody>
</table>
See IMxSpectrum topic.

Differential IQ

Create/Change DIQ Measurement

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:CUSTom:DEFine</td>
<td>CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>CALCulate:CUSTom:MODify</td>
<td>meas.Change Parameter</td>
</tr>
</tbody>
</table>

Differential IQ Setup Dialog

| SENSE:DIQ:FREQuency:RANGe:ADD            | AddRange Method                          |
| SENSE:DIQ:FREQuency:RANGe:DELeSe         | DeleteRange Method                        |
## DIQ Range Settings Dialog

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Frequencies</strong>&lt;br&gt;SENSe:DIQ:FREQuency:RANGe:STARt</td>
<td>RangeStartFrequency</td>
</tr>
<tr>
<td></td>
<td>RangeStartFrequency</td>
</tr>
<tr>
<td></td>
<td>RangeStopFrequency</td>
</tr>
<tr>
<td>SENSE:DIQ:FREQuency:RANGe:STOP</td>
<td>RangeStopFrequency</td>
</tr>
<tr>
<td>SENSE:DIQ:FREQuency:RANGe:IFBW</td>
<td>RangeIFBW Property</td>
</tr>
<tr>
<td><strong>Coupling</strong>&lt;br&gt;SENSe:DIQ:FREQuency:RANGe:COUPle:STATe</td>
<td>RangeCoupleState Property</td>
</tr>
<tr>
<td>SENSE:DIQ:FREQuency:RANGe:COUPle:ID</td>
<td>RangeCoupleId Property</td>
</tr>
<tr>
<td>SENSE:DIQ:FREQuency:RANGe:COUPle:OFSet</td>
<td>RangeOffset Property</td>
</tr>
<tr>
<td>SENSE:DIQ:FREQuency:RANGe:COUPle:UCONvert</td>
<td>RangeOffsetUp Property</td>
</tr>
<tr>
<td>SENSE:DIQ:FREQuency:RANGe:COUPle:MULTiplier</td>
<td>RangeMultiplier Property</td>
</tr>
<tr>
<td>SENSE:DIQ:FREQuency:RANGe:COUPle:DIVisor</td>
<td>RangeDivisor Property</td>
</tr>
</tbody>
</table>

## DIQ Source Configuration Dialog

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:DIQ:PORT:STATe</td>
<td>SourceState Property</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:RANGe</td>
<td>SourceRange Property</td>
</tr>
<tr>
<td>SOURce:PHASe:EXTernal:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>SOURce:PHASe:EXTernal:PORT</td>
<td>None</td>
</tr>
<tr>
<td><strong>Power</strong>&lt;br&gt;SENSe:DIQ:PORT:POWer:SWEep</td>
<td>PowerSweepState Property</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:POWer:STARt</td>
<td>PortStartPower Property</td>
</tr>
<tr>
<td>SENSE:DIQ:PORT:POWer:STOP</td>
<td>PortStopPower Property</td>
</tr>
<tr>
<td>DIQ Edit Parameters Dialog</td>
<td>SCPI</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------</td>
</tr>
<tr>
<td>SENSE:DIQ:PARameter:DEFine</td>
<td>DefineParameter Method</td>
</tr>
<tr>
<td>SENSE:DIQ:PARameter:DELete</td>
<td>DeleteParameter Method</td>
</tr>
</tbody>
</table>
### DIQ Select X-Axis Dialog

<table>
<thead>
<tr>
<th>Parameter</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-Axis Domain</td>
<td>CALCulate:X:AXIS:DOMain</td>
<td>XAxisDomain</td>
</tr>
<tr>
<td>X-Axis</td>
<td>CALCulate:X:AXIS</td>
<td>XAxis</td>
</tr>
</tbody>
</table>

### SA Application - SA Setup tab

<table>
<thead>
<tr>
<th>Parameter</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolution BW</td>
<td>SENSE:SA:BANDwidth:[RESolution]</td>
<td>ResolutionBW</td>
</tr>
<tr>
<td>Resolution BW Mode</td>
<td>SENSE:SA:BANDwidth:[RESolution]:AUTO</td>
<td>ResolutionBWMode</td>
</tr>
<tr>
<td>Resolution BW Min</td>
<td>SENSE:SA:BANDwidth:RESolution MIN</td>
<td>ResolutionBWMin</td>
</tr>
<tr>
<td>Resolution BW Max</td>
<td>SENSE:SA:BANDwidth:RESolution MAX</td>
<td>ResolutionBWMax</td>
</tr>
<tr>
<td>Video BW</td>
<td>SENSE:SA:BANDwidth:VIDeo</td>
<td>VideoBW</td>
</tr>
<tr>
<td>Video BW Mode</td>
<td>SENSE:SA:BANDwidth:VIDeo:AUTO</td>
<td>VideoBWMode</td>
</tr>
</tbody>
</table>
**Source Setup tab**

- **SENSe:SA:SOURce:SWEep:TYP**e
- **SENSe:SA:SOURce:FREQuency:STARt**
- **SENSe:SA:SOURce:FREQuency:STOP**
- **SENSe:SA:SOURce:FREQuency:CW**
- **SENSe:SA:SOURce:POW:SWEep:POINt:COUNt**
- **SENSe:SA:SOURce:POW:SWEep:REPeat:COUNt**
- **SourceSweepType**
- **SourceStartFrequency**
- **SourceStopFrequency**
- **SourceCWFrequency**
- **SourcePowerPoint Count**
- **SourcePowerRepeatCount**
<table>
<thead>
<tr>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:COHerence:MULTitone:Valid</td>
<td>MultiToneSettings Valid</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:VECTOR:AVERAGE[:STATE]</td>
<td>VectorAverageEnable</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:VECTOR:AVERAGE:VALUE</td>
<td>VectorAverageValue</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:VECTOR:AVERAGE:VALUEMax</td>
<td>VectorAverageValueMax</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PHASE[:STATE]</td>
<td>PhaseProcessState</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PHASE:DISPLAY:LEVEL</td>
<td>PhaseDisplayMinLevel</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSE:SEARCH:COUNT</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSE:SEARCH:EXECUTE</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSE:SEARCH:INITIALIZE</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSE:SEARCH:ITEM:SELECT</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:COHerence:PULSE:SEARCH:PERIOD:tolerance</td>
<td>None</td>
</tr>
</tbody>
</table>
SA Setup: Channel 1

**IF Gain**
- A: Auto
- B: Auto
- C: Auto
- D: Auto

**IF Bandwidth**
- ADC Filter: Auto
- DFT Bandwidth: Auto
  - Narrow: 1.000000 MHz to 10.000000 MHz
  - Wide: 1.000000 MHz to 34.000000 MHz

Trig. & Pulse Setup tab

**SENSe:SA:TRIGer:LEVel[:STATe]**
- TriggerADCLevelState
**SENSe:SA:TRIGer:LEVel:VA**
- TriggerADCLevelValue
**SENSe:SA:TRIGer:PERCoun**
- TriggerPeriodicCounterState

Bandwidth
- NarrowMin
- NarrowMax
- WideMin
- WideMax
### Processing Setup tab

<table>
<thead>
<tr>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:DATA:KEEP:FREQuencies:COUNt?</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:DATA:KEEP:RECeivers:COUNt?</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:DATA:KEEP:RECeivers[:CURRent]</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:DATA:KEEP:RECeivers:LIST</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:DATA:KEEP[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:DATA:KEEP:VECTor:GET?</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:DFT:TYPE</td>
<td>Type</td>
</tr>
<tr>
<td>Parameter</td>
<td>Value</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>SENSE:SA:TRACe:IMAGe[:STATE]</td>
<td>EnableImageRejectionTraces</td>
</tr>
<tr>
<td>SENSE:SA:ADC:ACQTime?</td>
<td>AcquisitionTime</td>
</tr>
<tr>
<td>SENSE:SA:LO:COUNt?</td>
<td>LOCount</td>
</tr>
<tr>
<td>SENSE:SA:SPAN:BINS:COUNt?</td>
<td>SpanBinsCount</td>
</tr>
<tr>
<td>SENSE:SA:DFT:RESolution?</td>
<td>Resolution</td>
</tr>
<tr>
<td>SENSE:SA:DFT:RECORD:SIZE?</td>
<td>RecordSize</td>
</tr>
<tr>
<td>SENSE:SA:ADC:OVERload:COUNT?</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:ADC:OVERload:LIST?</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:ADC:OVERload:PE RCent</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:ADC:RANGE:PERCENT[:MAXimum]?</td>
<td>None</td>
</tr>
<tr>
<td>SENSE:SA:ADC:SAMPLE:RATE</td>
<td>ADCSampleRate</td>
</tr>
<tr>
<td>SENSE:SA:ADC:SAMPLE:RATE:AUTO</td>
<td>EnableADCSampleRateAuto</td>
</tr>
<tr>
<td>SENSE:SA:ADC:SAMPLE:DECIMATION:FIR</td>
<td>ADCEnableFIRFor25Mhz</td>
</tr>
<tr>
<td>Original Text</td>
<td>Natural Text</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
</tr>
<tr>
<td>SENSe:SA:ADC:DITHer:[STATe]</td>
<td>EnableADCdither</td>
</tr>
<tr>
<td>SENSe:SA:ADC:RECord:SIZE:MIN?</td>
<td>ADCRecordSizeMin</td>
</tr>
<tr>
<td>SENSe:SA:ADC:STACKing:VALUE</td>
<td>ADCStacking</td>
</tr>
<tr>
<td>SENSe:SA:ADC:STACKing:STACKing</td>
<td>ADCStackingMax</td>
</tr>
<tr>
<td>SENSe:SA:ADC:STACKing:STATe</td>
<td>ADCStackingState</td>
</tr>
<tr>
<td>SENSe:SA:ADC:MREC:SIZE</td>
<td>ADCMultiRecSize</td>
</tr>
<tr>
<td>SENSe:SA:ADC:MREC:PERiod</td>
<td>ADCMultiRecPeriod</td>
</tr>
<tr>
<td>SENSe:SA:ADC:MREC[:STATe]</td>
<td>ADCMultiRecState</td>
</tr>
<tr>
<td>SENSe:SA:LO:RANDOM[:STATe]</td>
<td>EnableRandomizedLO</td>
</tr>
<tr>
<td>SENSe:SA:LO:BASEband[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>SENSe:SA:LO:FORCe:OFFSET:DIVider</td>
<td>None</td>
</tr>
<tr>
<td>SENSe:SA:LO:FORCe:OFFSET:MULTiplier</td>
<td>None</td>
</tr>
<tr>
<td>SENSe:SA:LO:FORCe:OFFSET:SOURce</td>
<td>None</td>
</tr>
<tr>
<td>SENSe:SA:LO:FORCe:FREQuency</td>
<td>None</td>
</tr>
<tr>
<td>SENSe:SA:LO:FORCe[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>SENSe:SA:LO:FREQ:FORCe (obsolete)</td>
<td>None</td>
</tr>
</tbody>
</table>

### Diagram

![Diagram of SA Setup: Channel 1](image-url)
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSE:SA:LO:FREQ:VALUE (obsolete)</td>
<td>Force LOTo Frequency</td>
</tr>
<tr>
<td>Data tab</td>
<td></td>
</tr>
<tr>
<td>SENSE:SA:DATA:TYPE</td>
<td>DataFormat</td>
</tr>
<tr>
<td>SENSE:SA:DATA:START?</td>
<td>DataFirstRFBin</td>
</tr>
<tr>
<td>SENSE:SA:DATA:RECEIVERS:LIST</td>
<td>ExportReceiverSet List</td>
</tr>
<tr>
<td>SENSE:SA:DATA:BINs:COUNT?</td>
<td>DataBinCount</td>
</tr>
<tr>
<td>SENSE:SA:DATA:SIZE?</td>
<td>DataByteSize</td>
</tr>
<tr>
<td>SENSE:SA:DATA:SIZE:BIN?</td>
<td>DataBytesPerBin</td>
</tr>
<tr>
<td>SENSE:SA:DATA:SIZE:LOW?</td>
<td>DataByteSizeLOW</td>
</tr>
<tr>
<td>SENSE:SA:DATA:SIZE:HIGH?</td>
<td>DataByteSizeHIGH</td>
</tr>
<tr>
<td>SENSE:SA:DATA:PREFIX</td>
<td>FilePrefix</td>
</tr>
<tr>
<td>SENSE:SA:DATA:FIFO[:STATE]</td>
<td>FIFOEnabled</td>
</tr>
<tr>
<td>SENSE:SA:DATA:SHARED:NAME</td>
<td>MemShareName</td>
</tr>
</tbody>
</table>
## SA Marker Settings

Sets and reads the bandwidth of the band density marker.

Returns the band density level in dBm/Hz from the band density marker.

Marker to SA

Read Band Power

Read/Set Band Power Span

Read/Set Band Power State

Sets and reads the state of the band density noise marker.

Sets and reads the bandwidth of the band power density marker.

Sets and reads the state of the band power density marker.

Sets and reads the bandwidth of the band tone density marker.

Sets and reads the state of the band tone density marker.

Sets and reads the spacing of the band tone density marker.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:MEASure:SA:MARKer:BDENsity:BW</td>
<td>Sets and reads the bandwidth of the band density</td>
<td>BandDensityBW</td>
</tr>
<tr>
<td>CALCulate:MEASure:SA:MARKer:BDENsity:DATA?</td>
<td>Returns the band density level in dBm/Hz from the</td>
<td>BandDensityValue</td>
</tr>
<tr>
<td>CALCulate:MEASure:MARKer:SET SA</td>
<td>Marker to SA</td>
<td>toSA</td>
</tr>
<tr>
<td>CALCulate:MEASure:SA:MARKer:BPOwer:SPAN</td>
<td>Read/Set Band Power Span</td>
<td>BandpowerSpan</td>
</tr>
<tr>
<td>CALCulate:MEASure:SA:MARKer:BPOwer:STATe</td>
<td>Read/Set Band Power State</td>
<td>BandpowerState</td>
</tr>
<tr>
<td>CALCulate:SA:MARKer:BDENsity:NOISe[:STATe]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALCulate:SA:MARKer:BDENsity:POWer:BW</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALCulate:MEASure:SA:MARKer:BDENsity:POWer[:STATe]</td>
<td>Sets and reads the state of the band power density marker.</td>
<td>BandDensityPowerState</td>
</tr>
<tr>
<td>CALCulate:SA:MARKer:BDENsity:POWer[:STATe]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALCulate:SA:MARKer:BDENsity:TONE:BW</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALCulate:SA:MARKer:BDENsity:TONE[:STATe]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALCulate:SA:MARKer:BDENsity:TONE:TSPacing</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Sets and reads the frequency span used by Power Density to normalize the power.

Read occupied bandwidth center frequency

Set and read occupied bandwidth percentage of span

Read the occupied bandwidth power.

Read occupied bandwidth span

Set occupied bandwidth state

CALCulate:MEASure:SA:MARKer:BDENsity:EQSPan

CALCulate:SA:MARKer:BDENsity:EQSPan

CALCulate:MEASure:SA:MARKer:OCCBand:CENTer?

CALCulate:MEASure:SA:MARKer:OCCBand:PERCent

CALCulate:MEASure:SA:MARKer:OCCBand:POWer?

CALCulate:MEASure:SA:MARKer:OCCBand:SPAN?

CALCulate:MEASure:SA:MARKer:OCCBand[:STATe]

BandDensityEQSPan

OccupiedBandCenter

OccupiedBandPercent

OccupiedBandPowerdBm

OccupiedBandSpan

OccupiedBandState

---

**IQMod - Modulation Settings for NPR Measurements**

<table>
<thead>
<tr>
<th>Modulation source</th>
<th>MyI2G_V</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Modulation</td>
<td>✔️</td>
</tr>
<tr>
<td>Modulation Filename</td>
<td>C:\Users\Public\Documents\Network Analyzer\t2.mdx</td>
</tr>
<tr>
<td>Autoset Frequencies and Coherence for current modulation</td>
<td>✔️</td>
</tr>
<tr>
<td>Autoset NPR Markers</td>
<td>✔️</td>
</tr>
<tr>
<td>Autoset ACPR Markers</td>
<td>✔️</td>
</tr>
<tr>
<td>Enable Modulation Correction</td>
<td>✔️</td>
</tr>
<tr>
<td>Source Amplification (= Power Offset)</td>
<td>36.608 dB</td>
</tr>
<tr>
<td></td>
<td><strong>OK</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Command</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURce:MODulation[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>SOURce:MODulation:LOAD</td>
<td>None</td>
</tr>
<tr>
<td>Create/Edit Mod File...</td>
<td>None</td>
</tr>
<tr>
<td>SOURce:MODulation:SA[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>SOURce:MODulation:AUTO:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>SOURce:MODulation:AUTO:NPR[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>SOURce:MODulation:AUTO:GBANd</td>
<td>None</td>
</tr>
<tr>
<td>SOURce:MODulation:AUTO:ACPRL[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>SOURce:MODulation:AUTO:ACPR:GBANd</td>
<td>None</td>
</tr>
</tbody>
</table>
### Miscellaneous SA Commands

<table>
<thead>
<tr>
<th>Operation</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zooms in on signal within a SA sweep.</td>
<td>SENSE:SA:FREQuency:TUNE:IMMediate</td>
<td>FrequencyAutoTune</td>
</tr>
</tbody>
</table>

### Mixer Setup commands

Used for FCA, Swept IMDx, IMx Spectrum, NFx, and GCx applications.

See the **Mixer / Converter Setup topic**

Create or Change a Custom (Application) Measurement

<table>
<thead>
<tr>
<th>Operation</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a custom measurement</td>
<td>CALCulate:CUStom:DEFine</td>
<td>App.CreateCustomMeasurementEx</td>
</tr>
<tr>
<td>Change a custom measurement</td>
<td>CALCulate:CUStom:MODify</td>
<td>meas.Change Parameter</td>
</tr>
<tr>
<td>Return handle to a converter object</td>
<td>None</td>
<td>Converter</td>
</tr>
</tbody>
</table>

Mixer Bottom Buttons and X-Axis display

<table>
<thead>
<tr>
<th>Operation</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load a mixer setup</td>
<td>SENSE:MIXer:LOAD</td>
<td>LoadFile</td>
</tr>
<tr>
<td>Save a mixer setup</td>
<td>SENSE:MIXer:SAVE</td>
<td>SaveFile</td>
</tr>
<tr>
<td>Apply mixer settings</td>
<td>SENSE:MIXer:APPLy</td>
<td>Apply</td>
</tr>
<tr>
<td>Discard Changes</td>
<td>SENSE:MIXer:DISCard</td>
<td>DiscardChanges</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>X-axis display</td>
<td>CALCulate:MEASure:MIXer:XAXis</td>
<td>ActiveXAxisRange</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mixer Frequency Tab</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Calculate Input and Output frequencies</strong></td>
</tr>
<tr>
<td><strong>Input to swept or fixed</strong></td>
</tr>
<tr>
<td><strong>Input start frequency</strong></td>
</tr>
<tr>
<td><strong>Input stop frequency</strong></td>
</tr>
<tr>
<td><strong>Input power level</strong></td>
</tr>
<tr>
<td><strong>Input fixed frequency</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set LO</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LO freq fixed or swept</strong></td>
</tr>
<tr>
<td><strong>LO fixed frequency</strong></td>
</tr>
<tr>
<td><strong>LO start frequency</strong></td>
</tr>
<tr>
<td><strong>LO stop frequency</strong></td>
</tr>
<tr>
<td><strong>Input Greater / Less that LO</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set IF</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sideband (high or low)</strong></td>
</tr>
<tr>
<td><strong>IF start frequency</strong></td>
</tr>
<tr>
<td><strong>IF stop frequency</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sideband (high or low)</strong></td>
</tr>
<tr>
<td><strong>Output start frequency</strong></td>
</tr>
<tr>
<td><strong>Output stop frequency</strong></td>
</tr>
</tbody>
</table>

8400
<table>
<thead>
<tr>
<th>Output to swept or fixed</th>
<th>Sense:Mixer:OUTPut:FREQuency:MODE</th>
<th>OutputRangeMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output fixed frequency</td>
<td>Sense:Mixer:OUTPut:FREQuency:FIXed</td>
<td>OutputFixedFrequency</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mixer (LO)Power tab</th>
<th>Sense:Mixer:LO:POW</th>
<th>LOPower</th>
</tr>
</thead>
<tbody>
<tr>
<td>LO power</td>
<td>Sense:Mixer:LO:POW:STARt</td>
<td>LOStartPower</td>
</tr>
<tr>
<td>LO power start</td>
<td>Sense:Mixer:LO:POW:STOP</td>
<td>LOStopPower</td>
</tr>
<tr>
<td>Source Attenuator</td>
<td>Source:POWER:ATTenuation</td>
<td>chan.Attenuator</td>
</tr>
<tr>
<td>Receiver Attenuator</td>
<td>Sense:POWER:ATTenuator</td>
<td>chan.ReceiverAttenuator</td>
</tr>
<tr>
<td>Leveling Mode</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mixer Setup tab</th>
<th>Sense:Mixer:STAGE</th>
<th>LOStage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of LOs (1 or 2)</td>
<td>Sense:Mixer:LO:NAME</td>
<td>LOName</td>
</tr>
<tr>
<td>Recall a previously-configured external source.</td>
<td>Sense:Mixer:ROLE:DEVi ce</td>
<td>AssignSourceToRole</td>
</tr>
<tr>
<td>Assign a source to mixer input or LO.</td>
<td>Sense:Mixer:ROLE:CATal og?</td>
<td>GetSourceByRole</td>
</tr>
<tr>
<td>Read all assigned roles</td>
<td>Sense:Mixer:ROLE:DEVi ce</td>
<td>GetSourceRoles</td>
</tr>
<tr>
<td>Read the source assigned to a role.</td>
<td>Sense:Mixer:ROLE:DEVi ce</td>
<td>GetSourceRoles</td>
</tr>
<tr>
<td>LO Denominator Frac.Mult</td>
<td>Sense:Mixer:LO:FREQuency:DENominator</td>
<td>LODenominator</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMD and IMS Limited Port Mapping</th>
<th></th>
<th></th>
</tr>
</thead>
</table>

8401
<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>
| Set port map           | SENSe:IMD:PMAP
SENSe:IMS:PMAP                                                          | SetPortMap                       |
| Read Input             | SENSe:IMD:PMAP:INPut?
SENSe:IMS:PMAP:INPut?                                                     | DeviceInputPort                  |
| Read Output            | SENSe:IMD:PMAP:OUTPut?
SENSe:IMS:PMAP:OUTPut?                                                     | DeviceOutputPort                 |

iTMSA - all other settings (including calibration) use standard commands

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Stimulus Mode</td>
<td>CALCulate: FSIMulator:BALun:STIMulus:MODE</td>
<td>Mode</td>
</tr>
</tbody>
</table>
| Set Phase Offset       | CALCulate: FSIMulator:BALun:BPORT:OFFSET:PHASe                           | BalPort1PhaseOffset
BalPort2PhaseOffset    |
| Set Phase Offset as fixture | CALCulate: FSIMulator:BALun:FIXTure:OFFSET:PHASe                    | PhaseAsFixture                   |
| Set Power Offset       | CALCulate: FSIMulator:BALun:BPORT:OFFSET:POWer                            | BalPort1PowerOffset
BalPort2PowerOffset    |
| Set Power Offset as fixture | CALCulate: FSIMulator:BALun:FIXTure:OFFSET:POWer                  | PowerAsFixture                   |
| Sets and reads the True Mode state for a specified balanced port. | CALCulate:FSIMulator:BALun:BPORT:STIMulus:TRUE:STATE                   | BalancedPortTrueState            |
| Set Source power for balanced ports | SOURce:POWer                                                                | TestPortPower                    |
| Returns the number of source ports. | N/A                                                                         | chan.SourcePortCount             |
| Returns the string names of source ports. | N/A                                                                        | chan.SourcePortNames             |
| Returns the port number for the specified string port name. | N/A                                                                        | chan.GetPortNumber               |

iTMSA Phase Sweep

<table>
<thead>
<tr>
<th>Action</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Phase sweep</td>
<td>CALCulate:FSIMulator:BALun:PHASE:SWEep:STATe</td>
<td>PhaseSwpState</td>
</tr>
<tr>
<td>Operation</td>
<td>Command Description</td>
<td>Example Command</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Start Phase port 1</td>
<td>CALCulate:FSIMulator:Balun:BPort:Sweep:Phase:STARt</td>
<td>BalPort1StartPhase</td>
</tr>
<tr>
<td>Start Phase port 2</td>
<td>CALCulate:FSIMulator:Balun:BPort:Sweep:Phase:STARt</td>
<td>BalPort2StartPhase</td>
</tr>
<tr>
<td>Stop Phase port 1</td>
<td>CALCulate:FSIMulator:Balun:BPort:Sweep:Phase:STOP</td>
<td>BalPort1StopPhase</td>
</tr>
<tr>
<td>Stop Phase port 2</td>
<td>CALCulate:FSIMulator:Balun:BPort:Sweep:Phase:STOP</td>
<td>BalPort2StopPhase</td>
</tr>
<tr>
<td>Enable as fixture offset</td>
<td>CALCulate:FSIMulator:Balun:FIXture:Phase</td>
<td>PhaseSwpAsFixture</td>
</tr>
</tbody>
</table>

**Enhanced Time Domain Analysis (TDR)**

Setup and execution of TDR measurement Commands

<table>
<thead>
<tr>
<th>Setup/Execution</th>
<th>Command</th>
<th>Example Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets the type of parameter and format allocation for each trace.</td>
<td>CALCulate:TDR:ALLOCate</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the DUT topology.</td>
<td>CALCulate:TDR:DEVice</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the emphasis post1 level.</td>
<td>CALCulate:TDR:EMPHasis:CURSor:POST1</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the emphasis post2 level.</td>
<td>CALCulate:TDR:EMPHasis:CURSor:POST2</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the emphasis pre1 level.</td>
<td>CALCulate:TDR:EMPHasis:CURSor:PRE1</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Turns the emphasis function state ON or OFF.</td>
<td>CALCulate:TDR:EMPHasis:STATE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the equalization CTLE DC gain parameter.</td>
<td>CALCulate:TDR:EQUalization:CTLE:DC</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the equalization CTLE Pole1 parameter.</td>
<td>CALCulate:TDR:EQUalization:CTLE:POLE1</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the equalization CTLE Pole2 parameter.</td>
<td>CALCulate:TDR:EQUalization:CTLE:POLE2</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the equalization CTLE zero parameter.</td>
<td>CALCulate:TDR:EQUalization:CTLE:ZERO1</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the filename of the equalization equation user file.</td>
<td>CALCulate:TDR:EQUalization:FILENAME</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Turns the equalization function state ON or OFF.</td>
<td>CALCulate:TDR:EQUalization:STATE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the equalization type.</td>
<td>CALCulate:TDR:EQUalization:TYPE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>Executes the calculation for the simulated eye diagram for the active trace.</td>
<td><code>CALCulate:TDR:EYE:EXECute</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the bits' power of 2 for a PRBS pattern.</td>
<td><code>CALCulate:TDR:EYE:INPut:BPATtern:LENGth</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the bit pattern type for the simulated eye function.</td>
<td><code>CALCulate:TDR:EYE:INPut:BPATtern:TYPE</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the bit rate in bits/sec for the simulated eye function.</td>
<td><code>CALCulate:TDR:EYE:INPut:DRATe</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the display limit value.</td>
<td><code>CALCulate:TDR:EYE:INPut:JITTer:DLIMit</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the periodic jitter frequency.</td>
<td><code>CALCulate:TDR:EYE:INPut:JITTer:PERiodic:FREQuency</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the periodic jitter magnitude in rms.</td>
<td><code>CALCulate:TDR:EYE:INPut:JITTer:PERiodic:MAGNitude</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the random jitter magnitude in rms</td>
<td><code>CALCulate:TDR:EYE:INPut:JITTer:RANDom:MAGNitude</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Turns the jitter function state with simulated eye ON or OFF.</td>
<td><code>CALCulate:TDR:EYE:INPut:JITTer:STATe</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the jitter function type for the simulated eye function.</td>
<td><code>CALCulate:TDR:EYE:INPut:JITTer:TYPE</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the voltage level for bit &quot;1&quot; for the simulated eye function.</td>
<td><code>CALCulate:TDR:EYE:INPut:OLEVel</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the rise time value for the simulated eye function.</td>
<td><code>CALCulate:TDR:EYE:INPut:RTIMe:DATA</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the rise time threshold for the simulated eye.</td>
<td><code>CALCulate:TDR:EYE:INPut:RTIMe:THReshold</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the voltage level for bit &quot;0&quot; for the simulated eye function.</td>
<td><code>CALCulate:TDR:EYE:INPut:ZLEVel</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Returns the mask test result.</td>
<td><code>CALCulate:TDR:EYE:MASK:FAIL?</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Turns the mask test function state with simulated eye ON or OFF.</td>
<td><code>CALCulate:TDR:EYE:MASK:STATe</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Returns the results of the eye measurement.</td>
<td><code>CALCulate:TDR:EYE:RESults:DATA?</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Turns the overlay ON or OFF.</td>
<td><code>CALCulate:TDR:EYE:RESults:DISplay:STATe</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Function Description</td>
<td>Command</td>
<td>Option</td>
<td></td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>Sets the rise time threshold level for the results of eye measurement.</td>
<td>CALCulate:TDR:EYE:RESults:THReshold</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Turns the Eye/Mask window ON or OFF.</td>
<td>CALCulate:TDR:EYE:STATE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets active marker number.</td>
<td>CALCulate:TDR:MEASure[1-16]:ACTive:MARKer</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Returns the delta time result value.</td>
<td>CALCulate:TDR:MEASure[1-16]:DTIMe:DATA?</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets delta time reference position.</td>
<td>CALCulate:TDR:MEASure[1-16]:DTIMe:POsition</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Turns the delta time marker in the marker search ON or OFF.</td>
<td>CALCulate:TDR:MEASure[1-16]:DTIMe:STATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the target trace number for the delta time function.</td>
<td>CALCulate:TDR:MEASure[1-16]:DTIMe:TARGet</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the trace format.</td>
<td>CALCulate:TDR:MEASure[1-16]:FORMat</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the measurement parameter.</td>
<td>CALCulate:TDR:MEASure[1-16]:PARameter</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets state for the peeling function.</td>
<td>CALCulate:TDR:MEASure[1-16]:PEELing:STATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets state for the smoothing function.</td>
<td>CALCulate:TDR:MEASure[1-16]:SMOothing:STATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the impulse width value for the transform function.</td>
<td>CALCulate:TDR:MEASure[1-16]:TIME:IMPulse:WIDTh</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets rise time value for the transform function.</td>
<td>CALCulate:TDR:MEASure[1-16]:TIME:STEP:RTIMe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the rise time threshold level for the results of eye measurement.</td>
<td>CALCulate:TDR:MEASure[1-16]:TIME:STEP:RTIMe:THReshold</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the stimulus type for the transform function.</td>
<td>CALCulate:TDR:MEASure[1-16]:TIME:TYPE</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Returns the rise time result value for marker search.</td>
<td>CALCulate:TDR:MEASure[1-16]:TTIMe:DATA?</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Displays the rise time marker.</td>
<td>CALCulate:TDR:MEASure[1-16]:TTIMe:STATe</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the rise time threshold for the rise time in the marker search function.</td>
<td>CALCulate:TDR:MEASure[1-16]:TTIMe:THReshold</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Display TDR Commands</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>Turns the continuous auto-scale mode for the eye y-axis ON or OFF.</td>
<td><code>DISPLAY:EYE:Y:SCALE:AUTO:STATE</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the value of the y-axis scale per division for eye diagram.</td>
<td><code>DISPLAY:EYE:Y:SCALE:PDIVision</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the value of the eye diagram y-axis reference line.</td>
<td><code>DISPLAY:EYE:Y:SCALE:RLEVEL</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the value of the eye diagram y-axis reference position.</td>
<td><code>DISPLAY:EYE:Y:SCALE:RPOSITION</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Changes the background color of the screen.</td>
<td><code>DISPLAY:IMAGE</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the display to off, data type, memory type, or data and memory type.</td>
<td><code>DISPLAY:MEASURE[1-16]:DMEMORY:TYPE</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Executes x-axis auto scaling.</td>
<td><code>DISPLAY:MEASURE[1-16]:X:SCALE:AUTO</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the value of the x-axis scale per division.</td>
<td><code>DISPLAY:MEASURE[1-16]:X:SCALE:PDIVISION</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the value of the x-axis reference line.</td>
<td><code>DISPLAY:MEASURE[1-16]:X:SCALE:RLEVEL</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Executes y-axis auto scaling.</td>
<td><code>DISPLAY:SCALE:AUTO</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets the x-axis reference position for the time domain measurement.</td>
<td><code>DISPLAY:SCALE:RPOSITION</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Selects the view point for waveform analysis either before or after the DUT.</td>
<td><code>DISPLAY:VIEW</code></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Sets or gets the minimize state.</td>
<td><code>DISPLAY:MINIMIZE:STATE</code></td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Memory TDR Commands**

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loads the specified user bit pattern file.</td>
<td><code>MEMORY:TDRLoad[:EYE]:BPATtern</code></td>
<td>None</td>
</tr>
<tr>
<td>Loads eye-mask file.</td>
<td><code>MEMORY:TDRLoad[:EYE][:MASK]</code></td>
<td>None</td>
</tr>
<tr>
<td>Stores the user bit pattern file.</td>
<td><code>MEMORY:STORe[:EYE]:BPATtern</code></td>
<td>None</td>
</tr>
<tr>
<td>Stores the eye-mask file.</td>
<td><code>MEMORY:STORe[:EYE][:MASK]</code></td>
<td>None</td>
</tr>
</tbody>
</table>

**Correction TDR Commands**

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Executes load measurement.</td>
<td><code>SENSe:CORRection:TDRLeCollection:DLComp:LOAD</code></td>
<td>None</td>
</tr>
<tr>
<td>Operation Description</td>
<td>Command</td>
<td>Parameters</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Executes open measurement.</td>
<td>SENSE:CORRection:TDR:COLLection:DLComp:OPEN</td>
<td>None</td>
</tr>
<tr>
<td>Saves the result of the Loss Compensation sequence.</td>
<td>SENSE:CORRection:TDR:COLLection:DLComp:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Executes a thru measurement.</td>
<td>SENSE:CORRection:TDR:COLLection:DLComp:THRU</td>
<td>None</td>
</tr>
<tr>
<td>Executes fixture compensation after ECAL.</td>
<td>SENSE:CORRection:TDR:COLLection:ECAL:FCOMP:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>Executes full calibration using the ECaL module.</td>
<td>SENSE:CORRection:TDR:COLLection:ECAL:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>Sets the dielectric constant value.</td>
<td>SENSE:CORRection:TDR:DCONstant</td>
<td>None</td>
</tr>
<tr>
<td>Executes deskew (auto port extension).</td>
<td>SENSE:CORRection:TDR:EXTension:AUTO:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>Sets the standard for auto port extension.</td>
<td>SENSE:CORRection:TDR:EXTension:AUTO:STANDard</td>
<td>None</td>
</tr>
<tr>
<td>Sets the reference impedance value.</td>
<td>SENSE:CORRection:TDR:RIMPedance</td>
<td>None</td>
</tr>
<tr>
<td><strong>Bandwidth, DUT information, avoid spurious function, and sweep TDR Commands</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets the IF bandwidth value.</td>
<td>SENSE:TDR:BWIDth[:RESolution]</td>
<td>None</td>
</tr>
<tr>
<td>Executes auto DUT length setting.</td>
<td>SENSE:TDR:DLENgth:AUTO:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>Sets the DUT length value.</td>
<td>SENSE:TDR:DLENgth:DATA</td>
<td>None</td>
</tr>
<tr>
<td>Executes avoid spurious.</td>
<td>SENSE:TDR:SPURious:AVOid:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>Queries the avoid spurious state.</td>
<td>SENSE:TDR:SPURious:AVOid:STATe?</td>
<td>None</td>
</tr>
<tr>
<td>Sets the value of input bit rate for avoid spurious.</td>
<td>SENSE:TDR:SPURious:INPut:DRATe</td>
<td>None</td>
</tr>
<tr>
<td>Queries the Hot TDR mode status.</td>
<td>SENSE:TDR:SPURious:STATe?</td>
<td>None</td>
</tr>
<tr>
<td>Sets the TDR averaging trigger state on/off.</td>
<td>SENSE:TDR:SWEep:AVERage</td>
<td>None</td>
</tr>
<tr>
<td>Sets trigger mode.</td>
<td>SENSE:TDR:SWEep:MODE</td>
<td>None</td>
</tr>
<tr>
<td>Executes single trigger.</td>
<td>SENSE:TDR:SWEep:SINGle</td>
<td>None</td>
</tr>
<tr>
<td>Enables/disables the Reduce IF BW at Low Frequencies feature in segments with IFBW arbitrary.</td>
<td>SENSE:TDR:BWIDth:TRACk:FORCe</td>
<td>None</td>
</tr>
<tr>
<td>Source Power Level TDR Command</td>
<td>Sets the source power level.</td>
<td>SOURce:TDR:POWer[:LEVel][:IMMediate][:AMPplitude]</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>Preset TDR Command</td>
<td>Executes a TDR preset.</td>
<td>SYSTem:TDR:PRESet</td>
</tr>
<tr>
<td>Active (Hot) Parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets whether or not interpolation is on for display.</td>
<td>SENSE:ACTive:DISP:INTERpolate[:STATE]</td>
<td>DisplayInterpolationState</td>
</tr>
<tr>
<td>Set and read a fixed input power level.</td>
<td>SENSE:ACTive:DISPLAY:TRACE:IPWer</td>
<td>DisplayInputPower</td>
</tr>
<tr>
<td>Set and read the number of phase points.</td>
<td>SENSE:ACTive:SWEep:PHASe:POINT</td>
<td>PhaseSweepPoints</td>
</tr>
<tr>
<td>Set and read the start power level for a 3D sweep.</td>
<td>SENSE:ACTive:SWEep:POWer:STARt</td>
<td>StartPowerIn3DSweep</td>
</tr>
<tr>
<td>Set and read the number of power steps for a 3D sweep.</td>
<td>SENSE:ACTive:SWEep:POWer:STEP</td>
<td>PowerStepsIn3DSweep</td>
</tr>
<tr>
<td>Set and read the stop power level for a 3D sweep.</td>
<td>SENSE:ACTive:SWEep:POWer:STOP</td>
<td>StopPowerIn3DSweep</td>
</tr>
<tr>
<td>Set and read the sweep type.</td>
<td>SENSE:ACTive:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Set and read the tuning tone mode.</td>
<td>SENSE:ACTive:TTONe:MODE</td>
<td>ExtractionToneMode</td>
</tr>
<tr>
<td>Set and read the absolute tone power level.</td>
<td>SENSE:ACTive:TTONe:ABSolute</td>
<td>AbsoluteExtractionToneLevel</td>
</tr>
<tr>
<td>Set and read the tone power relative to the input power (dBc).</td>
<td>SENSE:ACTive:TTONe:RELative</td>
<td>RelativeExtractionToneLevel</td>
</tr>
<tr>
<td>Set and read the X-axis domain type.</td>
<td>CALC:X:AXIS:DOMain</td>
<td>DisplayDomain</td>
</tr>
<tr>
<td>Modulation Distortion Commands</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets and returns the noise bandwidth.</td>
<td>SENSe:SA:BANDwidth:NOISe</td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable setting the noise bandwidth to its maximum possible value.</td>
<td>SENSe:SA:BANDwidth:NOISe:AUTO</td>
<td>None</td>
</tr>
<tr>
<td>Selects between a narrow and wide IF filter anti-aliasing path.</td>
<td>SENSe:DISTortion:ADC:FILTer:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>This command launches the Receiver IF Cal.</td>
<td>SENSe&lt;cnum&gt;:DISTortion:CORRection:COLLect:IF:ACQuire [SYNChronous</td>
<td>ASYNchronous]</td>
</tr>
<tr>
<td>Identifies the normalization used for the EVM measurements.</td>
<td>SENSe:DISTortion:EVM:NORMalize</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the frequency span window used for modeling the DUT's gain and distortion.</td>
<td>SENSe:DISTortion:MEASure:CORRelation:APERture</td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable automatic calculation of the frequency span window used for modeling the</td>
<td>SENSe:DISTortion:MEASure:CORRelation:APERture:AUTO[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>DUT's gain and distortion.</td>
<td>SENSE:DISTortion:MEASure:BAND:ACP:LOWer:IBW</td>
<td>None</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Sets and returns the integration bandwidth (IBW) of the lower ACP measurement.</td>
<td>SENSE:DISTortion:MEASure:BAND:ACP:LOWer:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the lower ACP measurement offset.</td>
<td>SENSE:DISTortion:MEASure:BAND:ACP:UPPer:IBW</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the integration bandwidth (IBW) of the upper ACP measurement.</td>
<td>SENSE:DISTortion:MEASure:BAND:ACP:UPPer:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the upper ACP measurement offset.</td>
<td>SENSE:DISTortion:MEASure:BAND:ADD</td>
<td>None</td>
</tr>
<tr>
<td>Adds a new distortion measurement band.</td>
<td>SENSE:DISTortion:MEASure:BAND:AUTofill</td>
<td>None</td>
</tr>
<tr>
<td>Automatically fills in the measurement settings for all bands from the currently active modulation file loaded in the source.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Default</td>
</tr>
<tr>
<td>-------------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>Sets and returns the carrier (Signal) integration bandwidth (IBW) for the distortion measurement.</td>
<td>SENSE:DISTortion:MEASure:BAND:CARRier:IBW</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the carrier offset (Signal).</td>
<td>SENSE:DISTortion:MEASure:BAND:CARRier:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>Queries the total number of distortion measurement bands.</td>
<td>SENSE:DISTortion:MEASure:BAND:COUNt?</td>
<td>None</td>
</tr>
<tr>
<td>Deletes the specified distortion measurement band.</td>
<td>SENSE:DISTortion:MEASure:BAND:DELeTe</td>
<td>None</td>
</tr>
<tr>
<td>Deletes all existing bands and adds a single default band back into the Measurement Band Table.</td>
<td>SENSE:DISTortion:MEASure:BAND:INITialize</td>
<td>None</td>
</tr>
<tr>
<td>Assigns a name to the specified distortion measurement band.</td>
<td>SENSE:DISTortion:MEASure:BAND:NAME</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the notch integration bandwidth</td>
<td>SENSE:DISTortion:MEASure:BAND:NOTCh:IBW</td>
<td>None</td>
</tr>
<tr>
<td>(IBW) for an NPR Notch modulation measurement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Sets and returns the notch offset for an NPR Notch modulation measurement.</td>
<td><code>SENSe:DISTortion:MEASure:BAND:NOTCh:OFFSet</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td>Set and read the modulation distortion type.</td>
<td><code>SENSe:DISTortion:MEASure:BAND:TYPE</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td>Sets the measurement filter to either None (default) or RRC (root-raised-cosine filter)</td>
<td><code>SENSe:DISTortion:MEASure:FILTer</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td>Sets and returns the Alpha factor of the measurement filter.</td>
<td><code>SENSe:DISTortion:MEASure:FILTer:ALPHa</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td>Sets and returns the Symbol Rate of the filter.</td>
<td><code>SENSe:DISTortion:MEASure:FILTer:SRATe</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td>Enables or disables using the symbol rate from the file.</td>
<td><code>SENSe:DISTortion:MEASure:FILTer:SRATe:AUTO[:STATe]</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td>Sets and returns the distortion modulation source name.</td>
<td><code>SENSe:DISTortion:MODulate:SOURce</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Sets and returns the VNA port number connected to the DUT input.</td>
<td>SENSE:DISTortion:PATH:DUT:INPut</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the nominal DUT gain.</td>
<td>SENSE:DISTortion:PATH:DUT:NOMinal:GAIN</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the nominal DUT noise figure.</td>
<td>SENSE:DISTortion:PATH:DUT:NOMinal:NF</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the VNA port number connected to the DUT output.</td>
<td>SENSE:DISTortion:PATH:DUT:OUTPut</td>
<td>None</td>
</tr>
<tr>
<td>Enables or disables the external modulated source to pass through the attenuator of the specified PNA source</td>
<td>SENSE:DISTortion:PATH:SOURce:ATTenuation:INCLUDe</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the nominal gain of an external source amplifier.</td>
<td>SENSE:DISTortion:PATH:SOURce:NOMinal:AMPLifier</td>
<td>None</td>
</tr>
<tr>
<td>Enables or disables receiver timing auto generation from the source pulse timing.</td>
<td>SENSE:DISTortion:PULSE:RECeiver:AUTO</td>
<td>None</td>
</tr>
<tr>
<td>Action</td>
<td>Command</td>
<td>Default</td>
</tr>
<tr>
<td>------------------------------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Sets and returns the carrier center frequency.</td>
<td>SENSe:DIStortion:SWEEP:CARRier:FREQuency</td>
<td>None</td>
</tr>
<tr>
<td>Selects between a ramp type power sweep or a list of power values to define a power sweep.</td>
<td>SENSe:DIStortion:SWEEP:POWER:CARRier:LEVel:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Adds a new row to the power sweep list table.</td>
<td>SENSe:DIStortion:SWEEP:POWER:CARRier:LIST:ADD</td>
<td>None</td>
</tr>
<tr>
<td>Deletes the specified row from the power sweep list table.</td>
<td>SENSe:DIStortion:SWEEP:POWER:CARRier:LIST:DELETE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the power level used for the specified row in the power sweep list table.</td>
<td>SENSe:DIStortion:SWEEP:POWER:CARRier:LIST:LEVel</td>
<td>None</td>
</tr>
<tr>
<td>Specifies the file path to recall a previous power sweep list file.</td>
<td>SENSe:DIStortion:SWEEP:POWER:CARRier:LIST:LOAD</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the noise bandwidth setting used for the specified row in the power sweep list table.</td>
<td>SENSe:DIStortion:SWEEP:POWER:CARRier:LIST:NBW</td>
<td>None</td>
</tr>
<tr>
<td>Selects the noise bandwidth mode in the power sweep list table.</td>
<td>SENSe:DIStortion:SWEEP:POWER:CARRier:LIST:NBW:MODE</td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Sets and returns the number of power points to measure for the power sweep list measurement.</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LIST:POINts</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the receiver attenuation for the specified row in the power sweep list table.</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LIST:RECeiver:ATTenuation</td>
<td>None</td>
</tr>
<tr>
<td>Selects the receiver attenuation mode in the power sweep list table.</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LIST:RECeiver:ATTenuation:MODE</td>
<td>None</td>
</tr>
<tr>
<td>Specifies the file path to save a power sweep list file.</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LIST:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the source attenuation for the specified row in the power sweep list table.</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LIST:SOURce:ATTenuation</td>
<td>None</td>
</tr>
<tr>
<td>Selects the source attenuation mode in the power sweep list table.</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LIST:SOURce:ATTenuation:MODE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the starting power level for the power sweep</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:RAMP:LEVEL:STARt</td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Sets and returns the stop power level for the power sweep ramp measurement.</td>
<td><code>SENSe:DIStortion:SWEep:POWer:CARRier:RAMP:LEVEL:STOP</code></td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable increasing noise bandwidth at high powers automatically for faster measurements.</td>
<td><code>SENSe:DIStortion:SWEep:POWer:CARRier:RAMP:NBW:AUTO</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the number of power points to measure for the power sweep ramp measurement.</td>
<td><code>SENSe:DIStortion:SWEep:POWer:CARRier:RAMP:POINts</code></td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable the re-use of the previous linear S-parameter measurements if available and skip sweep.</td>
<td><code>SENSe:DIStortion:SWEep:SPARam:REUSe</code></td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable noise sweep used for the distortion test.</td>
<td><code>SENSe:DIStortion:SWEep:NOISe</code></td>
<td>None</td>
</tr>
<tr>
<td>Enables or disables S-</td>
<td><code>SENSe:DIStortion:SWEep:SPARam[:STATe]</code></td>
<td>None</td>
</tr>
<tr>
<td>Parameter</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Sets and returns the power level used to measure the DUT gain when operating in its linear region.</td>
<td>SENSE:DISTortion:SWEep:POWer:SPARam:LEVel</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the IF bandwidth for the linear S21 sweep.</td>
<td>SENSE:DISTortion:SWEep:SPARam:BWIDth</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the step size for the linear S21 sweep.</td>
<td>SENSE:DISTortion:SWEep:SPARam:STEP</td>
<td>None</td>
</tr>
<tr>
<td>Selects between a chirp signal from the external signal generator or an internal CW signal in the VNA for measuring S-parameters.</td>
<td>SENSE:DISTortion:SWEep:SPARam:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the power level used for the distortion test at either the input or output of the DUT.</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LEVel</td>
<td>None</td>
</tr>
<tr>
<td>Set and read where to apply power to the</td>
<td>SENSE:DISTortion:SWEep:POWer:CARRier:LEVel:PORT</td>
<td>None</td>
</tr>
<tr>
<td>Command Description</td>
<td>Command Syntax</td>
<td>Value</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>DUT (input or output)</td>
<td>SENSe:DISTrortion:SWEep:RETRace:POWer</td>
<td>None</td>
</tr>
<tr>
<td>Keeps RF power ON at the End of Sweep.</td>
<td>SENSe:DISTrortion:SWEep:DWELI</td>
<td>None</td>
</tr>
<tr>
<td>Sets and returns the delay in seconds before the test signal to allow the RF source to settle.</td>
<td>SENSe:DISTrortion:SWEep:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Selects between a fixed or power sweep type of sweep.</td>
<td>SENSe:DISTrortion:TABLe:DISPlay:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Specifies the file path to save a modulation distortion table file.</td>
<td>SENSe:DISTrortion:TABLe:DISPlay:SORT</td>
<td>None</td>
</tr>
<tr>
<td>Sort distortion table by band or power.</td>
<td>SENSe:DISTrortion:TABLe:DISPlay:SORT</td>
<td>None</td>
</tr>
<tr>
<td>Return the last monitor result of LO feedthru monitor.</td>
<td>SENSe:SA:COHerance:LO:FTHRu:MONitor:LAST:VALue</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the target receiver</td>
<td>SENSe:SA:COHerance:LO:FTHRu:MONitor:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>for LO Feedthru monitor.</td>
<td>Set and read the LO feedthru monitoring state.</td>
<td>SENSE:SA:COHerence:LO:FTHRu:MONitor[:STATe]</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Set and read the threshold level for warning.</td>
<td>SENSE:SA:COHerence:LO:FTHRu:MONitor:TOLerance</td>
</tr>
<tr>
<td></td>
<td>Set and read the monitoring method for LO feedthru monitor.</td>
<td>SENSE:SA:COHerence:LO:FTHRu:MONitor:TYPE</td>
</tr>
</tbody>
</table>

## Calibrate Source Modulation Commands
<table>
<thead>
<tr>
<th>Set and read the ACP modulation calibration state.</th>
<th>SOURce:MODulation:CORRection:COLLection:ACP:ENABle</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and reads the maximum number of iterations for the lower ACP modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:LOWer:ITERations</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the receiver for the lower ACP modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:LOWer:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration span for a lower ACP modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:LOWer:SPAN</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired ACP calibration</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:LOWer:TOLerance</td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Tolerance for the lower ACP modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:ACP:UPPer:ENABle</td>
<td>None</td>
</tr>
<tr>
<td>Enables or disables the upper ACP (ACPUp) modulation calibration.</td>
<td>SOURCE:MODulation:CORRection:COLLection:ACP:UPPer:ITERations</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations for the upper ACP modulation calibration.</td>
<td>SOURCE:MODulation:CORRection:COLLection:ACP:UPPer:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the receiver for the upper ACP modulation calibration.</td>
<td>SOURCE:MODulation:CORRection:COLLection:ACP:UPPer:SPAN</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration span for an upper ACP modulation calibration.</td>
<td>SOURCE:MODulation:CORRection:COLLection:ACP:UPPer:TOLerance</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired ACP calibration tolerance for the upper ACP modulation calibration.</td>
<td>SOURCE:MODulation:CORRection:COLLection:ACP:LOWer:GBANd</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the ACP lower frequency delta from the edge of the carrier to the</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Default</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Sets and reads the ACP upper frequency delta from the edge of the carrier to the beginning of the calibration span.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:ACP:UPPer:GBANd</code></td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable the distortion calibration state.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:DISTortion:ENABLE</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations used by the calibration routine.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:DISTortion:ITERations</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the receiver for distortion calibration.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:DISTortion:RECeiver</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration span for a distortion calibration.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:DISTortion:SPAN</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired distortion calibration tolerance.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:DISTortion:TOLerance</code></td>
<td>None</td>
</tr>
<tr>
<td>Enable or disable the Equalization.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:EQUalization:ENABLE</code></td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Example</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>equalization calibration state.</td>
<td>SOURce:MODulation:CORRection:COLLection:EQUalization:ITERations</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations used by the calibration routine.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sets and reads the receiver for equalization calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:EQUalization:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration span for a equalization calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:EQUalization:SPAN</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired equalization calibration tolerance.</td>
<td>SOURce:MODulation:CORRection:COLLection:EQUalization:TOLerance</td>
<td>None</td>
</tr>
<tr>
<td>Returns a list of the calibrations stored in the .mdx file.</td>
<td>SOURce:MODulation:FILE:CORRection:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Deletes any of the calibration files stored in the .mdx file.</td>
<td>SOURce:MODulation:FILE:CORRection:DELETE?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the frequency of the specified source modulation calibration file.</td>
<td>SOURce:MODulation:FILE:CORRection:FREQuency?</td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Default</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>Returns the power level of the specified source modulation calibration file.</td>
<td><code>SOURce:MODulation:FILE:CORRection:POWer?</code></td>
<td>None</td>
</tr>
<tr>
<td>Append source modulation calibration files stored in the .mdx file.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:APPend</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads a fixed frequency.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:FREQuency[:FIXed]</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the number of frequency measurement points.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:FREQuency:POINts</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the start frequency to use for a swept frequency source modulation calibration.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:FREQuency:STARt</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the stop frequency to use for a swept frequency calibration.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:FREQuency:STOP</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration frequency type to fixed or swept.</td>
<td><code>SOURce:MODulation:CORRection:COLLection:FREQuency:TYPE</code></td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Default</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Enable or disable the LO feedthru calibration state.</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FTHRu:ENABle</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations used by the calibration routine.</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FTHRu:ITERations</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the receiver for LO feedthru modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FTHRu:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration span for a LO feedthru modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FTHRu:SPAN</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired LO feedthru modulation calibration tolerance.</td>
<td>SOURce:MODulation:CORRection:COLLection:LO:FTHRu:TOLerance</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads a fixed power level.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer[:FIXed]</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the number of power measurement points.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:POINts</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the start power level to use for a swept calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:STARt</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the stop power level to use for a swept calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:STOP</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration power type to fixed or swept.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:TYPE</td>
<td>None</td>
</tr>
</tbody>
</table>

**Create/Edit Modulation File**

<p>| Sets and reads the source sample rate. | SOURce:MODulation:FILE:INITialize | None |
| Specifies the file path to recall a previous modulation file. | SOURce:MODulation:FILE:LOAD | None |
| Sets and reads the source sample rate. | SOURce:MODulation:FILE:SAVE | None |
| Sets and reads the Carrier offset value. | SOURce:MODulation:FILE:SIGNal:CARRier:OFFSet | None |
| Sets and reads the number of compact modulation files to create. | SOURce:MODulation:FILE:SIGNal:COMPact:FILE:NUMBer | None |
| Sets and reads the compact | SOURce:MODulation:FILE:SIGNal:COMPact:FILE:NUMBer:SELect | None |</p>
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the file path to recall a previous modulation file from which to create a compact signal.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns the peak-to-average value of the original signal.</td>
<td><strong>SOURce:MODulation:FILE:SIGNal:COMPact:PAVG?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Returns the peak-to-average value of the signal created from the original signal.</td>
<td><strong>SOURce:MODulation:FILE:SIGNal:COMPact:PAVG:CALCulated?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the Peak-to-Avg priority for Compact signals.</td>
<td><strong>SOURce:MODulation:FILE:SIGNal:COMPact:PAVG:PRIority</strong></td>
<td>None</td>
</tr>
<tr>
<td>Allows setup of multiple carriers when defining a multicarrier signal.</td>
<td><strong>SOURce:MODulation:FILE:SIGNal:COMPact:SUBCarrier:NUMBER</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the offset of the selected subcarrier.</td>
<td><strong>SOURce:MODulation:FILE:SIGNal:COMPact:SUBCarrier:OFFSet</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the span of the selected subcarrier.</td>
<td><strong>SOURce:MODulation:FILE:SIGNal:COMPact:SUBCarrier:SPAN</strong></td>
<td>None</td>
</tr>
<tr>
<td>Function Description</td>
<td>Command</td>
<td>Default Value</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Sets and reads where to start the compact signal within the original signal.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:TIME:STARt</td>
<td>None</td>
</tr>
<tr>
<td>Returns the calculated start time of the signal created from the original signal.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:TIME:STARt:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the start time priority for Compact signals.</td>
<td>SOURce:MODulation:FILE:SIGNal:COMPact:TIME:STARt:PRIority</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the DAC scale as a percentage of full scale.</td>
<td>SOURce:MODulation:FILE:SIGNal:DAC:SCALing</td>
<td>None</td>
</tr>
<tr>
<td>Sets the NPR notch location type for the selected NPR Notch modulation type.</td>
<td>SOURce:MODulation:FILE:SIGNal:NPR:NOTCh:LOCation</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the number of NPR notches for NPR Notch modulation type.</td>
<td>SOURce:MODulation:FILE:SIGNal:NPR:NOTCh:NUMBer</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the NPR notch offset frequency of the selected notch.</td>
<td>SOURce:MODulation:FILE:SIGNal:NPR:NOTCh:OFFSet</td>
<td>None</td>
</tr>
<tr>
<td>Function Description</td>
<td>Command</td>
<td>Default Value</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Sets and reads the span of the selected notch.</td>
<td>SOURce:MODulation:FILE:SIGNal:NPR:NOTCh:SPAN</td>
<td>None</td>
</tr>
<tr>
<td>Enables or disables signal optimization settings.</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:ENABLE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the number of decimal digits limit for calculated frequencies.</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:FREQuency:LIMIT:DDIGits</td>
<td>None</td>
</tr>
<tr>
<td>Enables or disables the number of decimal digits limit for calculated frequencies.</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:FREQuency:LIMIT:ENABLE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the number of test signal harmonics you want to be protected against.</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:HREJect</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the maximum distance between each tone.</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:MAX:TONE:SPacing</td>
<td>None</td>
</tr>
<tr>
<td>Function Description</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Sets and reads the minimum number of tones.</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:MIN:TONE:NUMBER</td>
<td>None</td>
</tr>
<tr>
<td>Enables or disables the rejection of Nyquist frequencies.</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:NYQReject:ENABLE</td>
<td>None</td>
</tr>
<tr>
<td>Sets the optimize signal type.</td>
<td>SOURce:MODulation:FILE:SIGNal:OPTimize:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Returns the calculated peak-to-average value.</td>
<td>SOURce:MODulation:FILE:SIGNal:PAVG:CALculated?</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the phase when Fixed phase is the Phase Type.</td>
<td>SOURce:MODulation:FILE:SIGNal:PHASE:FIXed</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the phase seed when Random phase is the Phase Type.</td>
<td>SOURce:MODulation:FILE:SIGNal:PHASE:RANDom:SEED</td>
<td>None</td>
</tr>
<tr>
<td>Sets the phase type.</td>
<td>SOURce:MODulation:FILE:SIGNal:PHASE:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the signal span.</td>
<td>SOURce:MODulation:FILE:SIGNal:SPAN</td>
<td>None</td>
</tr>
<tr>
<td>Returns the calculated signal span.</td>
<td>SOURce:MODulation:FILE:SIGNal:SPAN:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td>Operation</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Sets and reads the signal span priority.</td>
<td>SOURce:MODulation:FILE:SIGNal:SPAN:PRIority</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the source sample rate.</td>
<td>SOURce:MODulation:FILE:SIGNal:SRATe</td>
<td>None</td>
</tr>
<tr>
<td>Returns the calculated sampling rate of the signal created from the original signal.</td>
<td>SOURce:MODulation:FILE:SIGNal:SRATe:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the auto sample rate.</td>
<td>SOURce:MODulation:FILE:SIGNal:SRATe:AUTO</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the number of tones.</td>
<td>SOURce:MODulation:FILE:SIGNal:TONE:NUMBer</td>
<td>None</td>
</tr>
<tr>
<td>Returns the calculated number of tones.</td>
<td>SOURce:MODulation:FILE:SIGNal:TONE:NUMBer:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the tone number priority.</td>
<td>SOURce:MODulation:FILE:SIGNal:TONE:NUMBer:PRIority</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the tone spacing.</td>
<td>SOURce:MODulation:FILE:SIGNal:TONE:SPACing</td>
<td>None</td>
</tr>
<tr>
<td>Returns the calculated spacing between the tones.</td>
<td>SOURce:MODulation:FILE:SIGNal:TONE:SPACing:CALCulated?</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the tone spacing priority.</td>
<td>SOURce:MODulation:FILE:SIGNal:TONE:SPACing:PRIority</td>
<td>None</td>
</tr>
<tr>
<td>Returns the number of tones.</td>
<td>SOURce:MODulation:FILE:TONE:NUMBer?</td>
<td>None</td>
</tr>
<tr>
<td>Returns the tone frequency in Hz</td>
<td>SOURce:MODulation:FILE:TONE:FREQuency?</td>
<td>None</td>
</tr>
<tr>
<td>Command</td>
<td>Syntax</td>
<td>Default</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>---------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Sets and reads the power in dBm of the specified tone number.</td>
<td><code>SOURce:MODulation:FILE:TONE:POWer</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the phase in degrees of the specified tone number.</td>
<td><code>SOURce:MODulation:FILE:TONE:PHASE</code></td>
<td>None</td>
</tr>
<tr>
<td>Set specified tone state to on or off.</td>
<td><code>SOURce:MODulation:FILE:TONE[:STATE]</code></td>
<td>None</td>
</tr>
<tr>
<td>Set all tone states to on or off.</td>
<td><code>SOURce:MODulation:FILE:TONE:ALL[:STATE]</code></td>
<td>None</td>
</tr>
<tr>
<td>Saves the specified multitone file.</td>
<td><code>SOURce:MODulation:FILE:TONE:SAVE</code></td>
<td>None</td>
</tr>
<tr>
<td>Loads the specified multitone file.</td>
<td><code>SOURce:MODulation:FILE:TONE:LOAD</code></td>
<td>None</td>
</tr>
<tr>
<td>Sets the modulation type.</td>
<td><code>SOURce:MODulation:FILE:TYPE</code></td>
<td>None</td>
</tr>
<tr>
<td>Distortion Table Commands</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns the currently displayed Distortion Table parameters.</td>
<td><code>SENSe:DISTortion:TABLE:DISPlay:CATalog?</code></td>
<td>None</td>
</tr>
<tr>
<td>Deletes the specified parameter from</td>
<td><code>SENSe:DISTortion:TABLE:DISPlay:DELeTe</code></td>
<td>None</td>
</tr>
<tr>
<td>Task</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Adds the specified parameter to the Distortion Table.</td>
<td>SENSE:DISTortion:TABLE:DISPLAY:FEED</td>
<td>None</td>
</tr>
<tr>
<td>Returns a list of existing band names from the Measurement Band Table.</td>
<td>SENSE:DISTortion:TABLE:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Returns a list of data parameter names corresponding to the currently specified Measurement Type.</td>
<td>SENSE:DISTortion:TABLE:DATA:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Returns a list of data values from the Distortion Table for the specified band and parameter name.</td>
<td>SENSE:DISTortion:TABLE:DATA:VALUE?</td>
<td>None</td>
</tr>
</tbody>
</table>

**Modulation Distortion Marker Commands**

<table>
<thead>
<tr>
<th>Task</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set and read the SA marker function type.</td>
<td>CALCulate:MEASure:SA:MARKer:FUNCTION</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the integration bandwidth marker.</td>
<td>CALCulate:MEASure:SA:MARKer:BAND:IBW</td>
<td>None</td>
</tr>
</tbody>
</table>
Sets and reads the ACPR density marker.
CALCulate:MEASure:SA:MARKer:BDENsity:ACPR[:STATe]
CALCulate:SA:MARKer:BDENsity:ACPR[:STATe]

Sets and reads the NPR density marker.
CALCulate:MEASure:SA:MARKer:BDENsity:NPR[:STATe]
CALCulate:SA:MARKer:BDENsity:NPR[:STATe]

Executes the search ACPR density marker.
CALCulate:MEASure:SA:MARKer:SEARch:ACPR

Executes the search NPR density marker.
CALCulate:MEASure:SA:MARKer:SEARch:NPR

Measurement Data

Retrieves trace data (Y data) from the modulation distortion measurement.
CALCulate:MEASure:DATA:BUFFer:Y?
GetDataBuffer
GetDataBufferCompact

Retrieves frequency tone data from the modulation distortion measurement.
CALCulate:MEASure:DATA:BUFFer:X?
GetXDataBuffer
GetXDataBufferCompact

Phase Noise Application Commands

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:MEASure:PN:AVARiance:DEViation</td>
<td>None</td>
</tr>
<tr>
<td>CALCulate:MEASure:PN:AVARiance:JITTer</td>
<td>None</td>
</tr>
<tr>
<td>CALCulate:MEASure:PN:AVARiance:VARiance</td>
<td>None</td>
</tr>
<tr>
<td>Function Description</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>Returns the measured carrier frequency.</td>
<td>CALCulate:MEASure:PN:CARRier:FREQuency?</td>
</tr>
<tr>
<td>Returns the measured carrier power level.</td>
<td>CALCulate:MEASure:PN:CARRier:LEVEL?</td>
</tr>
<tr>
<td>Sets and returns the dBC data array for the selected measurement.</td>
<td>CALCulate:MEASure:PN:DATA:PDATa</td>
</tr>
<tr>
<td>Sets and returns the dBC memory data for the selected measurement.</td>
<td>CALCulate:MEASure:PN:DATA:PMEMory</td>
</tr>
<tr>
<td>Returns the spurious data (0 or 1) for the selected measurement.</td>
<td>CALCulate:MEASure:PN:DATA:SPData</td>
</tr>
<tr>
<td>Returns the spurious memory data (0 or 1) for the selected measurement.</td>
<td>CALCulate:MEASure:PN:DATA:SPMemory</td>
</tr>
<tr>
<td>Returns the specified data for the selected range number.</td>
<td>CALCulate:MEASure:PN:INTegral:RANGe:DATA</td>
</tr>
<tr>
<td>Sets and returns the start frequency of the selected integration range.</td>
<td>CALCulate:MEASure:PN:INTegral:RANGe:STARt</td>
</tr>
<tr>
<td>Sets and returns the stop frequency of the selected integration range.</td>
<td>CALCulate:MEASure:PN:INTegral:RANGe:STOP</td>
</tr>
<tr>
<td>Sets and returns the integration range type of the selected integration range.</td>
<td>CALCulate:MEASure:PN:INTegral:RANGe:TYPE</td>
</tr>
<tr>
<td>Sets and returns the weighting filter of the selected integration range.</td>
<td>CALCulate:MEASure:PN:INTegral:RANGe:WEIGhting</td>
</tr>
<tr>
<td>Enables and disables the spot noise calculation on every decade offset frequency.</td>
<td>CALCulate:MEASure:PN:SNOise:DECades[:STATe]</td>
</tr>
<tr>
<td>Returns the spot noise x-axis array of all decade offset frequencies.</td>
<td>CALCulate:MEASure:PN:SNOise:DECades:X</td>
</tr>
<tr>
<td>Returns the spot noise y-axis array of all decade offset frequencies.</td>
<td>CALCulate:MEASure:PN:SNOise:DECades:Y</td>
</tr>
<tr>
<td>Enables and disables spot noise calculation for the selected measurement number.</td>
<td>CALCulate:MEASure:PN:SNOise[:STATe]</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Enables and disables spot noise calculation for the specified user-defined offset frequency.</td>
<td>CALCulate:MEASure:PN:SNOise:USER[:STATE]</td>
</tr>
<tr>
<td>Sets and returns the offset frequency on which the spot noise is calculated.</td>
<td>CALCulate:MEASure:PN:SNOise:USER:X</td>
</tr>
<tr>
<td>Returns the spot noise y-axis value of the specified offset frequency.</td>
<td>CALCulate:MEASure:PN:SNOise:USER:Y</td>
</tr>
<tr>
<td>Enables and disables spurious analysis.</td>
<td>CALCulate:MEASure:PN:SPURious:ANALysis[:STATE]</td>
</tr>
<tr>
<td>Returns a list of detected spurs.</td>
<td>CALCulate:MEASure:PN:SPURious:DATA</td>
</tr>
<tr>
<td>Enables and disables spur omission.</td>
<td>CALCulate:MEASure:PN:SPURious:OMISSion[:STATE]</td>
</tr>
<tr>
<td>Sets and returns the User Spur Table data which defines spurs to omit.</td>
<td>CALCulate:MEASure:PN:SPURious:OSSPur:DATA</td>
</tr>
<tr>
<td>Deletes the User Spur Table data which defines spurs to omit.</td>
<td>CALCulate:MEASure:PN:SPURious:OSSPur:DELETE</td>
</tr>
<tr>
<td>Enables and disables user specified spur omission.</td>
<td>CALCulate:MEASure:PN:SPURious:OSSPur[:STATE]</td>
</tr>
<tr>
<td>Sets and returns the spurious sensibility number.</td>
<td>CALCulate:MEASure:PN:SPURious:SENSibility</td>
</tr>
<tr>
<td>Sets and returns the spurious table sorting order.</td>
<td>CALCulate:MEASure:PN:SPURious:SORT</td>
</tr>
<tr>
<td>Sets and returns the minimum spurious threshold level.</td>
<td>CALCulate:MEASure:PN:SPURious:THReshold:LEVel:MINimum</td>
</tr>
<tr>
<td>Sets and returns the spurious threshold table data.</td>
<td>CALCulate:MEASure:PN:SPURious:THReshold:TABLE:DATA</td>
</tr>
<tr>
<td>Deletes the spurious threshold table.</td>
<td>CALCulate:MEASure:PN:SPURious:THReshold:TABLE:DELETE</td>
</tr>
<tr>
<td>Enable or disable displaying the integrated noise table.</td>
<td>DISPLAY:WINDow:TABLE:INOise:ENABLE</td>
</tr>
<tr>
<td>Enable or disable displaying the spot noise table.</td>
<td>DISPLAY:WINDow:TABLE:SNOise:ENABLE</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Enable or disable displaying the spurious table.</td>
<td><code>DISPLAY:WINDOW:TABLE:SPURious:ENABLE</code></td>
</tr>
<tr>
<td>Load user specified spurious frequency list of the active trace of the active channel.</td>
<td><code>MEMORY:LOAD:PN:SPURious:OSSPur</code></td>
</tr>
<tr>
<td>Load the threshold table of the active trace of the active channel.</td>
<td><code>MEMORY:LOAD:PN:SPURious:THRESHold</code></td>
</tr>
<tr>
<td>Enables and disables check for carrier.</td>
<td><code>SENSe:PN:ADJust:CONFigure:FREQuency:CHECK</code></td>
</tr>
<tr>
<td>Sets and returns the high frequency limit to use during a broadband carrier search.</td>
<td><code>SENSe:PN:ADJust:CONFigure:FREQuency:LIMit:HIGH</code></td>
</tr>
<tr>
<td>Sets and returns the low frequency limit to use during a broadband carrier search.</td>
<td><code>SENSe:PN:ADJust:CONFigure:FREQuency:LIMit:LOW</code></td>
</tr>
<tr>
<td>Enables and disables a broadband carrier search.</td>
<td><code>SENSe:PN:ADJust:CONFigure:FREQuency:SEARch[:STATE]</code></td>
</tr>
<tr>
<td>Sets and returns the threshold to use during a carrier search.</td>
<td><code>SENSe:PN:ADJust:CONFigure:LEVel:THReshold</code></td>
</tr>
<tr>
<td>Sets and returns the FFT average factor number.</td>
<td><code>SENSe:PN:FAVerage:FACTor</code></td>
</tr>
<tr>
<td>Sets and returns the resolution bandwidth ratio.</td>
<td><code>SENSe:PN:BWIDth[:RESolution]:RATio</code></td>
</tr>
<tr>
<td>Sets and returns the noise type to phase or residual noise.</td>
<td><code>SENSe:PN:NTYPe</code></td>
</tr>
<tr>
<td>Sets and returns the receiver for the phase noise measurement.</td>
<td><code>SENSe:PN:RECeiver</code></td>
</tr>
<tr>
<td>Sets and returns receiver at the DUT input for residual phase noise measurements.</td>
<td><code>SENSe:PN:RESidual:INPut</code></td>
</tr>
<tr>
<td>Sets and returns receiver at the DUT output for residual phase noise measurements.</td>
<td><code>SENSe:PN:RESidual:OUTput</code></td>
</tr>
<tr>
<td>Sets and returns the carrier frequency.</td>
<td><code>SENSe:PN:SWEep:CARRier:FREQuency</code></td>
</tr>
<tr>
<td>Marker Search Commands</td>
<td>Command</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Sets and returns the sweep noise mode.</td>
<td>SENSE:PN:SWEep:NOISe:MODE</td>
</tr>
<tr>
<td>Spurious Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:EXECute SPURious</td>
</tr>
<tr>
<td>Spurious Right &gt;&gt; Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:EXECute RSPurious</td>
</tr>
<tr>
<td>&lt;&lt; Spurious Left Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:EXECute LSPurious</td>
</tr>
<tr>
<td>Multi Spurious Search</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:MULTi:EXECute SPURious</td>
</tr>
</tbody>
</table>
Replacement Commands

As we expand the capability of the VNA, we will continue to develop new commands. Sometimes, a command is replaced with a new command that delivers more capability. The old (superseded) command will continue to work as usual, but we recommend using the new (replacement) command in new code that you develop.

In very few cases, commands become obsolete and no longer work as before. Obsolete commands will likely NOT have a replacement.

### COM Obsolete Commands
- SCPI Replacement Commands
- SCPI Obsolete Commands

### COM Replacement Commands

<table>
<thead>
<tr>
<th>Superseded Command</th>
<th>Replacement Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acquire Cal Standard Method</td>
<td>Acquire Cal Standard2 Method</td>
</tr>
<tr>
<td>Create SParameterEX Method</td>
<td>Create SParameter Method</td>
</tr>
<tr>
<td>CreateCustomMeasurement Method</td>
<td>CreateCustomMeasurementEx Method</td>
</tr>
<tr>
<td>Calibrator.getErrorTerm</td>
<td>GetErrorTermByString_Method</td>
</tr>
<tr>
<td>Calibrator.getStandard</td>
<td>GetStandardByString_Method</td>
</tr>
<tr>
<td>Calibrator.putErrorTerm</td>
<td>PutErrorTermByString_Method</td>
</tr>
<tr>
<td>Calibrator.putStandard</td>
<td>PutStandardByString_Method</td>
</tr>
<tr>
<td>AcquireCalConfidenceCheckECAL</td>
<td>AcquireCalConfidenceCheckECALEx</td>
</tr>
<tr>
<td>DoECAL1Port Method</td>
<td>DoECAL1PortEx Method</td>
</tr>
<tr>
<td>DoECAL2Port Method</td>
<td>DoECAL2PortEx Method</td>
</tr>
<tr>
<td>ECALCharacterization Property</td>
<td>ECALCharacterizationEx Property</td>
</tr>
<tr>
<td>ECALPortMap_Property</td>
<td>ECALPortMapEx_Property</td>
</tr>
<tr>
<td>Get ECAL Module Info Method</td>
<td>Get ECALModuleInfoEx Method</td>
</tr>
<tr>
<td>IsECALModuleFound Property</td>
<td>IsECALModuleFoundEx Property</td>
</tr>
<tr>
<td>Trigger Signal Property</td>
<td>Source Property</td>
</tr>
<tr>
<td>Trigger Type Property</td>
<td>Scope Property</td>
</tr>
<tr>
<td>get SourcePowerCalData Method</td>
<td>Get SourcePowerCalDataEx Method</td>
</tr>
<tr>
<td>get SourcePowerCalDataScalar Method</td>
<td>Get SourcePowerCalDataScalarEx Method</td>
</tr>
<tr>
<td>Method/Property</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>put SourcePowerCalData Method</td>
<td></td>
</tr>
<tr>
<td>put SourcePowerCalDataScalar Method</td>
<td></td>
</tr>
<tr>
<td>ConfigNarrowBand Method</td>
<td></td>
</tr>
<tr>
<td>ConfigNarrowBand2 Method</td>
<td></td>
</tr>
<tr>
<td>StandardForClass Property</td>
<td></td>
</tr>
<tr>
<td>Port 1 Property</td>
<td></td>
</tr>
<tr>
<td>Port 2 Property</td>
<td></td>
</tr>
<tr>
<td>Port_3_Property</td>
<td></td>
</tr>
<tr>
<td>Normalization Property</td>
<td></td>
</tr>
<tr>
<td>InterpolateNormalization Property</td>
<td></td>
</tr>
<tr>
<td>Get ErrorTermComplex2 Method</td>
<td></td>
</tr>
<tr>
<td>Get ErrorTermList Method</td>
<td></td>
</tr>
<tr>
<td>Get Standard Complex Method</td>
<td></td>
</tr>
<tr>
<td>Put ErrorTermComplex2 Method</td>
<td></td>
</tr>
<tr>
<td>Put Standard Complex Method</td>
<td></td>
</tr>
<tr>
<td>Get ErrorTerm2 Method</td>
<td></td>
</tr>
<tr>
<td>Get Standard2 Method</td>
<td></td>
</tr>
<tr>
<td>Get StandardsList Method</td>
<td></td>
</tr>
<tr>
<td>Put ErrorTerm2 Method</td>
<td></td>
</tr>
<tr>
<td>Put Standard2 Method</td>
<td></td>
</tr>
<tr>
<td>DataTableDivisor</td>
<td></td>
</tr>
<tr>
<td>LogMagnitudeOffset</td>
<td></td>
</tr>
<tr>
<td>Normalization</td>
<td></td>
</tr>
<tr>
<td>InterpolateNormalization</td>
<td></td>
</tr>
<tr>
<td>Save Cal Sets Method</td>
<td></td>
</tr>
<tr>
<td>app.Save .SNP</td>
<td></td>
</tr>
<tr>
<td>Get SnPData Method</td>
<td></td>
</tr>
<tr>
<td>AcquirePowerReadings Method</td>
<td></td>
</tr>
<tr>
<td>SetCalInfo2 (power meter) Method</td>
<td></td>
</tr>
<tr>
<td>Put SourcePowerCalDataEx Method</td>
<td></td>
</tr>
<tr>
<td>Put SourcePowerCalDataScalarEx Method</td>
<td></td>
</tr>
<tr>
<td>ConfigNarrowBand2 Method</td>
<td></td>
</tr>
<tr>
<td>ConfigNarrowBand3 Method</td>
<td></td>
</tr>
<tr>
<td>GetStandardForClass_Method</td>
<td></td>
</tr>
<tr>
<td>SetStandardForClass_Method</td>
<td></td>
</tr>
<tr>
<td>PortDelay Property</td>
<td></td>
</tr>
<tr>
<td>DoReceiverPowerCal Method</td>
<td></td>
</tr>
<tr>
<td>Interpolate Correction Property</td>
<td></td>
</tr>
<tr>
<td>GetErrorTermComplexByString_Method</td>
<td></td>
</tr>
<tr>
<td>GetErrorTermList2_Method</td>
<td></td>
</tr>
<tr>
<td>GetStandardComplexByString_Method</td>
<td></td>
</tr>
<tr>
<td>PutErrorTermComplexByString_Method</td>
<td></td>
</tr>
<tr>
<td>PutStandardComplexByString_Method</td>
<td></td>
</tr>
<tr>
<td>GetErrorTermByString_Method</td>
<td></td>
</tr>
<tr>
<td>Get StandardByString_Method</td>
<td></td>
</tr>
<tr>
<td>Get StandardList2_Method</td>
<td></td>
</tr>
<tr>
<td>PutErrorTermByString_Method</td>
<td></td>
</tr>
<tr>
<td>PutStandardByString_Method</td>
<td></td>
</tr>
<tr>
<td>Save (CalSet) Method</td>
<td></td>
</tr>
<tr>
<td>WriteSnpFileWithSpecifiedPorts Method</td>
<td></td>
</tr>
<tr>
<td>Get SnpDataWithSpecifiedPorts Method</td>
<td></td>
</tr>
<tr>
<td>AcquirePowerReadingsEx Method</td>
<td></td>
</tr>
<tr>
<td>SetCalInfoEx (power meter) Method</td>
<td></td>
</tr>
</tbody>
</table>
ThruCalMethod Property
PathCalMethod Property
PathThruMethod Property
ApplyPowerCorrectionValues Method
ApplyPowerCorrectionValuesEx Method
PowerMeterGPIBAddress Property
PowerMeterInterface Object
get InputVoltage Method
get InputVoltageEX Method
Initialize (ECal)
InitializeEx Method
guided.CompatibleCalKits Property
guided.GetCompatibleCalKits Method
AssignSourceToRole Method
RoleDevice Property
GetSourceByRole Method
DefinedRoles Property
GetSourceRoles Method
DefinedRoles Property

COM Obsolete Commands

Input A Property
No replacement
Input B Property
No replacement
Input_C_Property
No replacement
ECALIsolation Property
No replacement
CalibrationPort Property
No replacement

SCPI Replacement Commands

<table>
<thead>
<tr>
<th>Superseded Command</th>
<th>Replacement Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRIGger[:SEQUence]:LEVEL</td>
<td>CONTrol:SIGNal</td>
</tr>
<tr>
<td>DISPlay:TILE</td>
<td>DISP:ARRange</td>
</tr>
<tr>
<td>CALC:NORM</td>
<td>SENS:CORR:COLL:METH</td>
</tr>
<tr>
<td>CALC:NORM:STAT</td>
<td>SENS:CORR:COLL</td>
</tr>
<tr>
<td>CALC:NORM:INT</td>
<td>SENS:CORR</td>
</tr>
<tr>
<td>SENS:CORR:CKIT:INITialize</td>
<td></td>
</tr>
</tbody>
</table>
### SCPI Obsolete Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Replacement</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:EXT:REC</td>
<td>No replacement</td>
</tr>
<tr>
<td>SENS:CORR:ISOLation</td>
<td>No replacement</td>
</tr>
<tr>
<td>Calc:Data SCORR</td>
<td>SENS:CORR:CSET:DATA</td>
</tr>
<tr>
<td>HCOP:ITEM:SEGData:STATe</td>
<td>No replacement</td>
</tr>
<tr>
<td>DISP:TOOL:MEAS</td>
<td>No replacement</td>
</tr>
<tr>
<td>DISP:TOOL:STIM</td>
<td>No replacement</td>
</tr>
<tr>
<td>DISP:TOOL:SWEep</td>
<td>No replacement</td>
</tr>
</tbody>
</table>
RF PathConfig

The tables below show RF Path Configuration Elements for all PNA-X models and options. These settings are provided for configuring the elements using SCPI and COM commands.

- PNA-X Opt 423
- PNA-X Opt 224
- PNA-X Opt 201
- PNA-X Opt 401
- PNA-X Opt 029

26.5 GHz Noise Figure
50 GHz Noise Figure

See Also
- IF Path Configuration Elements
- COM commands
- SCPI commands

### PNA-X - 201  2 ports, single source

<table>
<thead>
<tr>
<th>Ref#</th>
<th>Element Name</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>&quot;Src1Out1LowBand&quot;</td>
<td>&quot;Filtered&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;HiPwr&quot;</td>
</tr>
<tr>
<td>#2</td>
<td>&quot;Port1RefMxr&quot;</td>
<td>&quot;Internal&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;External&quot;</td>
</tr>
</tbody>
</table>
PNA-X - 224  2 ports, add internal 2nd source, combiner & mechanical switches

Ref# | Element Name | Settings
---|-------------|----------
#1 | "Combiner" | "Normal"  
This selection only draws connection lines. No switches are thrown.  
"Reversed"
#2 | "Src1Out1LowBand" | "Filtered"  
"HiPwr"
#3 | "Src2Out1LowBand" | "Filtered"  
"HiPwr"
#4 | "Port1Bypass" | "Thru"  
"Combiner"
#5 | "Port2Bypass" | "Thru"  
"RearPanel"
#6 | "Port1RefMxr" | "Internal"  
"External"
This selection only draws connection lines. No switches are thrown.

Labeled: Src2 Main Bypass Switch

(routes Src1 signal to J6 on rear panel)

N5227A/47A only (routes Src2 signal to J12 on rear panel)

(SMC measurements only. Learn more.)

---

<table>
<thead>
<tr>
<th>Ref#</th>
<th>Element Name</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>&quot;Src1Out1LowBand&quot;</td>
<td>&quot;Filtered&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;HiPwr&quot;</td>
</tr>
<tr>
<td>#2</td>
<td>&quot;Src2Out1LowBand&quot;</td>
<td>&quot;Filtered&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;HiPwr&quot;</td>
</tr>
<tr>
<td>#3</td>
<td>&quot;Port1RefMxr&quot;</td>
<td>&quot;Internal&quot;</td>
</tr>
</tbody>
</table>

---

**PNA-X - 401 4 ports, dual source**
"External"

<table>
<thead>
<tr>
<th>Not shown</th>
<th>&quot;PortxCoupler&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(SMC measurements only. Learn more.)</td>
</tr>
<tr>
<td>&quot;Normal&quot;</td>
<td>&quot;Reversed&quot;</td>
</tr>
</tbody>
</table>

PNA-X Opt 423 4 ports, add internal combiner & mechanical switches (also second source)

<table>
<thead>
<tr>
<th>Ref#</th>
<th>Element Name</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>&quot;Combiner&quot;</td>
<td>&quot;Normal&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Reversed&quot;</td>
</tr>
<tr>
<td>#2</td>
<td>&quot;Src1Out1LowBand&quot;</td>
<td>&quot;Filtered&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;HiPwr&quot;</td>
</tr>
<tr>
<td>#3</td>
<td>&quot;Src2Out1LowBand&quot;</td>
<td>&quot;Filtered&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;HiPwr&quot;</td>
</tr>
<tr>
<td>#4</td>
<td>&quot;Port1Bypass&quot;</td>
<td>&quot;Thru&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Combiner&quot;</td>
</tr>
<tr>
<td>#5</td>
<td>&quot;Port2Bypass&quot;</td>
<td>&quot;Thru&quot;</td>
</tr>
<tr>
<td>#6</td>
<td>&quot;Port3Bypass&quot;</td>
<td>&quot;RearPanel&quot;</td>
</tr>
<tr>
<td>----</td>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>&quot;Thru&quot;</td>
<td>&quot;Combiner&quot;</td>
</tr>
<tr>
<td>#7</td>
<td>&quot;Port4Bypass&quot;</td>
<td>&quot;Thru&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;RearPanel&quot;</td>
<td></td>
</tr>
<tr>
<td>#8</td>
<td>&quot;Port1RefMxr&quot;</td>
<td>&quot;Internal&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;External&quot;</td>
<td></td>
</tr>
<tr>
<td>Not shown</td>
<td>&quot;Port2Src&quot; (Not shown in the above image) - <a href="#">See #7 in PNA-X Opt 029 - 224</a></td>
<td>&quot;Src1Out2&quot;</td>
</tr>
<tr>
<td></td>
<td>This selection only draws connection lines. No switches are thrown.</td>
<td>&quot;Src2Out1&quot;</td>
</tr>
<tr>
<td>Not shown</td>
<td>&quot;Src1RearOut&quot;</td>
<td>&quot;Normal&quot;</td>
</tr>
<tr>
<td></td>
<td>(routes Src1 signal to J6 on rear panel)</td>
<td>&quot;RFOut&quot;</td>
</tr>
<tr>
<td>Not shown</td>
<td>&quot;Src2RearOut&quot;</td>
<td>&quot;Normal&quot;</td>
</tr>
<tr>
<td></td>
<td>N5227A/47A only (routes Src2 signal to J12 on rear panel)</td>
<td>&quot;RFOut&quot;</td>
</tr>
<tr>
<td>Not shown</td>
<td>&quot;PortxCoupler&quot;</td>
<td>&quot;Normal&quot;</td>
</tr>
<tr>
<td></td>
<td>(SMC measurements only. <a href="#">Learn more.</a>)</td>
<td>&quot;Reversed&quot;</td>
</tr>
</tbody>
</table>

PNA-X Opt 029 - 26.5 GHz Models

[Diagram of RF Path Configuration and Noise Path Configuration]
PNA-X Opt 029 - 50 GHz Models

<table>
<thead>
<tr>
<th>Port1NoiseTuner</th>
<th>Internal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>External</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Port2NoiseReceiver</th>
<th>Normal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Noise</td>
</tr>
</tbody>
</table>

Change the impedance states of the built-in Noise Tuner. Choose from:

<table>
<thead>
<tr>
<th>InternalNoiseTuner</th>
<th>Impedance1 (ECal Open)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Impedance2 (ECal OffsetOpen)</td>
</tr>
<tr>
<td></td>
<td>Impedance3 (ECal Short)</td>
</tr>
<tr>
<td></td>
<td>Impedance4 (ECal OffsetShort1)</td>
</tr>
<tr>
<td></td>
<td>Impedance5 (ECal OffsetShort2)</td>
</tr>
<tr>
<td></td>
<td>Impedance6 (ECal OffsetShort3)</td>
</tr>
<tr>
<td></td>
<td>Impedance7 (ECal Load)</td>
</tr>
<tr>
<td></td>
<td>Thru (ECal Thru)</td>
</tr>
</tbody>
</table>
## Description | SCPI | COM
---|---|---
**Speed up Measurements !** |  |  
Measurement Trace On|Off | DISPlay:WINDow:TRACe[:STATe] | meas.View
Display Update On|Off | DISPlay:ENABle | None
Window Update On|Off | DISPlay:WINDow:ENABle | None
Analyzer Visible On|Off | DISPlay:VISible | app.Visible
Measurement display update | CALCulate:PARameter:SELect<name>[,fast] | None
CALCulate:PARameter:MNUMber[:SELect]<num>[,fast] |  |

### Auxiliary

**AuxInN Source Port N** | CALCulate:MEASure:PARameter | CreateCustomMeasurementEx
---|---|---

### Conversions

| Description | SCPI | COM |
---|---|---
Off | CALCulate:MEASure:CONVersion:FUNCtion | None |
Z-Reflect | CALCulate:MEASure:CONVersion:FUNCtion | None |
Z-Transmit | CALCulate:MEASure:CONVersion:FUNCtion | None |
Z-Trans-Shunt | CALCulate:MEASure:CONVersion:FUNCtion | None |
<table>
<thead>
<tr>
<th>Format</th>
<th>Command</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log Mag</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Lin Mag</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Phase</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Delay</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Smith</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Polar</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>SWR</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Real</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Imaginary</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Unwrapped Phase</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Positive Phase</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Inverted Smith</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
<tr>
<td>Group Delay Aperture Points</td>
<td>CALCulate:MEASure:GDElay:POINts</td>
<td>Points</td>
</tr>
<tr>
<td>Group Delay Aperture Percent of Span</td>
<td>CALCulate:MEASure:GDElay:PERCent</td>
<td>Percent</td>
</tr>
<tr>
<td>Group Delay Aperture Frequency</td>
<td>CALCulate:MEASure:GDElay:FREQuency</td>
<td>Frequency</td>
</tr>
<tr>
<td>Set Preference to 2 points</td>
<td>SYSTem:PREFerences:ITEM:GDElay:TWOPoint</td>
<td>TwoPointGroupDelayAperture</td>
</tr>
</tbody>
</table>

8450
### Set or return the units for the specified data format

<table>
<thead>
<tr>
<th>Data Format</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temperature</td>
<td>CALCulate:MEASure:FORMat:UNIT</td>
<td>FormatUnit</td>
</tr>
<tr>
<td>Temperature</td>
<td>CALCulate:MEASure:FORMat</td>
<td>Format</td>
</tr>
</tbody>
</table>

### Scale

<table>
<thead>
<tr>
<th>Mode</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoScale</td>
<td>DISPlay:WINDow:TRACe:Y:AUTO</td>
<td>Autoscale</td>
</tr>
<tr>
<td>AutoScale All</td>
<td>DISPlay:WINDow:Y:AUTO</td>
<td>Autoscale</td>
</tr>
<tr>
<td>Per Division</td>
<td>DISPlay:WINDow:TRACe:Y:PDIVision</td>
<td>YScale</td>
</tr>
<tr>
<td>Reference Level</td>
<td>DISPlay:WINDow:TRACe:Y:RLEVEL</td>
<td>ReferenceValue</td>
</tr>
<tr>
<td>Reference Position</td>
<td>DISPlay:WINDow:TRACe:Y:RPOSITION</td>
<td>ReferencePosition</td>
</tr>
</tbody>
</table>

### Scale Coupling

<table>
<thead>
<tr>
<th>Mode</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set method</td>
<td>DISPlay:WINDow:TRACe:Y:COUPLE:METHod</td>
<td>ScaleCouplingMethod</td>
</tr>
<tr>
<td>Enable window</td>
<td>DISPlay:WINDow:TRACe:Y:COUPLE</td>
<td>ScaleCouplingState</td>
</tr>
</tbody>
</table>

### Electrical Delay

<table>
<thead>
<tr>
<th>Mode</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electrical Delay</td>
<td>CALCulate:MEASure:CORRection:EDELay:TIME</td>
<td>meas.ElectricalDelay</td>
</tr>
<tr>
<td>Delay in distance</td>
<td>CALCulate:MEASure:CORRection:EDELay:DISTance</td>
<td>ElecDistanceDelay</td>
</tr>
<tr>
<td>Set units for distance</td>
<td>CALCulate:MEASure:CORRection:EDELay:UNIT</td>
<td>ElecDistanceDelayUnit</td>
</tr>
<tr>
<td>Velocity Factor</td>
<td>SENSE:CORRection:RVELocity:COAX</td>
<td>PortVelocityFactor</td>
</tr>
<tr>
<td>Media</td>
<td>CALCulate:MEASure:CORRection:EDELay:MEDium</td>
<td>PortMedium</td>
</tr>
<tr>
<td>Wavegd Cutoff</td>
<td>CALCulate:MEASure:CORRection:EDELay:WGcutoff</td>
<td>PortWGcutoffFreq</td>
</tr>
</tbody>
</table>
## Constants

<table>
<thead>
<tr>
<th>Constants</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Z0</td>
<td>SENSe:CORRection:IMPedance:INPut:MAGNitude</td>
<td>SystemImpedanceZ0</td>
</tr>
<tr>
<td>Phase Offset</td>
<td>CALCulate:MEASure:OFFSet:PHAse</td>
<td>PhaseOffset</td>
</tr>
<tr>
<td>Mag Offset</td>
<td>CALCulate:MEASure:OFFSet:MAGNitude</td>
<td>MagnitudeOffset</td>
</tr>
<tr>
<td>Mag Slope</td>
<td>CALCulate:MEASure:OFFSet:MAGNitude:SOPe</td>
<td>MagnitudeSlopeOffset</td>
</tr>
</tbody>
</table>

## Math

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Trace ON</td>
<td>OFF</td>
<td>DISPlay:WINDow:TRACe[:STATe]</td>
</tr>
<tr>
<td>Memory Trace ON</td>
<td>OFF</td>
<td>DISPlay:WINDow:TRACe:MEMory</td>
</tr>
<tr>
<td>Data =&gt; Memory</td>
<td>CALCulate:MEASure:MATH:MEMorize</td>
<td>meas.DataToMemory</td>
</tr>
<tr>
<td>Memory data interpolation ON</td>
<td>OFF</td>
<td>CALCulate:MEASure:MATH:INTerpolate[:STATe]</td>
</tr>
<tr>
<td>Data Math (Add</td>
<td>Sub</td>
<td>Mult</td>
</tr>
</tbody>
</table>

## Equation Editor

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay data processing to end of sweep</td>
<td>CALCulate:MEASure:EQUation:FAST</td>
<td>FastProcessing</td>
</tr>
<tr>
<td>Turn ON / OFF equation</td>
<td>CALCulate:MEASure:EQUation[:STATe]</td>
<td>State</td>
</tr>
<tr>
<td>Set equation</td>
<td>CALCulate:MEASure:EQUation:TEXT</td>
<td>Text</td>
</tr>
<tr>
<td>Return validity of equation</td>
<td>CALCulate:MEASure:EQUation:VALid?</td>
<td>Valid</td>
</tr>
<tr>
<td>Returns the functions in DLL</td>
<td>CALCulate:EQUation:LIBRary:FUNCTIONs</td>
<td>GetLibraryFunctions</td>
</tr>
<tr>
<td>Imports the functions in DLL</td>
<td>CALCulate:EQUation:LIBRary:IMPort</td>
<td>ImportLibrary</td>
</tr>
<tr>
<td>Remove a DLL</td>
<td>CALCulate:EQUation:LIBRary:REMove</td>
<td>RemoveLibrary</td>
</tr>
</tbody>
</table>
### Statistics

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain Range Start</td>
<td>CALCulate:MEASure:FUNCTION:DOMain:USER:STARt</td>
<td>UserRangeMin</td>
</tr>
<tr>
<td>Set Type (Pk-Pk</td>
<td>StdDev</td>
<td>Mean)</td>
</tr>
<tr>
<td>Get Mean</td>
<td>CALCulate:MEASure:FUNCTION:DATA?</td>
<td>meas.Mean</td>
</tr>
<tr>
<td>Get Peak to Peak</td>
<td>CALCulate:MEASure:FUNCTION:DATA?</td>
<td>meas.PeakToPeak</td>
</tr>
<tr>
<td>Get formatted data array of multiple traces</td>
<td>CALCulate:DATA:MFData</td>
<td>None</td>
</tr>
<tr>
<td>Get corrected data array of multiple traces</td>
<td>CALCulate:DATA:MSData</td>
<td>None</td>
</tr>
<tr>
<td>Executes the statistical analysis</td>
<td>CALCulate:MEASure:FUNCTION:EXECute</td>
<td>None</td>
</tr>
</tbody>
</table>

### AM Distortion Commands

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and returns the compression level.</td>
<td>CALCulate:MEASure:DISTortion:BACKoff:COMPression</td>
<td>None</td>
</tr>
<tr>
<td>Enable/disable compression calculation.</td>
<td>CALCulate:MEASure:DISTortion:BACKoff[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Displays phase or amplitude distortion.</td>
<td>CALCulate:MEASure:DISTortion:MODE</td>
<td>None</td>
</tr>
<tr>
<td>Sets the aperture value over which the phase or gain slope will be calculated.</td>
<td>CALCulate:MEASure:DISTortion:SLOPe:APERture</td>
<td>None</td>
</tr>
<tr>
<td>Enables/disables phase slope (AMPM) or gain slope (AMAM) over the slope aperture to be displayed.</td>
<td>CALCulate:MEASure:DISTortion:SLOPe[:STATe]</td>
<td>None</td>
</tr>
</tbody>
</table>

### Trace Deviation
Calculates the deviation from a least-squares best fit line

<table>
<thead>
<tr>
<th>Trace Deviation Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCulate:MEASure:COMPutation:DEViation</td>
</tr>
<tr>
<td>TraceDeviationType</td>
</tr>
</tbody>
</table>

### Uncertainty Analysis

<table>
<thead>
<tr>
<th>Trace</th>
<th>CALCulate:MEASure:UNCertainty:DISPlay:CFACtor</th>
<th>CoverageFactor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trace Type</td>
<td>CALCulate:MEASure:UNCertainty:DISPLAY:TYPE</td>
<td>DisplayType</td>
</tr>
<tr>
<td>Repeatability</td>
<td>CALCulate:MEASure:UNCertainty:MODE:CABLE:REPeat</td>
<td>CableRepeatabilityUncertainty</td>
</tr>
<tr>
<td>Calibration</td>
<td>CALCulate:MEASure:UNCertainty:MODE:ETERm</td>
<td>ErrorTermUncertainty</td>
</tr>
<tr>
<td>Save uncertainty data</td>
<td>CALCulate:MEASure:UNCertainty:SAVE</td>
<td>WriteUncertaintyFile</td>
</tr>
</tbody>
</table>

### Limits

<p>| Display Lines ON|OFF | CALCulate:MEASure:LI Mi t:DISPlay[:STATe] | LineDisplay |
|----------------|-----------------------------------------------|--------------|
| Fail Sound ON|OFF | CALCulate:MEASure:LI Mi t:SOUNd[:STATe] | SoundOnFail |
| Testing ON|OFF | CALCulate:MEASure:LI Mi t[:STATe] | Trans.State |
| Limit Test Failed | CALCulate:MEASure:LI Mi t:FAIL? | meas.LimitTestFailed |
| Count Limit Lines | None | chans.Count |
| Read Test Results | GP-IB_Command_Finder/Status | limts.GetTestResult |
| Set / Read entire Limit Line | CALCulate:MEASure:LI Mi t:DATA | None |
| Limit Line Type (Max|Min) | CALCulate:MEASure:LI Mi t:SEGMen t:TYPE | limts.Type |</p>
<table>
<thead>
<tr>
<th>Begin Stimulus</th>
<th>CALCulate:MEASure:LI Mi t:SEGMen t:STIMulus:STARt</th>
<th>limtseg.BeginStimulus</th>
</tr>
</thead>
</table>
| Read the bandwidth test results for the active trace of selected channel. | CALCulate:MEASure:LIMit:REPort:ALL  
CALCulate:LIMit:REPort:ALL | None |
| Read the stimulus values at all the measurement points that failed the limit test for the active trace of selected channel. | CALCulate:MEASure:LIMit:REPort:DATA  
CALCulate:LIMit:REPort:DATA | None |
| Reads the number of the measurement points that failed the limit test, for the active trace of selected channel. | CALCulate:MEASure:LIMit:REPort:POINts  
CALCulate:LIMit:REPort:POINts | None |
| Delete all limit line data | CALCulate:MEASure:LIMit:DATA:DELeTe | None |
| Show Limit table | DISPlay:WINDow:TABLe | win.ShowTable |
| Global Pass/Fail | | |
| Show / hide the pass/fail dialog. | DISPlay:FSIGn | app.DisplayGlobalPassFail |
| Sets the policy used to compute the global pass/fail value. | CONTrol:HANDler:PASSfail:POLicy  
CONTrol:AUXiliary:PASSfail:POLicy | obj.PassFailPolicy |
| Reads the most recent pass/fail status value. | CONTrol:HANDler:PASSfail:STATus?  
| Sets the logic of the AuxIO PassFail line. | CONTrol:HANDler:PASSfail:LOGic  
CONTrol:AUXiliary:PASSfail:LOGic | obj.PassFailLogic |
| Sets the default logical pass/fail state. | CONTrol:HANDler:PASSfail:MODE  
CONTrol:AUXiliary:PASSfail:MODE | obj.PassFailMode |
| Sets the scope (Global or channel) of AuxIO pass/fail testing. | CONTrol:HANDler:PASSfail:SCOPe  
CONTrol:AUXiliary:PASSfail:SCOPe | obj.PassFailScope |

**Bandwidth Tests**
### Bandwidth Test

<table>
<thead>
<tr>
<th>Function Description</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set bandwidth threshold value of bandwidth test.</td>
<td>CALCulate:MEASure:BLIMit:BWIDth:THReshold</td>
<td>None</td>
</tr>
<tr>
<td>Turn ON/OFF the bandwidth value display of the bandwidth test.</td>
<td>CALCulate:MEASure:BLIMit:BWIDth:DISPlay:MARker:STATe</td>
<td>None</td>
</tr>
<tr>
<td>Get the bandwidth limit test results.</td>
<td>CALCulate:MEASure:BLIMit:FAIL</td>
<td>None</td>
</tr>
<tr>
<td>Set/get the upper limit value of the bandwidth test.</td>
<td>CALCulate:MEASure:BLIMit:MAXimum</td>
<td>None</td>
</tr>
<tr>
<td>Set/get the lower limit value of the bandwidth test.</td>
<td>CALCulate:MEASure:BLIMit:MINimum</td>
<td>None</td>
</tr>
<tr>
<td>Get the bandwidth value of the bandwidth test.</td>
<td>CALCulate:MEASure:BLIMit:REPort:DATA</td>
<td>None</td>
</tr>
<tr>
<td>Turn ON/OFF the bandwidth test function.</td>
<td>CALCulate:MEASure:BLIMit:STATe</td>
<td>None</td>
</tr>
</tbody>
</table>

### Ripple Tests

<table>
<thead>
<tr>
<th>Function Description</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set or return the ripple limit table</td>
<td>CALCulate:MEASure:RLIMit:DATA</td>
<td>None</td>
</tr>
<tr>
<td>Turn ON/OFF the ripple limit line display</td>
<td>CALCulate:MEASure:RLIMit:DISPlay::LINE:STATe</td>
<td>None</td>
</tr>
<tr>
<td>Set/get the ripple limit band</td>
<td>CALCulate:MEASure:RLIMit:DISPlay:SELect</td>
<td>None</td>
</tr>
<tr>
<td>Set/get the display type of ripple value</td>
<td>CALCulate:MEASure:RLIMit:DISPlay:TYPE</td>
<td>None</td>
</tr>
<tr>
<td>Read the ripple test result</td>
<td>CALCulate:MEASure:RLIMit:FAIL</td>
<td>None</td>
</tr>
<tr>
<td>Read the ripple value</td>
<td>CALCulate:MEASure:RLIMit:REPort:DATA</td>
<td>None</td>
</tr>
<tr>
<td>Turn ON/OFF the ripple test function</td>
<td>CALCulate:MEASure:RLIMit:STATe</td>
<td>None</td>
</tr>
</tbody>
</table>

### Transform

<table>
<thead>
<tr>
<th>Function Description</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets the alignment of the time domain measurement.</td>
<td>CALCulate:MEASure:TRANsform:TIME:ALIgnment</td>
<td>Alignment</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td><strong>Start Time</strong></td>
<td>CALCulate:MEASure:TRANsform:TIME:STARt</td>
<td>trans.Start</td>
</tr>
<tr>
<td><strong>Stop Time</strong></td>
<td>CALCulate:MEASure:TRANsform:TIME:STOP</td>
<td>trans.Stop</td>
</tr>
<tr>
<td><strong>Center</strong></td>
<td>CALCulate:MEASure:TRANsform:TIME:CENTer</td>
<td>trans.Center</td>
</tr>
<tr>
<td><strong>Span</strong></td>
<td>CALCulate:MEASure:TRANsform:TIME:SPAN</td>
<td>trans.Span</td>
</tr>
<tr>
<td><strong>Set Low Pass Frequency</strong></td>
<td>CALCulate:MEASure:TRANsform:TIME:LPFREQuency</td>
<td>trans.SetFrequencyLowPass</td>
</tr>
<tr>
<td><strong>Set/get the impulse width</strong></td>
<td>CALCulate:MEASure:TRANsform:TIME:IMPulse:WIDTh</td>
<td>trans.ImpulseWidth</td>
</tr>
<tr>
<td><strong>TD Toolbar</strong></td>
<td>DISPLAY:TOOLbar:TRANsform[:STATe]</td>
<td>app.ShowToolbar</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Gating</strong></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>**ON</td>
<td>OFF**</td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME:STATe</td>
</tr>
<tr>
<td><strong>Type (BandPass, Notch)</strong></td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME[:TYPE]</td>
<td>gate.Type</td>
</tr>
<tr>
<td><strong>Shape</strong></td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME:SHAPe</td>
<td>gate.Shape</td>
</tr>
<tr>
<td><strong>Start</strong></td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME:STARt</td>
<td>gate.Start</td>
</tr>
<tr>
<td><strong>Stop</strong></td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME:STOP</td>
<td>gate.Stop</td>
</tr>
<tr>
<td><strong>Center</strong></td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME:CENTer</td>
<td>gate.Center</td>
</tr>
<tr>
<td><strong>Span</strong></td>
<td>CALCulate:MEASure:FILTER[:GATE]:TIME:SPAN</td>
<td>gate.Span</td>
</tr>
<tr>
<td><strong>Set gate coupling parameters</strong></td>
<td>CALCulate:MEASure:FILTER[:GATE]:COUPle:PARameters</td>
<td>gate.CoupledParameters</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Window</strong></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Impulse Width</strong></td>
<td>CALCulate:MEASure:TRANsform:TIME:IMPulse:WIDTh</td>
<td>trans.ImpulseWidth</td>
</tr>
</tbody>
</table>
### Coupling

<table>
<thead>
<tr>
<th>Enable trace coupling</th>
<th>SENSe:COUPle:PARameter[:STATe]</th>
<th>CoupleChannelParams</th>
</tr>
</thead>
</table>

### Distance Markers

<table>
<thead>
<tr>
<th>Specify measurement type for distance markers</th>
<th>CALCulate:MEASure:TRANsform:TIME:MARKer:MODE</th>
<th>trans.DistanceMarkerMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify units for distance markers</td>
<td>CALCulate:MEASure:TRANsform:TIME:MARKer:UNIT</td>
<td>trans.DistanceMarkerUnit</td>
</tr>
<tr>
<td>Set and return marker distance value</td>
<td>CALCulate:MEASure:MARKer:DISTance</td>
<td>mark.Distance</td>
</tr>
</tbody>
</table>

### Averaging

<table>
<thead>
<tr>
<th>Average ON</th>
<th>OFF</th>
<th>SENSe:AVERage[:STATe]</th>
<th>chan.Averaging</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Factor</td>
<td>SENSe:AVERage:COUNt</td>
<td>chan.AveragingFactor</td>
<td></td>
</tr>
<tr>
<td>Return the Average Count</td>
<td>None (STATus:OPERation:AVERaging)</td>
<td>chan.AveragingCount</td>
<td></td>
</tr>
<tr>
<td>Average Type</td>
<td>SENSe:AVERage:MODE</td>
<td>AverageMode</td>
<td></td>
</tr>
<tr>
<td>Average Restart</td>
<td>SENSe:AVERage:CLEar</td>
<td>chan.AveragingRestart</td>
<td></td>
</tr>
</tbody>
</table>

### IF Bandwidth

<table>
<thead>
<tr>
<th>IF Bandwidth</th>
<th>SENSe:BANDwidth</th>
<th>chan.IFBandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous IF Bandwidth</td>
<td>None</td>
<td>chan.Previous_IFBandwidth</td>
</tr>
<tr>
<td>Next IF Bandwidth</td>
<td>None</td>
<td>chan.Next_IFBandwidth</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>Reduce IF BW</td>
<td>SENSE:BANDwidth</td>
<td>BWIDth:TRACk</td>
</tr>
<tr>
<td>Smoothing</td>
<td>CALCulate:MEASure:SMOothing[:STATe]</td>
<td>meas.Smoothing</td>
</tr>
<tr>
<td>Smoothing ON/OFF</td>
<td>CALCulate:MEASure:SMOothing:APERture</td>
<td>meas.SmoothingAperture</td>
</tr>
<tr>
<td>Smoothing Points</td>
<td>CALCulate:MEASure:SMOothing:POINts</td>
<td>None</td>
</tr>
<tr>
<td>Marker Functions</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>meas.MarkerState</td>
</tr>
<tr>
<td>ON/OFF</td>
<td>CALCulate:MEASure:MARKer:AOFF</td>
<td>meas.DeleteAllMarkers</td>
</tr>
<tr>
<td>Delete All Markers</td>
<td>CALCulate:MEASure:MARKer[:STATe]</td>
<td>meas.DeleteMarker</td>
</tr>
<tr>
<td>Delete Marker</td>
<td>CALCulate:MEASure:MARKer:DELTe</td>
<td>mark.DeltaMarker</td>
</tr>
<tr>
<td>Viewing Marker readouts</td>
<td>Display</td>
<td>Display</td>
</tr>
<tr>
<td>Get a handle to Ref marker</td>
<td>None</td>
<td>meas.GetReferenceMarker</td>
</tr>
<tr>
<td>Advanced Settings</td>
<td>CALCulate:MEASure:MARKer:DISCrete</td>
<td>mark.Interpolated</td>
</tr>
<tr>
<td>Interpolate All Markers</td>
<td>None</td>
<td>meas.Interpolate</td>
</tr>
<tr>
<td>Interpolate Individ. Marker</td>
<td>CALCulate:MEASure:MARKer:DISCrete</td>
<td>mark.Interpolated</td>
</tr>
<tr>
<td>Type (Normal</td>
<td>Fixed)</td>
<td>CALCulate:MEASure:MARKer:TYPE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>meas.MarkerFormat</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Coupled Markers Method</td>
<td>CALCulate:MEASure:MARKer:COUPling:METHOD</td>
<td>CoupledMarkersMethod</td>
</tr>
<tr>
<td>Read/Set Data Point number</td>
<td>CALCulate:MEASure:MARKer:BUCKET</td>
<td>mark.BucketNumber</td>
</tr>
<tr>
<td>Read/Set X-axis value</td>
<td>CALCulate:MEASure:MARKer:X</td>
<td>mark.Stimulus</td>
</tr>
<tr>
<td>Read/Set Y-axis value</td>
<td>CALCulate:MEASure:MARKer:Y</td>
<td>mark.Value</td>
</tr>
<tr>
<td>Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Marker=&gt; SA</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>toSA</td>
</tr>
<tr>
<td>Marker=&gt; Span</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>None</td>
</tr>
<tr>
<td>Marker=&gt; Center (Freq)</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>mark.SetCenter</td>
</tr>
<tr>
<td>Marker=&gt; CW Freq and change sweep type</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>mark.SetCW</td>
</tr>
<tr>
<td>Marker=&gt; Start (Freq)</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>mark.SetStart</td>
</tr>
<tr>
<td>Marker=&gt; Stop (Freq)</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>mark.SetStop</td>
</tr>
<tr>
<td>Marker=&gt; Elect. Delay</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>mark.SetElectricalDelay</td>
</tr>
<tr>
<td>Marker=&gt; Ref. Level</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>mark.SetReferenceLevel</td>
</tr>
<tr>
<td>Marker=&gt; CW Freq - No sweep type change</td>
<td>CALCulate:MEASure:MARKer:SET</td>
<td>SetCWFreq</td>
</tr>
</tbody>
</table>

**SA Band Marker Settings**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Read Band Power Span</td>
<td>CALCulate:MEASure:SA:MARKer:BPOWER:SPAN</td>
<td>BandpowerSpan</td>
</tr>
<tr>
<td>Read Band Noise Span</td>
<td>CALCulate:MEASure:SA:MARKer:BNOise:SPAN</td>
<td>BandnoiseSpan</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><strong>Compression Marker Search</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Compression Marker level found.</strong></td>
<td>CALCulate:MEASure:MARKer:FUNCtion:COMPression:LEVel</td>
<td>CompressionLevel</td>
</tr>
<tr>
<td><strong>Read Compression Marker</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input power</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:COMPression:PIN</td>
<td>CompressionPin</td>
</tr>
<tr>
<td><strong>Read Compression Marker</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output power</td>
<td>CALCulate:MEASure:MARKer:FUNCtion:COMPression:POUT</td>
<td>CompressionPout</td>
</tr>
<tr>
<td><strong>Search function</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:MEASure:MARKer:FUNCtion[:SELet]</td>
<td>SearchCompressionPoint</td>
</tr>
<tr>
<td>**Turn ON</td>
<td>OFF the compression state**</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:MEASure:MARKer:FUNCtion:COMPression[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td><strong>Execute function</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CALCulate:MEASure:MARKer:FUNCtion:EXECute</td>
<td>SearchCompressionPoint</td>
</tr>
<tr>
<td><strong>PSAT Marker Search</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Initiate a PSAT search</strong></td>
<td>CALCulate:MEASure:MARKer:PSATuration:BACKoff</td>
<td>SearchPowerSaturation</td>
</tr>
<tr>
<td><strong>Set and read PSAT backoff</strong></td>
<td>CALCulate:MEASure:MARKer:PSATuration:BACKoff</td>
<td>PMaxBackOff</td>
</tr>
<tr>
<td><strong>Read PSat Out</strong></td>
<td>CALCulate:MEASure:MARKer:PSATuration:POUT?</td>
<td>POut</td>
</tr>
<tr>
<td><strong>Read PSat In</strong></td>
<td>CALCulate:MEASure:MARKer:PSATuration:PIN?</td>
<td>Pin</td>
</tr>
<tr>
<td><strong>Read PMax Out</strong></td>
<td>CALCulate:MEASure:MARKer:PSATuration:POUT:MAXimum?</td>
<td>PMaxOut</td>
</tr>
<tr>
<td><strong>Read PMax In</strong></td>
<td>CALCulate:MEASure:MARKer:PSATuration:PIN:MAXimum?</td>
<td>PMaxIn</td>
</tr>
<tr>
<td><strong>Read Gain Sat</strong></td>
<td>CALCulate:MEASure:MARKer:PSATuration:GAIN?</td>
<td>GainSaturation</td>
</tr>
<tr>
<td><strong>Read Gain Max</strong></td>
<td>CALCulate:MEASure:MARKer:PSATuration:GAIN:MAXimum?</td>
<td>GainMax</td>
</tr>
<tr>
<td><strong>Read Gain Linear</strong></td>
<td>CALCulate:MEASure:MARKer:PSATuration:GAIN:LINear?</td>
<td>GainLinear</td>
</tr>
<tr>
<td><strong>Read Comp Sat</strong></td>
<td>CALCulate:MEASure:MARKer:PSATuration:COMPression:SAUation?</td>
<td>CompressionSaturation</td>
</tr>
<tr>
<td>Function</td>
<td>Command</td>
<td>Value</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Read Comp Max</td>
<td>CALCulate:MEASure:MARKer:PSATuration:COMPression:MAXimum?</td>
<td>CompressionMax</td>
</tr>
<tr>
<td>Turn ON</td>
<td>OFF PSAT marker search</td>
<td>CALCulate:MEASure:MARKer:PSATuration[:STATe]</td>
</tr>
</tbody>
</table>

PNOP Marker Search

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiate a PNOP search</td>
<td>CALCulate:MEASure:MARKer:PNOP:BACKoff</td>
<td>SearchPowerNormalOperatingPoint</td>
</tr>
<tr>
<td>PNOP backoff</td>
<td>CALCulate:MEASure:MARKer:PNOP:BACKoff</td>
<td>BackOff</td>
</tr>
<tr>
<td>PNOP Power Offset</td>
<td>CALCulate:MEASure:MARKer:PNOP:POFFset</td>
<td>PinOffset</td>
</tr>
<tr>
<td>Read Pnop Out</td>
<td>CALCulate:MEASure:MARKer:PNOP:POUT?</td>
<td>POut</td>
</tr>
<tr>
<td>Read Pnop in</td>
<td>CALCulate:MEASure:MARKer:PNOP:PIN?</td>
<td>Pin</td>
</tr>
<tr>
<td>Read Pnop Gain</td>
<td>CALCulate:MEASure:MARKer:PNOP:GAIN?</td>
<td>Gain</td>
</tr>
<tr>
<td>Read Pnop Comp</td>
<td>CALCulate:MEASure:MARKer:PNOP:COMPression?</td>
<td>Compression</td>
</tr>
<tr>
<td>Read PMax Out</td>
<td>CALCulate:MEASure:MARKer:PNOP:POUT:MAXimum?</td>
<td>PMaxOut</td>
</tr>
<tr>
<td>Read PMax In</td>
<td>CALCulate:MEASure:MARKer:PNOP:PIN:MAXimum?</td>
<td>PMaxIn</td>
</tr>
<tr>
<td>Read Comp Max</td>
<td>CALCulate:MEASure:MARKer:PNOP:COMPression:MAXimum?</td>
<td>CompressionMax</td>
</tr>
<tr>
<td>Read PBO in</td>
<td>CALCulate:MEASure:MARKer:PNOP:BACKoff:PIN?</td>
<td>BackOffPIn</td>
</tr>
<tr>
<td>Turn ON</td>
<td>OFF PNOP marker search</td>
<td>CALCulate:MEASure:MARKer:PNOP[:STATe]</td>
</tr>
</tbody>
</table>

Basic Marker Search Functions
<table>
<thead>
<tr>
<th>Execute Search</th>
<th>CALCulate:MEASure:MARKer:FUNCTION:EXECute</th>
<th>SearchFunction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select Search Function</td>
<td>CALCulate:MEASure:MARKer:FUNCTION[:SELect]</td>
<td>select and execute each Search...</td>
</tr>
<tr>
<td>Maximum</td>
<td>CALCulate:MEASure:MARKer:FUNCTION[:SELect]</td>
<td>mark.SearchMax</td>
</tr>
<tr>
<td>Minimum</td>
<td>CALCulate:MEASure:MARKer:FUNCTION[:SELect]</td>
<td>mark.SearchMin</td>
</tr>
<tr>
<td>Target (Value)</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:TARGet[:VALue]</td>
<td>mark.TargetValue</td>
</tr>
<tr>
<td>Select transition type</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:TARGet:TRANsition</td>
<td>None</td>
</tr>
<tr>
<td>Excursion Value</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:PEAK:EXCursion</td>
<td>mark.PeakExcursion</td>
</tr>
<tr>
<td>Threshold Value</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:PEAK:THReshold</td>
<td>mark.PeakThreshold</td>
</tr>
<tr>
<td>Set or return polarity</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:PEAK:POLarity</td>
<td>None</td>
</tr>
<tr>
<td>of the peak search</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tracking</td>
<td>CALCulate:MEASure:MARKer:FUNCTION:TRACKing</td>
<td>mark.Tracking</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bandwidth &amp; Notch Marker Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bandwidth (Target)</td>
</tr>
<tr>
<td>Read Filter Center Freq</td>
</tr>
<tr>
<td>Set bandwidth marker function</td>
</tr>
<tr>
<td>reference to either MARKer or PEAK</td>
</tr>
<tr>
<td>Feature</td>
</tr>
<tr>
<td>------------------------------</td>
</tr>
<tr>
<td>Read notch search result</td>
</tr>
<tr>
<td>Notch Search</td>
</tr>
<tr>
<td>Notch Ref To</td>
</tr>
<tr>
<td>Notch Level</td>
</tr>
</tbody>
</table>

### Multi Peak & Target Marker Search

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi Peak Search</td>
<td><code>CALCulate:MEASure:MARKer:FUNCTION:MULTi:EXECute</code></td>
<td>None</td>
</tr>
<tr>
<td>Peak Threshold</td>
<td><code>CALCulate:MEASure:MARKer:FUNCTION:MULTi:PEAK:THReshold</code></td>
<td>None</td>
</tr>
<tr>
<td>Peak Excursion</td>
<td><code>CALCulate:MEASure:MARKer:FUNCTION:MULTi:PEAK:EXCursion</code></td>
<td>None</td>
</tr>
<tr>
<td>Peak Polarity</td>
<td><code>CALCulate:MEASure:MARKer:FUNCTION:MULTi:PEAK:POLarity</code></td>
<td>None</td>
</tr>
<tr>
<td>Multi Target Search</td>
<td><code>CALCulate:MEASure:MARKer:FUNCTION:MULTi:EXECute</code></td>
<td>None</td>
</tr>
<tr>
<td>Target Value</td>
<td><code>CALCulate:MEASure:MARKer:FUNCTION:MULTi:TARGet[:VALue]</code></td>
<td>None</td>
</tr>
<tr>
<td>Set or return search type of the multi search</td>
<td><code>CALCulate:MEASure:MARKer:FUNCTION:MULTi:SELeCT</code></td>
<td>None</td>
</tr>
<tr>
<td>Turn ON</td>
<td>OFF search tracking</td>
<td><code>CALCulate:MEASure:MARKer:FUNCTION:MULTi:TRACking</code></td>
</tr>
<tr>
<td>Transition</td>
<td><code>CALCulate:MEASure:MARKer:FUNCTION:MULTi:TARGet:TRANSition</code></td>
<td>None</td>
</tr>
</tbody>
</table>

### Status Bar

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status Bar On</td>
<td>Off</td>
<td><code>DISPlay:ANNotation[:STATus]</code></td>
</tr>
</tbody>
</table>

### Toolbars/ Title Bars

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toolbars On</td>
<td>Off</td>
<td><code>DISPlay:TOOLbar:ENTRY</code></td>
</tr>
<tr>
<td></td>
<td><code>DISPlay:TOOLbar:MARKer</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>DISPlay:TOOLbar:KEYS[:STATe]</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>DISPlay:TOOLbar:EXT[:STATe]</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>DISPlay:TOOLbar:TRAN[:STATe]</code></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Setting</td>
<td>Command</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Title Bars On</td>
<td>Off</td>
<td>None</td>
</tr>
<tr>
<td>Tables</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tables On</td>
<td>Off</td>
<td>DISPlay:WINDow:TABLEe</td>
</tr>
<tr>
<td>Data/Memory Trace</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory Trace On</td>
<td>Off</td>
<td>DISPlay:WINDow:TRACe:MEMory</td>
</tr>
<tr>
<td>Marker (readout) Display</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Marker Readout Size</td>
<td>DISPlay:WINDow:ANNotation:MARKer:SIZE</td>
<td>win.MarkerReadoutSize</td>
</tr>
<tr>
<td>Readouts Per Trace</td>
<td>DISPlay:WINDow:ANNotation:MARKer:NUMBer</td>
<td>MarkerReadoutsPerTrace</td>
</tr>
<tr>
<td>Stimulus decimal places</td>
<td>DISPlay:WINDow:ANNotation:MARKer:RESolution:STIMulus</td>
<td>MarkerReadoutStimulusPlaces</td>
</tr>
<tr>
<td>Response decimal places</td>
<td>DISPlay:WINDow:ANNotation:MARKer:RESolution:RESPonse</td>
<td>MarkerReadoutResponsePlaces</td>
</tr>
<tr>
<td>Readout position: X-axis</td>
<td>DISPlay:WINDow:ANNotation:MARKer:XPOSition</td>
<td>MarkerReadoutXPosition</td>
</tr>
<tr>
<td>Marker symbol</td>
<td>DISPlay:WINDow:ANNotation:MARKer:SYMBol</td>
<td>MarkerSymbol</td>
</tr>
</tbody>
</table>
### Frequency | Offset | Power | ALC | Recvr Lvl | Source Ports | # Points | Trigger : Ext | Auxiliary Trigger
Sweep settings:.. | Time | Setup | Segment | Power | Phase | Pulse Setup | DC Sources | Multi-Dimensional Sweep | Global Source

See [Remotely Specifying a Source Port](#).

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Frequency</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start Freq</td>
<td>SENSE:FREQuency:STARt</td>
<td>chan.StartFrequency</td>
</tr>
<tr>
<td>Stop Freq</td>
<td>SENSE:FREQuency:STOP</td>
<td>chan.StopFrequency</td>
</tr>
<tr>
<td>Center Freq</td>
<td>SENSE:FREQuency:CENTer</td>
<td>chan.CenterFrequency</td>
</tr>
<tr>
<td>Span</td>
<td>SENSE:FREQuency:SPAN</td>
<td>chan.FrequencySpan</td>
</tr>
<tr>
<td>CW Frequency</td>
<td>SENSE:FREQuency:CW</td>
<td>chan.CWFrequency</td>
</tr>
<tr>
<td>Number of Points</td>
<td>SENSE:SWEep:POINts</td>
<td>chan.NumberOfPoints</td>
</tr>
<tr>
<td><strong>Step size</strong></td>
<td>SENSE:SWEep:STEP</td>
<td>FrequencyStep Property</td>
</tr>
</tbody>
</table>

### Frequency Offset Commands

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freq Offset ON/Off</td>
<td>SENSE:FOM[:STATe]</td>
<td>FOM.State</td>
</tr>
<tr>
<td>Read available ranges</td>
<td>SENSE:FOM:CATalog?</td>
<td></td>
</tr>
<tr>
<td>Read number of ranges</td>
<td>SENSE:FOM:COUNT?</td>
<td>RangeCount</td>
</tr>
<tr>
<td>X-Axis display range</td>
<td>SENSE:FOM:DISPlay:SELect</td>
<td>DisplayRange</td>
</tr>
<tr>
<td>Read range name</td>
<td>SENSE:FOM:RANGe:NAME?</td>
<td>Name FOMRange</td>
</tr>
<tr>
<td>Read range number</td>
<td>SENSE:FOM:RNUM?</td>
<td>rangeNumber</td>
</tr>
<tr>
<td>Set range coupling</td>
<td>SENSE:FOM:RANGe:COUplled</td>
<td>Coupled</td>
</tr>
<tr>
<td>Set sweep type</td>
<td>SENSE:FOM:RANGe:SWEep:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Set CW freq</td>
<td>SENSE:FOM:RANGe:FREQuency:CW</td>
<td>CWFrequency</td>
</tr>
<tr>
<td>Set start freq</td>
<td>SENSE:FOM:RANGE: FREQUENCY: START</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Set stop freq</td>
<td>SENSE:FOM:RANGE: FREQUENCY: STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>Set offset value</td>
<td>SENSE:FOM:RANGE: FREQUENCY: OFFSET</td>
<td>Offset</td>
</tr>
<tr>
<td>Set divisor value</td>
<td>SENSE:FOM:RANGE: FREQUENCY: DIVISOR</td>
<td>Divisor</td>
</tr>
<tr>
<td>Set multiplier value</td>
<td>SENSE:FOM:RANGE: FREQUENCY: MULTIPLIER</td>
<td>Multiplier</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Freq. Offset Segment Sweep</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON/OFF</td>
</tr>
<tr>
<td>Add a segment</td>
</tr>
<tr>
<td>Delete a segment</td>
</tr>
<tr>
<td>Count the segments</td>
</tr>
<tr>
<td>Center Frequency</td>
</tr>
<tr>
<td>Frequency Span</td>
</tr>
<tr>
<td>Start Frequency</td>
</tr>
<tr>
<td>Stop Frequency</td>
</tr>
<tr>
<td>Number of Points</td>
</tr>
<tr>
<td>IF Bandwidth value</td>
</tr>
<tr>
<td>IF Bandwidth control</td>
</tr>
<tr>
<td>Source Power value</td>
</tr>
<tr>
<td>Sweep time value</td>
</tr>
<tr>
<td>Sweep time control</td>
</tr>
</tbody>
</table>

| Test Set Switch | ROUT: PATH: LOOP: R1 | chan.R1InputPath |

Power Settings   See Remotely Specifying a Source Port.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power ON</td>
<td>OUTP</td>
<td>app.SourcePowerState</td>
</tr>
<tr>
<td>Source Power (Auto</td>
<td>ON</td>
<td>OFF)</td>
</tr>
<tr>
<td>Power Value</td>
<td>SOURce:POWer[:LEVel][:IMMediate][:AMPLitude]</td>
<td>chan.TestPortPower</td>
</tr>
<tr>
<td>Port Selection</td>
<td>SENSE:SWepE:SRCPort</td>
<td>chan.TestPortPower</td>
</tr>
<tr>
<td>Couple Ports OFF</td>
<td>ON</td>
<td>SOURce:POWer:COUPle</td>
</tr>
<tr>
<td>Attenuation Value</td>
<td>SOURce:POWer:ATTenuation</td>
<td>chan.Attenuator</td>
</tr>
<tr>
<td>Power Slope ON</td>
<td>OFF</td>
<td>SOURce:POWer:SLOPe:STATE</td>
</tr>
<tr>
<td>Power Slope Value</td>
<td>SOURce:POWer:SLOPe</td>
<td>chan.PowerSlope</td>
</tr>
<tr>
<td>Receiver Attenuation</td>
<td>SENSE:POWer:ATTenuator</td>
<td>chan.ReceiverAttenuator</td>
</tr>
<tr>
<td>Receiver Reference Attenuation</td>
<td>SOURce:POWer:ATTenuation:RECeiver:REFERENCE</td>
<td>None</td>
</tr>
<tr>
<td>Receiver Test Attenuation</td>
<td>SOURce:POWer:ATTenuation:RECeiver:TEST</td>
<td>None</td>
</tr>
<tr>
<td>Shutdown or Restart System</td>
<td>SYSTem:POFF</td>
<td>None</td>
</tr>
</tbody>
</table>

See also Power Range remote commands

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

### Power Limit and Offsets

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set power limit</td>
<td>SYSTem:POWer:LIMit</td>
<td>Limit</td>
</tr>
<tr>
<td>Power limit ON/OFF</td>
<td>SYSTem:POWer:LIMit:STATe</td>
<td>State</td>
</tr>
<tr>
<td>Power limit UI lock</td>
<td>SYSTem:POWer:LIMit:LOCK</td>
<td>Lock</td>
</tr>
<tr>
<td>Set offset value</td>
<td>SOURce:POWer:ALC:MODE:RECeiver:OFFSet</td>
<td>PowerOffset</td>
</tr>
</tbody>
</table>

### Receiver Leveling

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rx Level ON/OFF</td>
<td>SOURce:POWer:ALC:MODE:RECeiver</td>
<td>State</td>
</tr>
<tr>
<td>Select the reference receiver</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:REFerence</code></td>
<td>ReferenceReceiver</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Sets all ports to pre-sweep or point leveling mode for the specified channel.</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:ACQuisition:MODE</code></td>
<td>None</td>
</tr>
<tr>
<td>Enables or disables the receiver leveling maximum iteration search function.</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:ITERation:ENABLE</code></td>
<td>None</td>
</tr>
<tr>
<td>Set maximum iterations</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:ITERation:VALue</code></td>
<td>IterationNumber</td>
</tr>
<tr>
<td>Set tolerance</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:TOLerance</code></td>
<td>Tolerance</td>
</tr>
<tr>
<td>Set offset value</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:OFFSet</code></td>
<td>PowerOffset</td>
</tr>
<tr>
<td>Separate IFBW</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:FAST</code></td>
<td>FastMode</td>
</tr>
<tr>
<td>Set Rx IFBW</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:IFBW</code></td>
<td>LevelingIFBW</td>
</tr>
<tr>
<td>Safe mode ON</td>
<td>OFF</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:SAFE</code></td>
</tr>
<tr>
<td>Safe mode Max power</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:SAFE:MAX</code></td>
<td>PowerMax</td>
</tr>
<tr>
<td>Safe mode Min power</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:SAFE:MIN</code></td>
<td>PowerMin</td>
</tr>
<tr>
<td>Use Last Result for Source Power Cal</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:LSPC</code></td>
<td>LastLevelingAsSPC</td>
</tr>
<tr>
<td>Sets the specific PULSe4 behavior.</td>
<td><code>SENSe:PULSe4:MODE</code></td>
<td>None</td>
</tr>
<tr>
<td>Enable the receiver leveling maximum iteration search function.</td>
<td><code>SOURce:POWer:ALC[:MODE]:RECeiver:ITERation:ENABLE</code></td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ALC Leveling</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns list of valid ALC Leveling Modes</td>
<td><code>SOURce:POWer:ALC[:MODE]:CATalog?</code></td>
</tr>
<tr>
<td>Set ALC Mode</td>
<td><code>SOURce:POWer:ALC[:MODE]</code></td>
</tr>
</tbody>
</table>
### Specifying Source Ports

See **Remotely Specifying a Source Port.**

- **Returns the number of source ports.**
  - None
  - SourcePortCount

- **Returns the string names of source ports.**
  - SOURce:CATalog?
  - SourcePortNames

- **Returns the source port number of the specified string port name.**
  - None
  - GetPortNumber

### IF Bandwidth

<table>
<thead>
<tr>
<th>IF Bandwidth</th>
<th>Command</th>
<th>Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF Bandwidth</td>
<td>SENS:e:BANDwidth</td>
<td>chan.IFBandwidth</td>
</tr>
<tr>
<td>Previous IF Bandwidth</td>
<td>None</td>
<td>chan.Previous_IFBandwidth</td>
</tr>
<tr>
<td>Next IFBandwidth</td>
<td>None</td>
<td>chan.Next_IFBandwidth</td>
</tr>
<tr>
<td>Reduce IF BW</td>
<td>SENS:e:BANDwidth</td>
<td>chan.ReduceIFBW</td>
</tr>
</tbody>
</table>

### Sweep

<table>
<thead>
<tr>
<th>Sweep Time Value</th>
<th>Command</th>
<th>Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the time the first point of a Time Sweep is measured.</td>
<td>SENS:e:SWEep:TIME:AUTO</td>
<td>chan.SweepTime</td>
</tr>
<tr>
<td>Sets the time the analyzer takes to complete one sweep.</td>
<td>SENS:e:SWEep:TIME[:STOP]</td>
<td>None</td>
</tr>
</tbody>
</table>

### Sweep Setup

<table>
<thead>
<tr>
<th>Number of Points</th>
<th>Command</th>
<th>Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sweep Type (Lin</td>
<td>Pwr</td>
<td>CW</td>
</tr>
<tr>
<td>Sweep Generation (Stepped</td>
<td>Analog)</td>
<td>SENS:e:SWEep:GENeration</td>
</tr>
<tr>
<td>Dwell Time Value</td>
<td>SENS:e:SWEep:DWEL</td>
<td>chan.DwellTime</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>--------------------------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>Dwell Time Auto set the minimum dwell time</td>
<td><code>SENSe:SWEep:DWEL:AUTO</code></td>
<td>chan.DwellTime</td>
</tr>
<tr>
<td>Sweep Delay</td>
<td><code>SENSe:SWEep:DWEL:SDELay</code></td>
<td>SweepDelay</td>
</tr>
<tr>
<td>Alternate Sweeps</td>
<td><code>SENSe:COUPle</code></td>
<td>chan.AlternateSweep</td>
</tr>
<tr>
<td>External ALC</td>
<td><code>SOURce:POWer:DETector</code></td>
<td>app.ExternalALC</td>
</tr>
<tr>
<td>Enable Point Sweep</td>
<td><code>SENSe:SWEep:GENeration:POINTsweep</code></td>
<td>PointSweepState</td>
</tr>
<tr>
<td>Fast Sweep</td>
<td><code>SENSe:SWEep:SPEed</code></td>
<td>SweepSpeedMode</td>
</tr>
<tr>
<td>Fast CW</td>
<td><code>SENSe:SWEep:TYPE:FACW</code></td>
<td>FastCWPointCount</td>
</tr>
<tr>
<td>Set shift LO maximum frequency</td>
<td><code>SENSe:SWEep:SLOCal:MAXimum</code></td>
<td>None</td>
</tr>
<tr>
<td>Turn shift LO on or off</td>
<td><code>SENSe:SWEep:SLOCal:STATe</code></td>
<td>None</td>
</tr>
<tr>
<td>Returns whether or not the VNA has the low frequency extension (LFE) installed.</td>
<td><code>SYSTem:CAPability:HARDware:LFEXtension:EXISts?</code></td>
<td>HasLowFrequencyExtension</td>
</tr>
<tr>
<td>Turns ON or OFF low frequency extension</td>
<td><code>SENSe:SWEep:LFEXtension:STATe</code></td>
<td>LowFrequencyExtension</td>
</tr>
</tbody>
</table>

### Power Sweep

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Power</td>
<td><code>SOURce:POWer:START</code></td>
<td>chan.StartPower</td>
</tr>
<tr>
<td>Stop Power</td>
<td><code>SOURce:POWer:STOP</code></td>
<td>chan.StopPower</td>
</tr>
<tr>
<td>Center</td>
<td><code>SOURce:POWer:CENTer</code></td>
<td></td>
</tr>
<tr>
<td>Span</td>
<td><code>SOURce:POWer:SPAN</code></td>
<td></td>
</tr>
</tbody>
</table>

### Segment Sweep

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>OFF</td>
<td><code>SENSe:SEGment[:STATe]</code></td>
</tr>
<tr>
<td>Add a segment</td>
<td><code>SENSe:SEGment:ADD</code></td>
<td>Segs.Add</td>
</tr>
<tr>
<td><strong>Delete a segment</strong></td>
<td><code>SENSe:SEGMenT:DELete</code></td>
<td><code>segments.Remove</code></td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td><strong>Delete all segments</strong></td>
<td><code>SENSe:SEGMenT:DELete:ALL</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td><strong>Count the segments</strong></td>
<td><code>SENSe:SEGMenT:COUNt</code></td>
<td><code>chans.Count</code></td>
</tr>
<tr>
<td><strong>Read the segment number</strong></td>
<td><code>None</code></td>
<td><code>seg.SegmentNumber</code></td>
</tr>
<tr>
<td><strong>Segment Center Frequency</strong></td>
<td><code>SENSe:SEGMenT:FREQuency:CENTer</code></td>
<td><code>chan.centerFrequency</code></td>
</tr>
<tr>
<td><strong>Segment Frequency Span</strong></td>
<td><code>SENSe:SEGMenT:FREQuency:SPAN</code></td>
<td><code>chan.FrequencySpan</code></td>
</tr>
<tr>
<td><strong>Segment Start Frequency</strong></td>
<td><code>SENSe:SEGMenT:FREQuency:STARt</code></td>
<td><code>Chan.StartFrequency</code></td>
</tr>
<tr>
<td><strong>Segment Stop Frequency</strong></td>
<td><code>SENSe:SEGMenT:FREQuency:STOP</code></td>
<td><code>Chan.StopFrequency</code></td>
</tr>
<tr>
<td><strong>Number of Points</strong></td>
<td><code>SENSe:SEGMenT:SWEep:POINt</code></td>
<td><code>seg.NumberOfPoints</code></td>
</tr>
<tr>
<td><strong>IF Bandwidth</strong></td>
<td><code>SENSe:SEGMenT:BWIDth</code></td>
<td><code>seg.IFBandwidth</code></td>
</tr>
<tr>
<td><strong>IF Bandwidth Option</strong></td>
<td><code>SENSe:SEGMenT:BWIDth:CONTrol</code></td>
<td><code>segs.IFBandwidthOption</code></td>
</tr>
<tr>
<td><strong>IF Bandwidth Per Port</strong></td>
<td><code>SENSe:SEGMenT:BWIDth:PORT:CONTrol</code></td>
<td><code>PortIFBandwidthOption</code></td>
</tr>
<tr>
<td><strong>Sweep Delay Time</strong></td>
<td><code>SENSe:SEGMenT:SWEep:DELay</code></td>
<td><code>DelayOption</code></td>
</tr>
<tr>
<td><strong>Sweep Dwell</strong></td>
<td><code>SENSe:SEGMenT:SWEep:DWELLI</code></td>
<td><code>SweepTimeOption</code></td>
</tr>
<tr>
<td><strong>Sweep Mode</strong></td>
<td><code>SENSe:SEGMenT:SWEep:GENeration</code></td>
<td><code>SweepModeOption</code></td>
</tr>
<tr>
<td><strong>Total Sweep Points</strong></td>
<td><code>SENSe:SEGMenT:SWEep:POINts:TOTal?</code></td>
<td><code>NumberOfPoints</code></td>
</tr>
<tr>
<td><strong>Total Sweep Time</strong></td>
<td><code>SENSe:SEGMenT:SWEep:TIME:TOTal?</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td><strong>Source Power</strong></td>
<td><code>SENSe:SEGMenT:POWer</code></td>
<td><code>chan.TestPortPower</code></td>
</tr>
<tr>
<td>Source Power Option</td>
<td>SENSe:SEGMent:POWer:CONTrol</td>
<td>segs.SourcePowerOption</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>X-Axis Point Spacing</td>
<td>SENSe:SEGMent:X:SPACing</td>
<td>chan.XAxisPointSpacing</td>
</tr>
<tr>
<td>Allow Arbitrary Segments</td>
<td>SENSe:SEGMent:ARBitrary</td>
<td>segs.AllowArbitrarySegments</td>
</tr>
<tr>
<td>Upload a segment table</td>
<td>SENSe:SEGMent:LIST</td>
<td>SetAllSegments</td>
</tr>
<tr>
<td>Download a segment table</td>
<td>SENSe:SEGMent:LIST</td>
<td>GetAllSegments</td>
</tr>
<tr>
<td>Sweep delay ON</td>
<td>OFF</td>
<td>SENSe:SEGMent:SWEep:DELay:CONTrol</td>
</tr>
<tr>
<td>Sweep dwell ON</td>
<td>OFF</td>
<td>SENSe:SEGMent:SWEep:DWELl:CONTrol</td>
</tr>
<tr>
<td>IF Bandwidth resolution</td>
<td>SENSe:SEGMent:BWIDth:PORT[:RESolution]</td>
<td>PortIFBandwidthOption/PortIFBandwidth</td>
</tr>
<tr>
<td>Sets or returns the SA data threshold</td>
<td>SENSe:SEGMent:SA:DTHReshold</td>
<td>SADataThreshold</td>
</tr>
<tr>
<td>Specifies whether SA Data Threshold can be set independently for each segment</td>
<td>SENSe:SEGMent:SA:DTHReshold:CONTrol</td>
<td>SADataThresholdOption</td>
</tr>
<tr>
<td>Sets or returns the SA multitone reference</td>
<td>SENSe:SEGMent:SA:MTReference</td>
<td>SAMTReference</td>
</tr>
<tr>
<td>Specifies whether SA Reference Tone can be set independently for each segment</td>
<td>SENSe:SEGMent:SA:MTReference:CONTrol</td>
<td>SAMTReferenceFreqOption</td>
</tr>
<tr>
<td>Queries the maximum value of the SA Reference Tone</td>
<td>SENSe:SEGMent:SA:MTReference:MAX?</td>
<td>None</td>
</tr>
<tr>
<td>Queries the minimum value of the SA Reference Tone</td>
<td>SENSe:SEGMent:SA:MTReference:MIN?</td>
<td>None</td>
</tr>
<tr>
<td>Sets or returns the SA vector average</td>
<td>SENSe:SEGMen:SA:VAVerage</td>
<td>SAVectorAverage</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>--------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Specifies whether SA Vector Averaging can be set independently for each segment</td>
<td>SENSe:SEGMen:SA:VAVerage:CONTrol</td>
<td>SAVectorAverageOption</td>
</tr>
<tr>
<td>Sets or returns the SA video bandwidth</td>
<td>SENSe:SEGMen:SA:VIDeobw</td>
<td>SAVideoBandwidth</td>
</tr>
<tr>
<td>Sets or returns the noise figure bandwidth</td>
<td>SENSe:SEGMen:NFBW</td>
<td>NoiseFigureBW</td>
</tr>
<tr>
<td>Turns ON or OFF the noise figure bandwidth setting.</td>
<td>SENSe:SEGMen:NFBW:CONTrol</td>
<td>NoiseFigureBWOption</td>
</tr>
<tr>
<td>Specifies whether SA Video Bandwidth can be set independently for each segment</td>
<td>SENSe:SEGMen:SA:VIDeobw:CONTrol</td>
<td>SAVideoAverageOption</td>
</tr>
</tbody>
</table>

### Source Phase Control / Sweep

<table>
<thead>
<tr>
<th>Source Phase Control / Sweep</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase Sweep type</td>
<td>SENSe:SWEep:TYPE</td>
</tr>
<tr>
<td>Set fixed phase value</td>
<td>SOURce:PHASe [:FIXed]</td>
</tr>
<tr>
<td>Phase sweep start value</td>
<td>SOURce:PHASe:STARt</td>
</tr>
<tr>
<td>Phase sweep stop value</td>
<td>SOURce:PHASe:STOP</td>
</tr>
<tr>
<td>Phase parameter</td>
<td>SOURce:PHASe:PARameter</td>
</tr>
<tr>
<td>Set Phase control mode</td>
<td>SOURce:PHASe:PARameter:MODE</td>
</tr>
<tr>
<td>Set reference port</td>
<td>SOURce:PHASe:PARameter:PORT</td>
</tr>
<tr>
<td>Read available phase control modes for the port</td>
<td>SOURce:PHASe:PARameter:MODE:CAT?</td>
</tr>
<tr>
<td>Couple sweep settings</td>
<td>SOURce:PHASe:CONTrol:COUPle</td>
</tr>
<tr>
<td>Set number of sweep iterations</td>
<td>SOURce:PHASe:CONTrol:ITERation</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>Set sweep tolerance</td>
<td>SOURce:PHASE:CONTrol:TOLerance</td>
</tr>
<tr>
<td>Set and read an array of phase offsets.</td>
<td>SOURce:PHASE:CORRection:DATA</td>
</tr>
<tr>
<td>Use phase offset array.</td>
<td>SOURce:PHASE:CORRection:STATE</td>
</tr>
<tr>
<td>Set and read an array of ratioed power offsets.</td>
<td>SOURce:PHASE:POFFset:CORRection:DATA</td>
</tr>
<tr>
<td>Use power offset array.</td>
<td>SOURce:PHASE:POFFset:CORRection:STATE</td>
</tr>
<tr>
<td>Set the fixed power ratioed value</td>
<td>SOURce:PHASE:POFFset:FIXed</td>
</tr>
<tr>
<td>Set the start power ratioed value</td>
<td>SOURce:PHASE:POFFset:STARt</td>
</tr>
<tr>
<td>Set the stop power ratioed value</td>
<td>SOURce:PHASE:POFFset:STOP</td>
</tr>
<tr>
<td>Returns the available internal ports that the external port can be set to.</td>
<td>SOURce:PHASE:EXTernal:CATalog?</td>
</tr>
<tr>
<td>Sets and returns the internal port that the external port is routed through.</td>
<td>SOURce:PHASE:EXTernal:PORT</td>
</tr>
<tr>
<td>Returns the available phase control modes for the specified port.</td>
<td>SOURce:PHASE:MODE:CATalog?</td>
</tr>
<tr>
<td>Sets and returns the Phase Control mode.</td>
<td>SOURce:PHASE:MODE:[VALUE]</td>
</tr>
<tr>
<td>Returns the available parameters.</td>
<td>SOURce:PHASE:PARameter:[CATalog?]</td>
</tr>
<tr>
<td>Sets and returns the ratioed receivers (parameter) to use for phase control.</td>
<td>SOURce:PHASE:PARameter:[VALUE]</td>
</tr>
<tr>
<td>Returns the available ports that can be used as phase control reference ports for the phase controlled port.</td>
<td>SOURce:PHASE:REReference:CATalog?</td>
</tr>
</tbody>
</table>
Sets and returns the reference port for the Phase Control measurement.

<table>
<thead>
<tr>
<th>Source (where trigger comes from)</th>
<th>Trigger</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source (Int</td>
<td>Ext</td>
</tr>
<tr>
<td>Internal</td>
<td>Manual</td>
</tr>
<tr>
<td>Scope (what is triggered)</td>
<td>TRIGger[:SEQUence]:SCOPe</td>
</tr>
<tr>
<td>Channel Settings (how the channel responds to triggers)</td>
<td></td>
</tr>
<tr>
<td>Continuous</td>
<td>SENSE:SWEep:MODE CONTinuous</td>
</tr>
<tr>
<td>Read Continuous Mode</td>
<td>None</td>
</tr>
<tr>
<td>Number of Groups</td>
<td>SENSE:SWEep:GROup:COUNt</td>
</tr>
<tr>
<td>Read Groups</td>
<td>None</td>
</tr>
<tr>
<td>Hold</td>
<td>SENSE:SWEep:MODE HOLD</td>
</tr>
<tr>
<td>Hold Mode (read-only)</td>
<td>None</td>
</tr>
<tr>
<td>All channels in Hold</td>
<td>SYSTem:CHANnels:HOLD</td>
</tr>
<tr>
<td>All channels Resume</td>
<td>SYSTem:CHANnels:RESume</td>
</tr>
<tr>
<td>Single</td>
<td>SENSE:SWEep:MODE SINGle</td>
</tr>
<tr>
<td>Trigger Mode (Channel</td>
<td>Point</td>
</tr>
<tr>
<td>Restart</td>
<td>INITiate[:IMMediate]</td>
</tr>
<tr>
<td>Abort</td>
<td>ABORT</td>
</tr>
</tbody>
</table>
### External Meas Trigger Input

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope (Global/Chan)</td>
<td>TRIGger[:SEQUence]:SCOPe</td>
<td>trigSetup.Scope</td>
</tr>
<tr>
<td>Trigger Delay (Global)</td>
<td>TRIGger:DELay</td>
<td>app.TriggerDelay</td>
</tr>
<tr>
<td>Trigger Delay (Channel)</td>
<td>SENSE:SWEep:TRIGger:DELay</td>
<td>chan.ExternalTriggerDelay</td>
</tr>
<tr>
<td>MeasTrigIn/ Hand I/O</td>
<td>TRIGger:ROUTE:INPut</td>
<td>ExternalTriggerConnectionBehavior</td>
</tr>
<tr>
<td>Level or Edge</td>
<td>TRIGger:TYPE</td>
<td>ExternalTriggerConnectionBehavior</td>
</tr>
<tr>
<td>Neg/Low or Pos/High</td>
<td>TRIGger:SLOPE</td>
<td>ExternalTriggerConnectionBehavior</td>
</tr>
<tr>
<td>Accept Trigger Before Armed</td>
<td>CONTROL:SIGNal:TRIGger:ATBA</td>
<td>AcceptTriggerBeforeArmed</td>
</tr>
</tbody>
</table>

### Ready for Trigger Indicator (Out)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>MeasTrig Rdy/ Hand I/O</td>
<td>TRIGger:ROUTE:READY</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>CONTROL:SIGNal:STReaml ine:RTRigger[:STATe]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CONTROL:SIGNal:PXI:RTRigger[:STATe]</td>
<td></td>
</tr>
<tr>
<td>Checks if the PNA is ready for a hardware trigger</td>
<td>TRIGger:STATus:READY?</td>
<td>ReadyForTriggerStatus</td>
</tr>
<tr>
<td>High / Low</td>
<td>TRIGger:READY:POLarity</td>
<td>ReadyForTriggerPolarity</td>
</tr>
</tbody>
</table>

### Auxiliary Triggering (PNA-X and N522x models)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Which AuxTrig connector pair being used</td>
<td>N/A</td>
<td>Number</td>
</tr>
<tr>
<td>How many Aux connector pairs</td>
<td>TRIGger:AUXiliary:COUNt?</td>
<td>AuxiliaryTriggerCount</td>
</tr>
</tbody>
</table>

### AUX TRIG OUT

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable</td>
<td>TRIGger:CHANnel:AUXiliary</td>
<td>Enable</td>
</tr>
<tr>
<td>Global or Channel Pref.</td>
<td>TRIGger:PREFerence:ALGLocal</td>
<td>AuxTriggerScopIsGlobal</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>Polarity (Pos/Neg)</td>
<td>TRIGger:CHANnel:AUXiliary:OPOLarity</td>
<td>TriggerOutPolarity</td>
</tr>
<tr>
<td>Position (Before/After acq)</td>
<td>TRIGger:CHANnel:AUXiliary:POSition</td>
<td>TriggerOutPosition</td>
</tr>
<tr>
<td>OUT Pulse width</td>
<td>TRIGger:CHANnel:AUXiliary:DURation</td>
<td>TriggerOutDuration</td>
</tr>
<tr>
<td>Point or Sweep.</td>
<td>TRIGger:CHANnel:AUXiliary:INTerval</td>
<td>TriggerOutInterval</td>
</tr>
<tr>
<td>Rear SMB</td>
<td>CONTrol:SIGNal:STReamline:TRIGger:OUTPut[:STATe]</td>
<td></td>
</tr>
<tr>
<td>Backplane</td>
<td>CONTrol:SIGNal:PXI:TRIGger:OUTPut[:STATe]</td>
<td></td>
</tr>
<tr>
<td>AUX TRIG (Ready) IN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable Handshake</td>
<td>TRIGger:CHANnel:AUXiliary:HANDshake</td>
<td>HandshakeEnable</td>
</tr>
<tr>
<td>Edge or Level</td>
<td>TRIGger:CHANnel:AUXiliary:TYPE</td>
<td>TriggerInType</td>
</tr>
<tr>
<td>Level NOT in UI.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polarity High/leading or</td>
<td>TRIGger:CHANnel:AUXiliary:IPOLarity</td>
<td>TriggerInPolarity</td>
</tr>
<tr>
<td>Low/trailing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delay</td>
<td>TRIGger:CHANnel:AUXiliary:DELay</td>
<td>Delay</td>
</tr>
</tbody>
</table>

### Pulse Measurement Setup (PNA-X)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse Meas Mode</td>
<td>SENSE:SWEPep:PULSe:MODE</td>
<td>PulseMeasMode</td>
</tr>
<tr>
<td>Autodetect Pulse mode</td>
<td>SENSE:SWEPep:PULSe:DETectmode</td>
<td>AutoDetection</td>
</tr>
<tr>
<td>Set Pulse Mode (Narrow</td>
<td>Wide)</td>
<td>SENSE:SWEPep:PULSe:WIdeband</td>
</tr>
<tr>
<td>Autoselect IFBW</td>
<td>SENSE:SWEPep:PULSe:CWTime[:AUTO]</td>
<td>AutoIFBandWidth</td>
</tr>
<tr>
<td>Autoselect IF Gain</td>
<td>SENSE:SWEPep:PULSe:IFGain</td>
<td>None</td>
</tr>
<tr>
<td>Autoselect Pulse clock period</td>
<td>SENSE:SWEPep:PULSe:PRF</td>
<td>AutoOptimizePRF</td>
</tr>
<tr>
<td>Autoselect Width and Delay</td>
<td>SENSE:SWEPep:PULSe:TIMing</td>
<td>AutoPulseTiming</td>
</tr>
<tr>
<td>Autoselect Pulse Gens</td>
<td>SENSE:SWEPep:PULSe:DRIVE</td>
<td>AutoSelectPulseGen</td>
</tr>
<tr>
<td>Autoselect CW Sweep Time</td>
<td>SENSE:SWEPep:PULSe:CWTtime</td>
<td>AutoCWSweepTime</td>
</tr>
</tbody>
</table>
### Pulse (manual) Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set master pulse frequency</td>
<td>SENSE:SWEep:PULSe:MASTer:FREQuency</td>
<td>MasterFrequency</td>
</tr>
<tr>
<td>Set master pulse period</td>
<td>SENSE:SWEep:PULSe:MASTer:PERiod</td>
<td>MasterPeriod</td>
</tr>
<tr>
<td>Set master pulse width</td>
<td>SENSE:SWEep:PULSe:MASTer:WIDTh</td>
<td>MasterWidth</td>
</tr>
<tr>
<td>Set pulse start time</td>
<td>SENSE:SWEep:PULSe:PROFile:STARt</td>
<td>PulseProfileStart</td>
</tr>
<tr>
<td>Set pulse stop time</td>
<td>SENSE:SWEep:PULSe:PROFile:STOP</td>
<td>PulseProfileStop</td>
</tr>
</tbody>
</table>

### Pulse Generators (PNA-X)

<table>
<thead>
<tr>
<th>Setting</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Pulse output ON</td>
<td>OFF</td>
<td>SENSE:PULSe&lt;n&gt;[:STATe]</td>
</tr>
<tr>
<td>Set Pulse Period for ALL pulse generators</td>
<td>SENSE:PULSe:PERiod</td>
<td>Period</td>
</tr>
<tr>
<td>Set Pulse Delay</td>
<td>SENSE:PULSe&lt;n&gt;:DElay</td>
<td>Delay</td>
</tr>
<tr>
<td>Set Pulse Width</td>
<td>SENSE:PULSe&lt;n&gt;:WIDTh</td>
<td>Width</td>
</tr>
<tr>
<td>Set Pulse delay increment</td>
<td>SENSE:PULSe&lt;n&gt;:DINCrement</td>
<td>DelayIncrement</td>
</tr>
<tr>
<td>Enables Subpoint triggering</td>
<td>SENSE:PULSe&lt;n&gt;:SUBPointtrig</td>
<td>SubPointTrigger</td>
</tr>
<tr>
<td>Set Pulse Invert</td>
<td>SENSE:PULSe&lt;n&gt;:INVert</td>
<td>Invert</td>
</tr>
<tr>
<td>Internal Pulse Modulator Enable</td>
<td>SENSE:PATH:CONFig:ELEMent</td>
<td>Element Value</td>
</tr>
</tbody>
</table>

**Choose from element value 8 or 9**

- Returns the ADC delay.
  - SENSE:PULSe:HDELay:ADC?
  - ADCDelay
- Sets the time lag between the pulse drive signal and the actual RF output.
  - SENSE:PULSe:HDELay:MODulator
  - ModulatorDelay
- Enables / Disables offset delays.
  - SENSE:PULSe:HDELay[:STATe]
  - EnableOffsetDelays
- Sets and reads the device being controlled by the pulse generator output.
  - SENSE:PULSe:MTIMing:DEVice
  - PulseTimingDevice
- Enable pulse4 to monitor ADC activity.
  - SENSE:PULSe4:OPTion
  - Pulse4OutAsADCActivity
- Sets the specific PULSe4 behavior.
  - SENSE:PULSe4:MODE
  - None
<table>
<thead>
<tr>
<th>DC Source Control</th>
<th>See Ext DC Device commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source names catalog</td>
<td>SOURce:DC:CATalog?</td>
</tr>
<tr>
<td>Enable source outputs</td>
<td>SOURce:DC:ENABLE</td>
</tr>
<tr>
<td>Source state</td>
<td>SOURce:DC:STATe</td>
</tr>
<tr>
<td>Start DC</td>
<td>SOURce:DC:STARt</td>
</tr>
<tr>
<td>Stop DC</td>
<td>SOURce:DC:STOP</td>
</tr>
<tr>
<td>Data</td>
<td>SOURce:DC:DATA</td>
</tr>
<tr>
<td>Set and return the Max DC limit value for a DC source</td>
<td>SOURce:DC:LIMit:MAXimum</td>
</tr>
<tr>
<td>Set and return the Min DC limit value for a DC source</td>
<td>SOURce:DC:LIMit:MINimum</td>
</tr>
</tbody>
</table>

**Multi-Dimensional Sweep**

Set and read the order for the specified DC source in the multi-dimensional sweep.

<table>
<thead>
<tr>
<th>SOURce:DC:DIMension:ORDER</th>
<th>DCOrder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Set and read the specified DC source’s ON/OFF state in the multi-dimensional sweep.</td>
<td>SOURce:DC:DIimension[:STATe]</td>
</tr>
<tr>
<td>Read the names of source domains in the multi-dimensional sweep whose state is ON and whose dimension order is the specified dimension order.</td>
<td>SOURce:DIimension:CATalog?</td>
</tr>
<tr>
<td>Read the highest dimension order in the multi-dimensional sweep.</td>
<td>SOURce:DIimension:COUNT?</td>
</tr>
<tr>
<td>Set and read the point count for the specified dimension order in the multi-dimensional sweep.</td>
<td>SOURce:DIimension:POINts</td>
</tr>
<tr>
<td>Set and read the repeat count for the specified dimension order in the multi-dimensional sweep.</td>
<td>SOURce:DIimension:REPeat:COUNT?</td>
</tr>
<tr>
<td>Set and read the source frequency domain’s order in the multi-dimensional sweep.</td>
<td>SOURce:FREQuency:DIimension:ORDer</td>
</tr>
<tr>
<td>Set and read the source frequency domain’s ON/OFF state in the multi-dimensional sweep.</td>
<td>SOURce:FREQuency:DIimension[:STATe]</td>
</tr>
<tr>
<td>Set and read the fixed frequency value for a specific port.</td>
<td>SOURce:FREQuency:FIXed</td>
</tr>
<tr>
<td>Set and read the start frequency value for a specific port.</td>
<td>SOURce:FREQuency:STARt</td>
</tr>
<tr>
<td>Set and read the stop frequency value for a specific port.</td>
<td>SOURce:FREQuency:STOP</td>
</tr>
<tr>
<td>Set and read the source phase domain’s order in the multi-dimensional sweep.</td>
<td>SOURce:PHAse:DIimension:ORDer</td>
</tr>
<tr>
<td>Set and read the source phase domain’s ON/OFF state in the multi-dimensional sweep.</td>
<td>SOURce:PHAse:DIimension[:STATe]</td>
</tr>
<tr>
<td>Set and read the source power domain’s order in the multi-dimensional sweep.</td>
<td>SOURce:POWer:DIimension:ORDer</td>
</tr>
<tr>
<td>Set and read the source power domain’s ON/OFF state in the multi-dimensional sweep.</td>
<td>SOURce:POWer:DIimension[:STATe]</td>
</tr>
<tr>
<td>Global Source Commands</td>
<td>Command</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Set and return the frequency of the specified global source.</td>
<td>SYSTem:PREFerences:SOURce:GLOBal:FREQuency</td>
</tr>
<tr>
<td>Set and return the output state of the specified global source.</td>
<td>SYSTem:PREFerences:SOURce:GLOBal:OUTPut[:STATe]</td>
</tr>
<tr>
<td>Set and return the global sources that ignore the power off setting.</td>
<td>SYSTem:PREFerences:SOURce:GLOBal:POFF:IGNore[:STATe]</td>
</tr>
<tr>
<td>Set and return the power of the specified global source.</td>
<td>SYSTem:PREFerences:SOURce:GLOBal:POWer</td>
</tr>
<tr>
<td>Set and return the global state of the specified global source.</td>
<td>SYSTem:PREFerences:SOURce:GLOBal[:STATe]</td>
</tr>
<tr>
<td>Description</td>
<td>SCPI</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------------------------------------</td>
</tr>
<tr>
<td>New Trace</td>
<td>DISPlay:WINDow:TRACe[:STATE]</td>
</tr>
<tr>
<td>Select Trace</td>
<td>DISPlay:WINDow:TRACe:SELect</td>
</tr>
<tr>
<td>Measure</td>
<td>CALCulate:MEASure:PARameter</td>
</tr>
<tr>
<td>Trace Title</td>
<td>DISPlay:WINDow:TRACe:TITLE:DATA</td>
</tr>
<tr>
<td></td>
<td>DISPlay:WINDow:TRACe:TITLE[:STATE]</td>
</tr>
<tr>
<td>Add Trace</td>
<td>DISPlay:WINDow:TRACe[:STATE]</td>
</tr>
<tr>
<td>Delete Trace</td>
<td>DISPlay:WINDow:TRACe:DELeete</td>
</tr>
<tr>
<td>Move Trace</td>
<td>DISPlay:WINDow:TRACe:MOVE</td>
</tr>
<tr>
<td>Hold Trace</td>
<td>CALCulate:MEASure:HOLD:TYPE</td>
</tr>
<tr>
<td></td>
<td>CALCulate:MEASure:HOLD:CLEar</td>
</tr>
<tr>
<td>Set/get the number of traces of selected channel</td>
<td>CALCulate:PARameter:COUNt</td>
</tr>
<tr>
<td>Trace Maximize</td>
<td>DISPlay:TMAX</td>
</tr>
</tbody>
</table>

### Channel

<table>
<thead>
<tr>
<th></th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>None</td>
<td>chans.Add</td>
</tr>
<tr>
<td>Make Active</td>
<td>None</td>
<td>app.ActiveChannel</td>
</tr>
<tr>
<td>Read Channel Number</td>
<td>SYSTem:ACTive:CHANnel?</td>
<td>chan.ChannelNumber</td>
</tr>
<tr>
<td>Read UNUSED channel numbers</td>
<td>None</td>
<td>chans.UnusedChannelNumbers</td>
</tr>
<tr>
<td>Read used channel numbers</td>
<td>None</td>
<td>chans.UsedChannelNumbers</td>
</tr>
<tr>
<td>Read number of Channels</td>
<td>None</td>
<td>chans.Count</td>
</tr>
<tr>
<td>Copy all Channel settings</td>
<td>SYSTem:MACRo:COPY:CHANnel</td>
<td>chan.CopyToChannel</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Copy ONLY mechanical switches and attenuator settings.</td>
<td>SENSE:PATH:CONFig:COPY</td>
<td>CopyFrom</td>
</tr>
<tr>
<td>Delete a channel</td>
<td>SYSTem:CHANnels:DELeTe</td>
<td>Remove Method RemoveChannelNumber</td>
</tr>
<tr>
<td>Set and return the group of channels</td>
<td>SYSTem:CHANnels:COUPle:GROup</td>
<td>None</td>
</tr>
<tr>
<td>Set and return the Multi DUT parallel measurement state</td>
<td>SYSTem:CHANnels:COUPle:PARallel[:ENABLE]</td>
<td>None</td>
</tr>
<tr>
<td>Get the information if the parallel measurement is executed in the last sweep</td>
<td>SYSTem:CHANnels:COUPle:PARallel:STATe?</td>
<td>None</td>
</tr>
<tr>
<td>Set up multiple channels for manual trigger</td>
<td>SYSTem:CHANnels:SINGle</td>
<td>None</td>
</tr>
</tbody>
</table>

### Display - Window Setup

<table>
<thead>
<tr>
<th>Select Window</th>
<th>DISPlay:WINDow:TRACE:SELect</th>
<th>ActivateWindow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Window Title</td>
<td>DISPlay:WINDow:TITLe[:STAte]</td>
<td>TitleState</td>
</tr>
<tr>
<td>Add Window</td>
<td>DISPlay:WINDow[:STAte]</td>
<td>Add</td>
</tr>
<tr>
<td>Delete Window</td>
<td>DISPlay:WINDow[:STAte]</td>
<td>None</td>
</tr>
<tr>
<td>Move Window</td>
<td>DISPlay:WINDow:TRACE:MOVE</td>
<td>Move</td>
</tr>
<tr>
<td>Window Layout</td>
<td>DISPlay:ARRange</td>
<td>ArrangeWindows</td>
</tr>
<tr>
<td>Window Max</td>
<td>DISPlay:WINDow:SIZe</td>
<td>WindowState</td>
</tr>
<tr>
<td>Return Window Number(s)</td>
<td>DISPlay:CATalog?</td>
<td>win.WindowNumber</td>
</tr>
<tr>
<td>Read the window number of the selected trace</td>
<td>CALCulate:PARameter:WNUMber</td>
<td>win.WindowNumber</td>
</tr>
<tr>
<td>Creates N windows</td>
<td>DISPlay:SPLit</td>
<td>None</td>
</tr>
<tr>
<td>Feed specified measurement to specified window</td>
<td>DISPlay:WINDow:TRACE:FEED:MNUMber</td>
<td>None</td>
</tr>
<tr>
<td>Returns the next unused trace number</td>
<td>DISPlay:WINDow:TRACE:NEXT[:NUMBer]</td>
<td>None</td>
</tr>
<tr>
<td>Set graph divisions</td>
<td>DISPlay:WINDow:Y[:SCALe]:DIVisions</td>
<td>None</td>
</tr>
</tbody>
</table>
### Display - Sheet Setup

<table>
<thead>
<tr>
<th>Operation</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select Sheet</td>
<td>DISPlay:SHEet:STATE</td>
<td>None</td>
</tr>
<tr>
<td>Sheet Title</td>
<td>DISPlay:SHEet:TITle:DATA</td>
<td>None</td>
</tr>
<tr>
<td>Add Sheet</td>
<td>DISPlay:SHEet:STATE DISPlay:WINDow:FEED</td>
<td>None</td>
</tr>
<tr>
<td>Delete Sheet</td>
<td>DISPlay:SHEet:STATE DISPlay:WINDow:FEED</td>
<td>None</td>
</tr>
<tr>
<td>Sheet Layout</td>
<td>DISPlay:SHEet:STATE</td>
<td>None</td>
</tr>
<tr>
<td>Get list of window numbers which the sheet contains</td>
<td>DISPlay:SHEet:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Feed specified window to a sheet</td>
<td>DISPlay:WINDow:FEED</td>
<td>None</td>
</tr>
<tr>
<td>Return active sheet number</td>
<td>SYSTem:ACTive:SHEet</td>
<td>None</td>
</tr>
<tr>
<td>Return list of visible sheets</td>
<td>SYSTem:SHEets:CATalog?</td>
<td>None</td>
</tr>
</tbody>
</table>

### Display Setup

<table>
<thead>
<tr>
<th>Setting</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trace Status</td>
<td>DISPlay:WINDow:ANNotation[:TRACe][:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Y-axis Labels</td>
<td>DISPlay:WINDow:ANNotation:Y[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Show Marker Readout</td>
<td>DISPlay:WINDow:ANNotation:MARKer[:STATE]</td>
<td>MarkerReadout</td>
</tr>
<tr>
<td>Large Readout</td>
<td>DISPlay:WINDow:ANNotation:MARKer:SIZE</td>
<td>MarkerReadoutSize</td>
</tr>
<tr>
<td>Readouts Per Trace</td>
<td>DISPlay:WINDow:ANNotation:MARKer:NUMBer</td>
<td>MarkerReadoutsPerTrace</td>
</tr>
<tr>
<td>Sets the marker readouts to coupled (one combination annotation) or not coupled (one annotation per trace).</td>
<td>DISPlay:WINDow:ANNotation:MARKer:COUPle</td>
<td>None</td>
</tr>
<tr>
<td>Shows the marker readouts only for active trace or for all traces.</td>
<td>DISPlay:WINDow:ANNotation:MARKer:VISible</td>
<td>None</td>
</tr>
<tr>
<td>Symbol - Triangle, Flag, and Line</td>
<td>DISPlay:WINDow:ANnotation:MARKer:SYMBol</td>
<td>MarkerSymbol</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Marker Colors</td>
<td>DISPlay:COlor:TRACe:MARKer</td>
<td>Markers</td>
</tr>
<tr>
<td>N Trace: Markers</td>
<td>DISPlay:COlor:TRACe:MARKer</td>
<td>Markers</td>
</tr>
<tr>
<td>N Trace: Memory Markers</td>
<td>DISPlay:COlor:TRACe:MMARker</td>
<td>MemoryMarkers</td>
</tr>
<tr>
<td>Reset Color</td>
<td>DISPlay:COlor:RESet</td>
<td>None</td>
</tr>
<tr>
<td>Save Theme</td>
<td>DISPlay:COlor:STORe</td>
<td>StoreTheme</td>
</tr>
<tr>
<td>Recall Theme</td>
<td>DISPlay:COlor:LOAD</td>
<td>LoadTheme</td>
</tr>
<tr>
<td>Reset Theme</td>
<td>DISPlay:COlor:RESet</td>
<td>ResetTheme</td>
</tr>
<tr>
<td>Grid Lines - Solid</td>
<td>DISPlay:WINDow:TRACe:GRATicule:GRID:LTYPe</td>
<td>GridLineType</td>
</tr>
<tr>
<td>Dotted</td>
<td>DISPlay:WINDow:TRACe:GRATicule:GRID:LTYPe</td>
<td>GridLineType</td>
</tr>
<tr>
<td>Y-axis Divisions - 2 to 30</td>
<td>DISPlay:WINDow:TRACe:Y[:SCALe]:PDIvision</td>
<td>YScale</td>
</tr>
<tr>
<td>Show Table - None, Marker, Limit, Ripple, and Segment</td>
<td>DISPlay:WINDow:TABLe</td>
<td>ShowTable</td>
</tr>
<tr>
<td>Toolbar Softkey</td>
<td>DISPlay:TOOLbar:KEYS[:STATe]</td>
<td>ShowToolbar</td>
</tr>
<tr>
<td>Toolbar Hardkey</td>
<td>DISPlay:TOOLbar:KEYS[:STATe]</td>
<td>ShowToolbar</td>
</tr>
<tr>
<td>Toolbar Port Extensions</td>
<td>DISPlay:TOOLbar:EXTensions[:STATe]</td>
<td>ShowToolbar</td>
</tr>
<tr>
<td>Toolbar Transform</td>
<td>DISPlay:TOOLbar:TRANsform[:STATe]</td>
<td>ShowToolbar</td>
</tr>
<tr>
<td>Toolbar Marker</td>
<td>DISPlay:TOOLbar:MARKer[:STATe]</td>
<td>ShowToolbar</td>
</tr>
<tr>
<td>Toolbar Cal Set Viewer</td>
<td>DISPlay:TOOLbar:CSET[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Feature</td>
<td>Command Details</td>
<td>Status</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>Active Entry Toolbar</td>
<td>DISPlay:TOOLbar:ENTRY[:STATE]</td>
<td>ShowToolbar</td>
</tr>
<tr>
<td>Toolbar Status Bar</td>
<td>DISPlay:ANNote[:STATus]</td>
<td>ShowStatusBar</td>
</tr>
<tr>
<td>Status Bar</td>
<td>DISPlay:ANNote[:STATus]</td>
<td>ShowStatusBar</td>
</tr>
<tr>
<td>Status Bar Clock</td>
<td>SYSTem:CLOCk[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Default Colors</td>
<td>DISPlay:COlor:RESet</td>
<td>ResetTheme</td>
</tr>
<tr>
<td>Background Colors</td>
<td>DISPlay:COlor:BACKground</td>
<td>Background</td>
</tr>
<tr>
<td>Active Background Color</td>
<td>DISPlay:COlor:ABACkground</td>
<td>ActiveBackground</td>
</tr>
<tr>
<td>Grid Colors</td>
<td>DISPlay:COlor:GRAT2</td>
<td>Grid</td>
</tr>
<tr>
<td>Active Labels, Grid Frame Colors</td>
<td>DISPlay:COlor:GRAT1</td>
<td>ActiveLabels</td>
</tr>
<tr>
<td>Inactive Window Labels Colors</td>
<td>DISPlay:COlor:ILABel</td>
<td>InactiveLabels</td>
</tr>
<tr>
<td>Failed Trace Colors</td>
<td>DISPlay:COlor:LIM1</td>
<td>FailedTraces</td>
</tr>
<tr>
<td>N Trace: Data and Limits Colors</td>
<td>DISPlay:COlor:TRACe:DATA</td>
<td>DataAndLimits</td>
</tr>
<tr>
<td>N Trace: Memory</td>
<td>DISPlay:COlor:TRACe:MEMory</td>
<td>Memory</td>
</tr>
<tr>
<td>Display Update</td>
<td>DISPlay:UPDate[:STATE]</td>
<td>None</td>
</tr>
</tbody>
</table>

### Configure External Devices

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Command Details</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adds an external device to the system.</td>
<td>SYSTem:CONFigure:EDEVice:ADD</td>
<td>Add (External Device)</td>
</tr>
<tr>
<td>Returns names of all configured devices</td>
<td>SYSTem:CONFigure:EDEVice:CAT?</td>
<td>Items</td>
</tr>
<tr>
<td>Set driver for the external device.</td>
<td>SYSTem:CONFigure:EDEVice:DRiver</td>
<td>Driver</td>
</tr>
<tr>
<td>Set type of device.</td>
<td>SYSTem:CONFigure:EDEVice:DTYPe</td>
<td>DeviceType</td>
</tr>
<tr>
<td>Configuration path for external device.</td>
<td>SYSTem:CONFigure:EDEVice:IOConfig</td>
<td>IOConfiguration</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Option</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>Enable or disable communication with device.</td>
<td><code>SYSTem:CONFigure:EDEVice:IOENable</code></td>
<td><code>IOEnable</code></td>
</tr>
<tr>
<td>Enable/disable an SMU channel.</td>
<td><code>SYSTem:CONFigure:EDEVice:SMU:CHANnel[1-4]:STATe</code></td>
<td><code>ChanActive</code></td>
</tr>
<tr>
<td>Sets and returns the name of the External Device.</td>
<td><code>None</code></td>
<td><code>Name</code></td>
</tr>
<tr>
<td>Activation state of the device.</td>
<td><code>SYSTem:CONFigure:EDEVice:STATe</code></td>
<td><code>Active</code></td>
</tr>
<tr>
<td>Time out value for external device.</td>
<td><code>SYSTem:CONFigure:EDEVice:TOUT</code></td>
<td><code>TimeOut</code></td>
</tr>
<tr>
<td>Remove a device</td>
<td><code>SYSTem:CONFigure:EDEVice:REMove</code></td>
<td><code>Remove</code></td>
</tr>
<tr>
<td>Save configuration file</td>
<td><code>SYSTem:CONFigure:EDEVice:SAVE</code></td>
<td><code>SaveFile</code></td>
</tr>
<tr>
<td>Load configuration file</td>
<td><code>SYSTem:CONFigure:EDEVice:LOAD</code></td>
<td><code>LoadFile</code></td>
</tr>
<tr>
<td>Returns if specified device responds</td>
<td><code>SYSTem:CONFigure:EDEVice:EXISts?</code></td>
<td><code>IsDevicePresent</code></td>
</tr>
</tbody>
</table>

**External Source Config**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Dwell per Point</td>
<td><code>SYSTem:CONFigure:EDEVice:SOURce:DPP</code></td>
<td><code>DwellPerPoint</code></td>
</tr>
<tr>
<td>Set Trigger Mode</td>
<td><code>SYSTem:CONFigure:EDEVice:SOURce:TMODe</code></td>
<td><code>Trigger Mode</code></td>
</tr>
<tr>
<td>Set Trigger Port</td>
<td><code>SYSTem:CONFigure:EDEVice:SOURce:TPORt</code></td>
<td><code>TriggerPort</code></td>
</tr>
<tr>
<td>Set Modulation Control</td>
<td><code>SYSTem:CONFigure:EDEVice:SOURce:MODulation:CONTrol</code></td>
<td><code>None</code></td>
</tr>
</tbody>
</table>

**Source Modulation**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and reads the frequency of the arbitrary waveform generator.</td>
<td><code>SOURce:MODulation:ARB:CLOCk:SRATe</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td>Sets and reads the I data for I/Q modulation.</td>
<td><code>SOURce:MODulation:ARB:DATA:I</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td>Sets and reads the Q data for I/Q modulation.</td>
<td><code>SOURce:MODulation:ARB:DATA:Q</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td>Set and read the modulation state.</td>
<td><code>SOURce:MODulation:[STATe]</code></td>
<td><code>None</code></td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Result</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Checks if pulse source exists.</td>
<td><strong>SOURce:PULSe:EXISts?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Turns pulse modulation on and off with an external source.</td>
<td><strong>SOURce:PULSe:MODulation[:STATe]</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations for an ACP modulation calibration.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:ACP:ITERations</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration plane for an ACP modulation calibration.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:ACP:RECeiver</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration span for an ACP modulation calibration.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:ACP:SPAN</strong></td>
<td>None</td>
</tr>
<tr>
<td>Set and read the ACP modulation calibration state.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:ACP:ENABle</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired ACP calibration tolerance for the ACP modulation calibration.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:ACP:TOLerance</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets the Notch location.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:ACQuire</strong></td>
<td>None</td>
</tr>
<tr>
<td>Returns a message indicating if the calibration was successful or not.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:ACQuire:STATus?</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations for a flatness modulation calibration.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:FLATness:ITERations</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration plane for flatness modulation calibration.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:FLATness:RECeiver</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration span for a flatness modulation calibration.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:FLATness:SPAN</strong></td>
<td>None</td>
</tr>
<tr>
<td>Set and read the flatness modulation calibration state.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:FLATness:ENABle</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired flatness calibration tolerance for the flatness modulation calibration.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:FLATness:TOLerance</strong></td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations to provide the deepest notch.</td>
<td><strong>SOURce:MODulation:CORRection:COLLection:NOTch:ITERations</strong></td>
<td>None</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Result</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Sets and reads the calibration plane for an notch modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration span for a notch modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:SPAN</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the notch modulation calibration state.</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:ENABLE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired notch calibration tolerance for the notch modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:NOTch:TOLerance</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the maximum number of iterations for a power modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:ITERations</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration plane for a power modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:RECeiver</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the calibration span for a power modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:SPAN</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the power modulation calibration state.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:ENABLE</td>
<td>None</td>
</tr>
<tr>
<td>Sets and reads the desired power calibration tolerance for the power modulation calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:POWer:TOLerance</td>
<td>None</td>
</tr>
<tr>
<td>Set and read the modulation correction state.</td>
<td>SOURce:MODulation:CORRection[:STATE]</td>
<td>None</td>
</tr>
<tr>
<td>Returns a list of modulation files (*.mdx).</td>
<td>SOURce:MODulation:FILE</td>
<td>None</td>
</tr>
<tr>
<td>Loads the specified modulation file.</td>
<td>SOURce:MODulation:LOAD</td>
<td>None</td>
</tr>
<tr>
<td>Saves the specified modulation file.</td>
<td>SOURce:MODulation:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>Enables fast calibration.</td>
<td>SOURce:MODulation:CORRection:COLLection:FAST:ENABLE</td>
<td>None</td>
</tr>
</tbody>
</table>

**Power Meter As Receiver (PMAR) Config**

*See commands to configure and specify a Non- PMAR Power Sensor*

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable use of internal cal factors</td>
<td>SYSTem:CONFigure:EDEVice:PMAR:CFACtors[:STATE]</td>
<td>UseInternalCalFactors</td>
</tr>
<tr>
<td>Feature</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>Enable min and max freq</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:FLIMit</td>
<td>LimitFrequency</td>
</tr>
<tr>
<td>Set Max freq</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:FMAXimum</td>
<td>MaximumFrequency</td>
</tr>
<tr>
<td>Set Min freq</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:FMINimum</td>
<td>MinimumFrequency</td>
</tr>
<tr>
<td>Set max number of PM readings</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:READing:COUNT</td>
<td>ReadingsPerPoint</td>
</tr>
<tr>
<td>Set tolerance level</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:READing:NTOLerance</td>
<td>ReadingsTolerance</td>
</tr>
<tr>
<td>Select sensor</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:SENSor</td>
<td>SensorIndex</td>
</tr>
<tr>
<td>Set Cal Factor data</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:TABLE:CFAC:DATA</td>
<td>ReferenceCalFactor</td>
</tr>
<tr>
<td>Set Cal Factor frequencies</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:TABLE:CFAC:FREQuency</td>
<td>Frequency</td>
</tr>
<tr>
<td>Set Power loss data</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:TABLE:LOSS:DATA</td>
<td>Loss</td>
</tr>
<tr>
<td>Set Power loss frequencies</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:TABLE:LOSS:FREQuency</td>
<td>Frequency</td>
</tr>
<tr>
<td>Enable Power loss data</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:TABLE:LOSS:STATe</td>
<td>UsePowerLossSegments</td>
</tr>
<tr>
<td>Set reference cal factor</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:TABLE:RFACtor</td>
<td>ReferenceCalFactor</td>
</tr>
<tr>
<td>Set Zero method</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:ZERO</td>
<td>None</td>
</tr>
<tr>
<td>Perform Cal</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:CALibrate</td>
<td>None</td>
</tr>
<tr>
<td>Returns a list of available power meters</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:UNCertainty:CATalog?</td>
<td>UncertaintyModelCatalog?</td>
</tr>
<tr>
<td>Sets and returns a custom model uncertainty file</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:UNCertainty:FILE</td>
<td>UncertaintyFile</td>
</tr>
<tr>
<td>Returns a list of available power meters</td>
<td>SYSTem:CONFigure'EDEVice:PMAR:UNCertainty:MODel</td>
<td>UncertaintyModel</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Variable</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>Returns the power level for best accuracy.</td>
<td><code>SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:PLEVel?</code></td>
<td><code>PowerForBestAccuracy</code></td>
</tr>
<tr>
<td>Returns the power meter reading uncertainty.</td>
<td><code>SYSTem:CONFigure:EDEVice:PMAR:UNCertainty:READ?</code></td>
<td><code>PowerMtrReadingUncertainty</code></td>
</tr>
<tr>
<td><strong>External DC Source/Meter</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>See DC Source sweep commands</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction ON/OFF</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:CORRection</code></td>
<td><code>DCCorrection</code></td>
</tr>
<tr>
<td>Offset correction value.</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:OFFSet</code></td>
<td><code>DCOffset</code></td>
</tr>
<tr>
<td>Scale correction value.</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:SCALe</code></td>
<td><code>DCScale</code></td>
</tr>
<tr>
<td>DC Type (Units).</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:TYPE</code></td>
<td><code>DCType</code></td>
</tr>
<tr>
<td>Dwell Before/After Point</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:DPOInt</code></td>
<td><code>PointDwell</code></td>
</tr>
<tr>
<td>Dwell Before Sweep value</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:DSWeep</code></td>
<td><code>SweepDwell</code></td>
</tr>
<tr>
<td>Set and return the maximum output current value of the external DC Source</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:LIMit:CURRent</code></td>
<td><code>CurrentLimit</code></td>
</tr>
<tr>
<td>Set and return the maximum output voltage value of the external DC Source</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:LIMit:VOLTage</code></td>
<td><code>VoltageLimit</code></td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source Abort Sweep command</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:COMMand:SWEep:ABORt</code></td>
<td><code>AbortSweepCmd</code></td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source After Sweep command</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:COMMand:SWEep:AFTer</code></td>
<td><code>AfterSweepCmd</code></td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source Before Sweep command</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:COMMand:SWEep:BEFore</code></td>
<td><code>BeforeSweepCmd</code></td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source Error Query command</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:QUERy:ERRor</code></td>
<td><code>ErrorQuery</code></td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source Disable I/O command</td>
<td><code>SYSTem:CONFigure:EDEVice:DC:COMMand:EXIT</code></td>
<td><code>ExitCmd</code></td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source ID Query command</td>
<td>SYSTem:CONFigure:EDEVice:DC:QUERy:ID</td>
<td>IDQuery</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>--------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Set and return the DC Meter/DC Source Enable I/O command</td>
<td>SYSTem:CONFigure:EDEVice:DC:COMmand:INIT</td>
<td>InitCmd</td>
</tr>
<tr>
<td>Set and return the DC Source maximum output</td>
<td>SYSTem:CONFigure:EDEVice:DC:MAX:VALue</td>
<td>MaxOutput</td>
</tr>
<tr>
<td>Set and return the DC Source maximum output state</td>
<td>SYSTem:CONFigure:EDEVice:DC:MAX[:STATE]</td>
<td>MaxOutputState</td>
</tr>
<tr>
<td>Set and return the DC Source minimum output</td>
<td>SYSTem:CONFigure:EDEVice:DC:MIN:VALue</td>
<td>MinOutput</td>
</tr>
<tr>
<td>Set and return the DC Source minimum output state</td>
<td>SYSTem:CONFigure:EDEVice:DC:MIN[:STATE]</td>
<td>MinOutputState</td>
</tr>
<tr>
<td>Set and return the Point Read commands and Point Set commands</td>
<td>SYSTem:CONFigure:EDEVice:DC:COMmand:POINt:SET</td>
<td>PointCmd</td>
</tr>
</tbody>
</table>

**External Pulse Generators**

<table>
<thead>
<tr>
<th>PG Names catalog</th>
<th>SENSE:PUlSe:CATalog?</th>
<th>PulseGeneratorNames</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read the integer of the name</td>
<td>Not applicable</td>
<td>PulseGeneratorID</td>
</tr>
<tr>
<td>Set output channel</td>
<td>SYSTem:CONFigure:EDEVice:PUlSe:CHAN</td>
<td>OutputChannel</td>
</tr>
<tr>
<td>Set output Hi amplitude (volts)</td>
<td>SYSTem:CONFigure:EDEVice:PUlSe:HAMP</td>
<td>HighAmplitude</td>
</tr>
<tr>
<td>Set output Low amplitude (volts)</td>
<td>SYSTem:CONFigure:EDEVice:PUlSe:LAMP</td>
<td>LowAmplitude</td>
</tr>
<tr>
<td>Set load impedance</td>
<td>SYSTem:CONFigure:EDEVice:PUlSe:LIMP</td>
<td>LoadImpedance</td>
</tr>
<tr>
<td>Set source impedance</td>
<td>SYSTem:CONFigure:EDEVice:PUlSe:SIMP</td>
<td>SourceImpedance</td>
</tr>
<tr>
<td>Master Mode</td>
<td>SYSTem:CONFigure:EDEVice:PUlSe:MMODe</td>
<td>MasterMode</td>
</tr>
<tr>
<td>Optional Name/ID argument added to some Pulse gen commands.</td>
<td>SENSE:PUlSe</td>
<td>PulseGenerator Object</td>
</tr>
</tbody>
</table>

**New Measurement**
<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create S-Parameter Meas.</td>
<td>CALCulate:MEASure::DEFine</td>
<td>app.CreateSParameterEx</td>
</tr>
<tr>
<td>Create Measurement</td>
<td>CALCulate:MEASure::DEFine</td>
<td>app.CreateMeasurement</td>
</tr>
<tr>
<td>Add Measurement</td>
<td>None</td>
<td>meass.Add</td>
</tr>
<tr>
<td>List Measurements</td>
<td>CALCulate:PARameter:CATalog:EXTended</td>
<td>None</td>
</tr>
<tr>
<td>Delete Measurement</td>
<td>CALCulate:MEASure::DELeTe</td>
<td>Measurements.Remove</td>
</tr>
<tr>
<td>Delete ALL measurements</td>
<td>CALCulate:MEASure::DELeTe:ALL</td>
<td>None</td>
</tr>
<tr>
<td>Manage Measurements</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Get a handle to a Trace</td>
<td>None</td>
<td>win.ActiveTrace</td>
</tr>
<tr>
<td>Select a Measurement Parameter</td>
<td>CALCulate:PARameter:SELeCT</td>
<td>app.ActiveMeasurement</td>
</tr>
<tr>
<td>Read Channel Number</td>
<td>SYSTem:ACTive:CHANnel?</td>
<td>chan.ChannelNumber</td>
</tr>
<tr>
<td>Read Channel Numbers in use</td>
<td>SYSTem:CHANnels:CATalog?</td>
<td>None</td>
</tr>
<tr>
<td>Read Number of Measurements</td>
<td>None</td>
<td>chans.Count</td>
</tr>
<tr>
<td>Read Measurement Parameter</td>
<td>None</td>
<td>meas.Parameter</td>
</tr>
<tr>
<td>Set / Read Measurement Name</td>
<td>SYSTem:ACTive:MEASurement (read-only)</td>
<td>meas.Name</td>
</tr>
<tr>
<td>Read Active Measurement Number</td>
<td>SYSTem:ACTive:MEASurement:NUMBer?</td>
<td>meas.Number</td>
</tr>
<tr>
<td>Change Parameter</td>
<td>CALCulate:MEASure:PARameter</td>
<td>meas.ChangeParameter</td>
</tr>
<tr>
<td>Returns the Measurement Class name</td>
<td>SENE:CLASs:NAME?</td>
<td>Get_MeasurementClass</td>
</tr>
<tr>
<td>Read the window number of the selected trace</td>
<td>CALCulate:PARameter:WNUMber</td>
<td>None</td>
</tr>
<tr>
<td>Read the trace number of the selected trace</td>
<td>CALCulate:PARameter:TNUMber?</td>
<td>None</td>
</tr>
<tr>
<td>Maximize (Isolate) trace</td>
<td>DISPLAY:TMAX</td>
<td>TraceMax</td>
</tr>
<tr>
<td>Move a trace to another window</td>
<td>DISPLAY:WINDow:TRACe:MOVE</td>
<td>Trace.Move</td>
</tr>
<tr>
<td>Trace Hold</td>
<td>CALCulate:MEASure:HOLD:TYPE</td>
<td>TraceHoldType</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Trace Hold Clear</td>
<td>CALCulate:MEASure:HOLD:CLEar</td>
<td>TraceHoldClear</td>
</tr>
<tr>
<td>Deletes the trace associated with the specified measurement number</td>
<td>DISPlay:MEASure:DELeate</td>
<td>None</td>
</tr>
<tr>
<td>Create a new trace in the specified window</td>
<td>DISPlay:MEASure:FEED</td>
<td>None</td>
</tr>
<tr>
<td>Turn the memory trace ON or OFF for the specified measurement</td>
<td>DISPlay:MEASure:MEMory[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Move a trace associated with measurement number to the specified window</td>
<td>DISPlay:MEASure:MOVE</td>
<td>None</td>
</tr>
<tr>
<td>Activate the specified measurement to be selected</td>
<td>DISPlay:MEASure:SELect</td>
<td>None</td>
</tr>
<tr>
<td>Turn trace display associated with the specified measurement ON or OFF</td>
<td>DISPlay:MEASure[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Set or return the title for the specified measurement</td>
<td>DISPlay:MEASure:TITLe:DATA</td>
<td>None</td>
</tr>
<tr>
<td>Turn the measurement title ON or OFF</td>
<td>DISPlay:MEASure:TITLe[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Autoscale the specified trace in the specified measurement</td>
<td>DISPlay:MEASure:Y[:SCALe]:AUTO</td>
<td>None</td>
</tr>
<tr>
<td>Set the Y axis Scale Per Division value of the specified trace associated with the specified measurement</td>
<td>DISPlay:MEASure:Y[:SCALe]:PDIvision</td>
<td>None</td>
</tr>
<tr>
<td>Set the Y axis Reference Level of the specified trace associated with the specified measurement</td>
<td>DISPlay:MEASure:Y[:SCALe]:RLEVel</td>
<td>None</td>
</tr>
<tr>
<td>Set the Reference Position of the specified trace associated with the specified measurement</td>
<td><strong>DISPlay:MEASure:Y[:SCALe]:RPOSition</strong></td>
<td>None</td>
</tr>
<tr>
<td>All Measurement Classes / Applications</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Balanced Measurements and Fixturing

| Configure Topology | **CALCulate:FSIMulator:BALun** | BalancedTopology Object |
| Configure Balanced Measurement | **CALCulate:FSIMulator:BALun** | BalancedMeasurement Object |
| Maps the physical VNA ports to a device of balanced and single-ended logical ports for multi-port systems with greater than 4 ports | **CALCulate:DTOPology** | SetCustomDUTTopology |

### Interface Control

| Interface Control ON|OFF | **CONTrol:CHANnel:INTerface:CONTrol[:STATe]** | IC.State |
| Recall Interface Control file | **CONTrol:CHANnel:INTerface:CONTrol:CONFig:RECall** | IC.ConfigurationFile |

### External Testset Control (also for E5092A)

<p>| Returns a list of currently supported test sets. | <strong>SENSe:MULTiplexer:CATalog?</strong> | tsts.TestsetCatalog |
| Load config file and Restart VNA. | <strong>SYStem:CONFigure</strong> | app.Configure |
| Loads a test set configuration file. (and SCPI only - sets ID value). | <strong>SENSe:MULTiplexer:TYPE</strong> | ts.Add |
| Returns the test set model | <strong>SENSe:MULTiplexer:TYPE</strong> | ts.Type |
| Returns the test set ID number. | None | ts.ID |
| Returns the number of input ports | <strong>SENSe:MULTiplexer:INCount?</strong> | None |</p>
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switches an input to one of the valid outputs (E5091A only).</td>
<td>None</td>
<td>OutputPort</td>
</tr>
<tr>
<td>Returns the total number of ports on the test set.</td>
<td>SENSE:MULTiplexer:COUNt?</td>
<td>NumberOfPorts</td>
</tr>
<tr>
<td>Sets and returns the address for the external test set at the specified ID.</td>
<td>SENSE:MULTiplexer:ADDRess</td>
<td>None</td>
</tr>
<tr>
<td>Turns ON/OFF the port mapping and control line output.</td>
<td>SENSE:MULTiplexer:STATe</td>
<td>Enabled</td>
</tr>
<tr>
<td>Sets and returns the port mappings for ALL ports.</td>
<td>SENSE:MULTiplexer:ALLPorts</td>
<td>OutputPorts</td>
</tr>
<tr>
<td>Sets and returns the mapping for a single port.</td>
<td>SENSE:MULTiplexer:PORT:SELect</td>
<td>ts.SelectPort</td>
</tr>
<tr>
<td>Returns the label on a given channel.</td>
<td>SENSE:MULTiplexer:LABel</td>
<td>Label</td>
</tr>
<tr>
<td>Turns ON/OFF status bar display of test set properties.</td>
<td>SENSE:MULTiplexer:DISPlay</td>
<td>ShowProperties</td>
</tr>
<tr>
<td>Sets the control lines.</td>
<td>SENSE:MULTiplexer:OUTPut</td>
<td>ControlLines</td>
</tr>
<tr>
<td>Returns the selections available for a given logical port.</td>
<td>SENSE:MULTiplexer:PORT:CATalog?</td>
<td>PortCatalog</td>
</tr>
<tr>
<td>Reads a Cal Set for the Test Set model.</td>
<td>SENSE:CORRection:CSET:TSET:ALLPorts?</td>
<td>calset.OutputPorts</td>
</tr>
<tr>
<td>Reads a Cal Set for the Port Mapping.</td>
<td>SENSE:CORRection:CSET:TSET:TYPE?</td>
<td>calset.TestSetType</td>
</tr>
<tr>
<td>All Sense Multiplexer commands</td>
<td>SENSE:MULTiplexer</td>
<td>TestsetControl Object</td>
</tr>
<tr>
<td>All Control Multiplexer commands (E5092A only)</td>
<td>CONTROL:MULTiplexer</td>
<td>None</td>
</tr>
</tbody>
</table>

**RF / IF Path Configuration (PNA-X)**
<table>
<thead>
<tr>
<th>Catalog configuration names</th>
<th>SENSE:PATH:CONFig:CATalog?</th>
<th>Configurations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load a configuration</td>
<td>SENSE:PATH:CONFig:SELect</td>
<td>LoadConfiguration</td>
</tr>
<tr>
<td>Store a configuration</td>
<td>SENSE:PATH:CONFig:STORe</td>
<td>StoreConfiguration</td>
</tr>
<tr>
<td>Delete a configuration</td>
<td>SENSE:PATH:CONFig:DELete</td>
<td>DeleteConfiguration</td>
</tr>
<tr>
<td>Read the name of a configuration</td>
<td>SENSE:PATH:CONFig:NAME?</td>
<td>Name config</td>
</tr>
<tr>
<td>Write descriptive text</td>
<td>SENSE:PATH:CONFig:DTEXT</td>
<td>DescriptiveText</td>
</tr>
<tr>
<td>Catalog all elements</td>
<td>SENSE:PATH:CONFig:ELEMent:CATalog?</td>
<td>Elements</td>
</tr>
<tr>
<td>Catalog all settings</td>
<td>SENSE:PATH:CONFig:ELEMent:VALue:CATalog?</td>
<td>Values</td>
</tr>
<tr>
<td>Set element</td>
<td>SENSE:PATH:CONFig:ELEMent</td>
<td>Element</td>
</tr>
<tr>
<td>Set value for an element</td>
<td>SENSE:PATH:CONFig:ELEMent</td>
<td>Value</td>
</tr>
<tr>
<td>Read name of current element</td>
<td>SENSE:PATH:CONFig:ELEMent</td>
<td>element.Name</td>
</tr>
</tbody>
</table>

**RF Path configuration elements / values**

**IF Path configuration elements / values**

**Power Range commands**

**All Pulse commands**

### DSP Settings (PNA-X Remote only)

<table>
<thead>
<tr>
<th>Sets ADC capture mode: auto or manual</th>
<th>SENSE:IF:FILTER:CMODE</th>
<th>ADCCaptureMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets and returns method for specifying the way the IF Frequency is determined.</td>
<td>SENSE:IF:FREQuency:AUTO</td>
<td>IFFrequencyMode</td>
</tr>
<tr>
<td>Sets and returns the IF frequency.</td>
<td>SENSE:IF:FREQuency</td>
<td>IFFrequency</td>
</tr>
<tr>
<td>Returns the Maximum allowed IFFrequency</td>
<td>SENSE:IF:FREQuency</td>
<td>MaximumIFFrequency</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Parameter</td>
</tr>
<tr>
<td>-----------------------------------------------------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>Returns the Minimum allowed IFFrequency</td>
<td>SENSE:IF:FREQuency</td>
<td>MinimumIFFrequency</td>
</tr>
<tr>
<td>Sets digital filter mode.</td>
<td>SENSE:IF:FILTer:AUTO</td>
<td>FilterMode</td>
</tr>
<tr>
<td>Sets Stage1Coefficients</td>
<td>SENSE:IF:FILTer:STAGe&lt;n&gt;:COEFicients</td>
<td>Stage1Coefficients</td>
</tr>
<tr>
<td>Sets Stage1 NCO frequency</td>
<td>SENSE:IF:FILTer:STAGe1:FREQuency</td>
<td>Stage1Frequency</td>
</tr>
<tr>
<td>Returns the maximum value of any single stage1 coefficient.</td>
<td>SENSE:IF:FILTer:STAGe&lt;n&gt;:COEFicients</td>
<td>Stage1MaximumCoefficient</td>
</tr>
<tr>
<td>Returns the maximum number of Stage1 coefficients</td>
<td>SENSE:IF:FILTer:STAGe&lt;n&gt;:COUNt?</td>
<td>Stage1MaximumCoefficientCount</td>
</tr>
<tr>
<td>Returns the minimum number of Stage1 coefficients</td>
<td>SENSE:IF:FILTer:STAGe&lt;n&gt;:COUNt?</td>
<td>Stage1MinimumCoefficientCount</td>
</tr>
<tr>
<td>Sets Stage2Coefficients</td>
<td>SENSE:IF:FILTer:STAGe&lt;n&gt;:COEFicients</td>
<td>Stage2Coefficients</td>
</tr>
<tr>
<td>Returns the maximum value of any single stage2 coefficient.</td>
<td>SENSE:IF:FILTer:STAGe&lt;n&gt;:COEFicients</td>
<td>Stage2MaximumCoefficient</td>
</tr>
<tr>
<td>Returns the maximum number of Stage2 coefficients</td>
<td>SENSE:IF:FILTer:STAGe&lt;n&gt;:COUNt?</td>
<td>Stage2MaximumCoefficientCount</td>
</tr>
<tr>
<td>Returns the minimum number of Stage2 coefficients</td>
<td>SENSE:IF:FILTer:STAGe&lt;n&gt;:COUNt?</td>
<td>Stage2MinimumCoefficientCount</td>
</tr>
<tr>
<td>Sets and returns stage3 filter type</td>
<td>SENSE:IF:FILTer:STAGe3:TYPE</td>
<td>Stage3FilterType</td>
</tr>
<tr>
<td>Returns the names of supported types of Stage3 filters.</td>
<td>SENSE:IF:FILTer:STAGe3:CATalog?</td>
<td>Stage3FilterTypes</td>
</tr>
<tr>
<td>Sets and returns the parameter value of the current filter type.</td>
<td>SENSE:IF:FILTer:STAGe3:PARameter</td>
<td>Stage3Parameter</td>
</tr>
<tr>
<td>Returns maximum parameter value for the current filter type.</td>
<td>SENSE:IF:FILTer:STAGe3:PARameter</td>
<td>Stage3ParameterMaximum</td>
</tr>
<tr>
<td>Returns minimum parameter value for the current filter type.</td>
<td>SENSE:IF:FILTer:STAGe3:PARameter</td>
<td>Stage3ParameterMinimum</td>
</tr>
<tr>
<td>Description</td>
<td>Command</td>
<td>Result</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>Returns the names of parameters for the current filter type.</td>
<td>SENSe:IF:FILTER:STAGE3:PCATalog?</td>
<td>Stage3Parameters</td>
</tr>
</tbody>
</table>
## Preset / User

<table>
<thead>
<tr>
<th>Function</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preset</td>
<td><code>SYSTem:PRESet *RST</code></td>
<td><code>app.Preset</code></td>
</tr>
<tr>
<td>Preset plus delete window</td>
<td><code>SYSTem:FPReset</code></td>
<td><code>app.Reset</code></td>
</tr>
<tr>
<td>User Preset</td>
<td><code>SYSTem:UPReset</code></td>
<td><code>app.UserPreset</code></td>
</tr>
<tr>
<td>Load User Preset file</td>
<td><code>SYSTem:UPReset:LOAD</code></td>
<td><code>app.UserPresetLoadFile</code></td>
</tr>
<tr>
<td>Save Current state as User Preset</td>
<td><code>SYSTem:UPReset:SAVE</code></td>
<td><code>app.UserPresetSaveState</code></td>
</tr>
<tr>
<td>Enable Front Panel for User Preset</td>
<td><code>SYSTem:UPReset:FPANel</code></td>
<td><code>app.UserPresetEnable</code></td>
</tr>
<tr>
<td>Quit application</td>
<td>None</td>
<td><code>app.Quit</code></td>
</tr>
</tbody>
</table>

## Security

<table>
<thead>
<tr>
<th>Function</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency Blanking</td>
<td><code>SYSTem:SECurity[:LEVel]</code></td>
<td><code>app.SecurityLevel</code></td>
</tr>
</tbody>
</table>

## Configure

<table>
<thead>
<tr>
<th>Function</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local Lockout</td>
<td><code>Local Lockout</code></td>
<td><code>app.LocalLockoutState</code></td>
</tr>
<tr>
<td>Set and return GPIB address</td>
<td>None</td>
<td><code>app.GPIBAAddress</code></td>
</tr>
<tr>
<td>Set VNA to GPIB system controller or talker/listener</td>
<td>None</td>
<td><code>app.GPIBMode</code></td>
</tr>
<tr>
<td>Set and return SICL address</td>
<td>None</td>
<td><code>app.SICLAddress</code></td>
</tr>
<tr>
<td>Control the VNA via SICL</td>
<td>None</td>
<td>app.SICL</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------</td>
<td>---------</td>
</tr>
<tr>
<td>Return Full computer name</td>
<td>None</td>
<td>app.SystemName</td>
</tr>
<tr>
<td>System Impedance</td>
<td>SENSE:CORRection:IMPedance:INPut:MAGNitude</td>
<td>SystemImpedanceZ0</td>
</tr>
<tr>
<td>Load Test Set Config file and Restart VNA.</td>
<td>SYSTEM:CONFigure</td>
<td>app.Configure</td>
</tr>
<tr>
<td>Hardware</td>
<td>Instrument Menu</td>
<td>Instrument Menu</td>
</tr>
<tr>
<td>IO Configuration</td>
<td>None</td>
<td>IOConfiguration Object</td>
</tr>
<tr>
<td>Modify the manufacturer name</td>
<td>SYSTem:PERSona:MANufacturer</td>
<td>None</td>
</tr>
<tr>
<td>Reset to original manufacturer identification</td>
<td>SYSTem:PERSona:MANufacturer:DEFAULT</td>
<td>None</td>
</tr>
<tr>
<td>Modify the product model</td>
<td>SYSTem:PERSona:MODel</td>
<td>None</td>
</tr>
<tr>
<td>Reset to original product model name</td>
<td>SYSTem:PERSona:MODel:DEFAULT</td>
<td>None</td>
</tr>
<tr>
<td>Set and return Source Port Control</td>
<td>SYSTem:ISPControl[:STATe]</td>
<td>None</td>
</tr>
</tbody>
</table>

### Hardware and Capabilities

<table>
<thead>
<tr>
<th>DSP Revision</th>
<th>SYSTem:CONFigure:REVision:DSP?</th>
<th>DspRevision</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSP FPGA</td>
<td>SYSTem:CONFigure:REVision:DSFPfpga?</td>
<td>DspFpgaRevision</td>
</tr>
<tr>
<td>CPU Speed</td>
<td>SYSTem:CONFigure:REVision:CPU?</td>
<td>CpuRevision</td>
</tr>
<tr>
<td></td>
<td>SYSTem:CONFigure:REVision:PNA:SYNthesizer:VERSion?</td>
<td>None</td>
</tr>
<tr>
<td>Hostname</td>
<td>SYSTem:COMMunicate:LAN:HOSTname?</td>
<td>GetIPConfigurationStruct, SystemName</td>
</tr>
<tr>
<td>Disk Drive Version</td>
<td>SYSTem:DISK:REVision?</td>
<td>None</td>
</tr>
<tr>
<td>Set and return the coupler state</td>
<td>SYSTem:FCORrection:CHANnel:COUpler[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>SYSTem:CONFigure:BIT?</td>
<td>None</td>
</tr>
</tbody>
</table>
Many queries regarding the capability of a specific VNA

<table>
<thead>
<tr>
<th>System:Capability</th>
<th>Capabilities Object</th>
</tr>
</thead>
</table>

### Macros

| Execute Macro | ΣΥΣΤμ:ΣΗΟΡτυ:ΕΞΕΧυ | app.ExecuteShortcut |
| Delete Macro  | ΣΥΣΤμ:ΣΗΟΡτυ:ΔΕΛετε | app.DeleteShortcut  |
| Write macro path, argument, and title | ΣΥΣΤμ:ΣΗΟΡτυ:ΠΑΤΗ | app.PutShortcut |
| Read macro path, argument, and title | ΣΥΣΤμ:ΣΗΟΡτυ:ΑΡγυμεντο | app.GetShortcut |

### Status Commands

<table>
<thead>
<tr>
<th>Status Registers</th>
<th>GP-IB/Status Registers</th>
<th>N/A</th>
</tr>
</thead>
<tbody>
<tr>
<td>*OPC;*WAI</td>
<td>GP-IB/Common_Commands</td>
<td>N/A</td>
</tr>
</tbody>
</table>

### Events

<table>
<thead>
<tr>
<th>Events</th>
<th>None</th>
<th>app.</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowAllEvents Method</td>
<td>None</td>
<td>AllEvents</td>
</tr>
<tr>
<td>AllowEventCategory Method</td>
<td>None</td>
<td>AllowEventCategory</td>
</tr>
<tr>
<td>AllowEventMessage Method</td>
<td>None</td>
<td>AllowEventMessage</td>
</tr>
<tr>
<td>AllowEventSeverity Method</td>
<td>None</td>
<td>AllowEventSeverity</td>
</tr>
<tr>
<td>DisallowAllEvents Method</td>
<td>None</td>
<td>DisallowAllEvents</td>
</tr>
<tr>
<td>MessageText Method</td>
<td>None</td>
<td>MessageText</td>
</tr>
<tr>
<td>OnCalEvent</td>
<td>None</td>
<td>OnCalEvent</td>
</tr>
<tr>
<td>OnChannelEvent</td>
<td>None</td>
<td>OnChannelEvent</td>
</tr>
<tr>
<td>OnDisplayEvent</td>
<td>None</td>
<td>OnDisplayEvent</td>
</tr>
<tr>
<td>OnHardwareEvent</td>
<td>None</td>
<td>OnHardwareEvent</td>
</tr>
<tr>
<td>Event Type</td>
<td>Description</td>
<td>Method Call</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>OnMeasurementEvent</td>
<td>None</td>
<td>app.OnMeasurementEvent</td>
</tr>
<tr>
<td>OnSCPIEvent</td>
<td>None</td>
<td>app.OnSCPIEvent</td>
</tr>
<tr>
<td>OnSystemEvent</td>
<td>None</td>
<td>app.OnSystemEvent</td>
</tr>
<tr>
<td>OnUserEvent</td>
<td>None</td>
<td>app.OnUserEvent</td>
</tr>
<tr>
<td>SetFailOnOverRange</td>
<td>None</td>
<td>app.SetFailOnOverRange</td>
</tr>
</tbody>
</table>

### Rear Panel Connector Controls

<table>
<thead>
<tr>
<th>Connector Type</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Material Handler I/O Connector</td>
<td>GPIB/Control</td>
<td>Material Handler I/O</td>
</tr>
<tr>
<td>External Test Set Connector</td>
<td>GPIB/Control</td>
<td>External Test Set</td>
</tr>
<tr>
<td>Power I/O Connector - Analog IN/Out</td>
<td>Control SCPI</td>
<td>InputVoltageEX get OutputVoltage put Output Voltage</td>
</tr>
</tbody>
</table>

### Read Temperature

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read temperature on the receiver microcircuit.</td>
<td>SENSe:TEMPerature?</td>
<td>ReceiverTemperature</td>
</tr>
</tbody>
</table>

### GPIB Pass Through

<table>
<thead>
<tr>
<th>Action Description</th>
<th>Command</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open a GPIB pass-through session</td>
<td>SYSTem:COMMunicate:GPIB:RDEvice:OPEN</td>
<td>None</td>
</tr>
<tr>
<td>Write string data to the GPIB pass-through device</td>
<td>SYSTem:COMMunicate:GPIB:RDEvice:WRITe</td>
<td>None</td>
</tr>
<tr>
<td>Write data to the GPIB pass-through device - with header</td>
<td>SYSTem:COMMunicate:GPIB:RDEvice:WBlock</td>
<td>None</td>
</tr>
<tr>
<td>Write data to the GPIB pass-through device - without header</td>
<td>SYSTem:COMMunicate:GPIB:RDEvice:WBINary</td>
<td>None</td>
</tr>
<tr>
<td>Reads string data from the GPIB pass-through device</td>
<td>SYSTem:COMMunicate:GPIB:RDEvice:READ?</td>
<td>None</td>
</tr>
<tr>
<td>Closes a GPIB pass-through session</td>
<td>SYSTem:COMMunicate:GPIB:RDEvice:CLOSe</td>
<td>None</td>
</tr>
<tr>
<td>Closes ALL GPIB pass-through sessions</td>
<td><strong>SYSTem:COMMunicate:GPIB:RDEVice:RESet</strong></td>
<td>None</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Closes ALL VISA pass-through sessions</td>
<td><strong>SYSTem:COMMunicate:VISA:RDEVice:RESet</strong></td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>VISA Pass Through</strong></th>
<th><strong>SYSTem:COMMunicate:VISA:RDEVice:OPEN</strong></th>
<th><strong>Open</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Open a VISA pass-through session</td>
<td><strong>Write string data to the VISA pass-through device.</strong></td>
<td><strong>WriteString</strong></td>
</tr>
<tr>
<td>Write string data to the VISA pass-through device.</td>
<td><strong>Write data to the VISA pass-through device - without header.</strong></td>
<td><strong>WriteBinary</strong></td>
</tr>
<tr>
<td>Write data to the VISA pass-through device - without header.</td>
<td><strong>Returns list of visa address strings or aliases.</strong></td>
<td><strong>Find</strong></td>
</tr>
<tr>
<td>Returns list of visa address strings or aliases.</td>
<td><strong>Sets timeout value for VISA pass-through commands.</strong></td>
<td><strong>GetVISATimeout</strong> <strong>SetVISATimeout</strong></td>
</tr>
<tr>
<td>Sets timeout value for VISA pass-through commands.</td>
<td><strong>Reads string data from the VISA pass-through device.</strong></td>
<td><strong>ReadString</strong></td>
</tr>
<tr>
<td>Reads string data from the VISA pass-through device.</td>
<td><strong>Reads data from the VISA pass-through device as a Safe Array of variants.</strong></td>
<td><strong>ReadBinary</strong></td>
</tr>
<tr>
<td>Reads data from the VISA pass-through device as a Safe Array of variants.</td>
<td><strong>Reads binary data in a more compact form of Safe Array.</strong></td>
<td><strong>ReadBinaryCompact</strong></td>
</tr>
<tr>
<td>Reads binary data in a more compact form of Safe Array.</td>
<td><strong>Closes a VISA pass-through session</strong></td>
<td><strong>Close</strong></td>
</tr>
<tr>
<td>Closes a VISA pass-through session</td>
<td><strong>Closes ALL VISA pass-through sessions</strong></td>
<td><strong>Reset</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Preferences</strong></th>
<th><strong>SYSTem:PREFerences:DEFine</strong></th>
<th><strong>RestoreDefaults</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset Preference default settings</td>
<td>**Touchscreen ON</td>
<td>Off**</td>
</tr>
<tr>
<td>Touchscreen ON</td>
<td><strong>Selected trace is wider</strong></td>
<td>None</td>
</tr>
<tr>
<td>Selected trace changes width briefly</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Cal: Simulated Cal Behavior</td>
<td>SENSE:CORRection:PREFerence:SIMCal</td>
<td>None</td>
</tr>
<tr>
<td>Cal: Use Primary FOM (for mmWave)</td>
<td>SENSE:CORRection:PREFerence:CALibration[:FOM]:RANGe</td>
<td>FrequencyOffsetRangeForCalComputations</td>
</tr>
<tr>
<td>Memory: Data Math 8510 Mode</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Power: RF power On during frequency sweep retrace</td>
<td>SYSTem:PREFerences:ITEM:RETRace:POWer</td>
<td>PowerOnDuringRetraceMode</td>
</tr>
<tr>
<td>Trigger: External Trigger OUT is Global</td>
<td>TRIGGER:PREferences:AIGlobal</td>
<td>AuxTriggerScopIsGlobal</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-----------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Meas: Port 1 Noise Tuner Switch state</td>
<td>SYSTem:PREferences:ITEM:SWITch:DEF</td>
<td>Port1NoiseTunerSwitchPresetsToExternal</td>
</tr>
<tr>
<td>Meas: Mathematical offset for source attenuation</td>
<td>SYSTem:PREferences:ITEM:OFFSET:SRC</td>
<td>OffsetSourceAttenuator</td>
</tr>
<tr>
<td>Marker: Programming treats Mkr 10 as Reference</td>
<td>SYSTem:PREferences:ITEM:REFMarker</td>
<td>TreatMkr10AsReference</td>
</tr>
<tr>
<td>Marker: On Preset, Coupled Markers is ON</td>
<td>SYSTem:PREferences:ITEM:MCPreset</td>
<td>MarkCoupPresetIsOn</td>
</tr>
<tr>
<td>Marker: On Preset, Coupling Method is Channel</td>
<td>SYSTem:PREferences:ITEM:MCMethod</td>
<td>MarkCoupMethPresetIsChan</td>
</tr>
<tr>
<td>Marker: Coupling controls on/off state of markers.</td>
<td>SYSTem:PREferences:ITEM:MCControl</td>
<td>MarkCoupControlsMkrState</td>
</tr>
<tr>
<td>marker for marker search</td>
<td>Marker: Sets the bandwidth search preference</td>
<td>SYSTem:PREFERENCES:ITEM:MARKer:BANDwidth:SEARCH</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>Ext Device: De-activate on PRESET and recall.</td>
<td>SYSTem:PREFERENCES:ITEM:EDEV:DPOLicy</td>
<td>ExternalDeviceDeActivatePolicy</td>
</tr>
<tr>
<td>Avg: On PRESET set two-point group delay aperture</td>
<td>SYSTem:PREFERENCES:ITEM:GDELAY:TWOPoint</td>
<td>TwoPointGroupDelayAperture</td>
</tr>
<tr>
<td>Power: On PRESET always turn power ON</td>
<td>SYSTem:PREFERENCES:ITEM:PRESet:POWER</td>
<td>PresetPowerState</td>
</tr>
<tr>
<td>Power: Report source unleveled events as errors</td>
<td>SYSTem:ERRor:REPORT:SUNLeveled</td>
<td>EnableSourceUnleveledEvents</td>
</tr>
<tr>
<td>Power: Turn source power OFF when receiver is overloaded</td>
<td>SYSTem:PREFERENCES:ITEM:RECeivers:OVERload:POWER</td>
<td>RFOffOnReceiverOverload</td>
</tr>
<tr>
<td>Power: Report when receiver is overloaded</td>
<td>SYSTem:PREFERENCES:ITEM:RECeivers:CERRor</td>
<td>ReportReceiverOverload</td>
</tr>
<tr>
<td>Setting</td>
<td>Configuration</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Preset: On PRESET show Quick Start dialog</td>
<td>SYSTem:PREFerences:ITEM:QSTart</td>
<td>ShowQuickStartOnPreset</td>
</tr>
<tr>
<td>Controls the on/off state of the preference, &quot;Use keyboard to navigate softkeys&quot;</td>
<td>SYSTem:PREFerences:ITEM:SOFTkeys:NAVigation</td>
<td>None</td>
</tr>
<tr>
<td>Sweep: Use only ramp sweeps for Auto Sweep Mode</td>
<td>SYSTem:PREFerences:ITEM:ASMRamp</td>
<td>None</td>
</tr>
<tr>
<td>System: Enable Sound</td>
<td>SYSTem:BEEPer:STATe</td>
<td>None</td>
</tr>
<tr>
<td>System: Set limit test warning sound</td>
<td>SYSTem:BEEPer:WARNing:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>System: Set sound after operation completion</td>
<td>SYSTem:BEEPer:COMPLETE:IMMediate</td>
<td>None</td>
</tr>
<tr>
<td>System: Use parallel processing</td>
<td>SYSTem:PREFerences:ITEM:CORRection:PARallel:PROCess</td>
<td>None</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>System: Set front panel remote state when a SCPI command is received</td>
<td>SYSTem:PREFerences:ITEM:REMote:AUTO[:STATe]</td>
<td>None</td>
</tr>
<tr>
<td>Ext Reference: Modify Settings on Preset and Recall</td>
<td>SYSTem:PREFerences:ITEM:ROSCillator:RECall</td>
<td>None</td>
</tr>
</tbody>
</table>

**More buttons**

<table>
<thead>
<tr>
<th>Define Data Saves</th>
<th>See File Menu</th>
<th>See File Menu</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Preset</td>
<td>See Preset</td>
<td>See Preset</td>
</tr>
<tr>
<td>Printer Page Setup</td>
<td>Hardcopy</td>
<td>None</td>
</tr>
<tr>
<td>Power Limit</td>
<td>See Power Limits</td>
<td>See Power Limits</td>
</tr>
<tr>
<td>Display and Print Colors</td>
<td>See Display</td>
<td>See Display</td>
</tr>
</tbody>
</table>

**LXI**

<table>
<thead>
<tr>
<th>Returns <strong>Structured</strong> status of the VNA networking configuration.</th>
<th>None</th>
<th>GetIPConfigurationStruct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns <strong>string</strong> status of the VNA networking configuration.</td>
<td>None</td>
<td>LANConfiguration</td>
</tr>
<tr>
<td>Resets the VNA LAN configuration.</td>
<td>None</td>
<td>LANConfigurationInitialize</td>
</tr>
<tr>
<td>Modifies settings of the VNA computer networking configuration.</td>
<td>None</td>
<td>SetIPConfiguration</td>
</tr>
</tbody>
</table>
Displays the LAN Status dialog with LAN Status Indicator showing IDENTITY.

<table>
<thead>
<tr>
<th>Error Messages</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable the display of Error Messages</td>
<td>DISPLAY:ANNotation:MESSage:STATe</td>
</tr>
<tr>
<td>Timed vs Dialog messages</td>
<td>None</td>
</tr>
</tbody>
</table>

### Millimeter-Wave Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTem:CONFigure:MWAVE:CONF:ACTive</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:CONF:ADD</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:CONF:CATalog?</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:CONF:REMove</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:FREQuency:LO:MULTiplier</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:FREQuency:LO:SOURce</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:FREQuency:LO:STARt?</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:FREQuency:LO:STOP?</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:FREQuency:RF:MULTiplier</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:FREQuency:RF:SOURce</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:FREQuency:RF:STARt?</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:FREQuency:RF:STOP?</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:FREQuency:STARt</td>
<td>No</td>
</tr>
<tr>
<td>SYSTem:CONFigure:MWAVE:FREQuency:STOP</td>
<td>No</td>
</tr>
</tbody>
</table>

**Millimeter Configuration**

- **Select Configuration**:
  - Standard PNA
  - N291A Broadband
- **Enable Modules**:
  - Port 1
  - Port 2

**Frequencies**

- **Port 1**
  - **Module**: 10.000000000 MHz
  - **Start**: 125.000000000 GHz
- **Port 2**
  - **Module**: 10.000000000 MHz
  - **Start**: 125.000000000 GHz
Your Programs on Windows

Programs that run on Workstations

If your program is running on a separate workstation and talking to the VNA using SCPI or DCOM, you may encounter problems when your program saves data to the root directory of the VNA HDD. This is because of User Account Control (UAC), a new security model with Windows 7 or Windows 10.

In the UAC default security setting, programs will not be able to save files to the root folder of the VNA. Try changing the UAC setting to the lowest security level. If it runs, but it is not acceptable to rewrite the program, then leave the UAC in this setting. The VNA software runs with UAC on either of these settings.

To access UAC settings for Windows 7:

- Minimize the VNA application
- Click the Windows Start button
- In Search Programs and Files, type UAC
To access UAC settings for Windows 10:

Minimize the VNA application
Click the Type here to search icon
In the Type here to search text field, type UAC

Default setting

![Default setting image]

Lowest security setting

![Lowest security setting image]

Programs that run on the VNA

The following issues relate to problems that may occur when you install and run your programs on the VNA. Most programs that run in Windows XP will run without changes on Windows 7 or Windows 10. However, there are differences between the two operating systems that could break more complicated programs. These incompatibilities are the same issues that software developers deal with on personal computers.

32-bit Programs

If your 32-bit program is saving files, Windows will redirect memory access to correct for different file locations. If problems occur, the path locations may need to be modified. See new path locations.

16-bit Programs

On Windows XP, there was a compatibility layer for 16-bit programs. This compatibility layer has been removed. So, your very old 16-bit programs will no longer be able to run on the VNA.

Special note about .NET applications with 64-bit Windows
In Visual Studio 2008 and earlier, the default .NET platform target was AnyCPU. On Windows 7 or Windows 10 64-bit, .NET applications that are written against the "AnyCPU" platform will compile down to a native 64-bit application. This can cause odd failures in your code as you will not be able to load any 3rd party libraries that are 32-bit only. You should always be compiling .NET applications as the "x86" platform. Starting with Visual Studio 2010, the default platform was changed to x86.
N523xRP
N523xB Rear Panel

Click Image to learn more.

See Also
PNA-X and N522xB rear-panel.
N523xB Models and Options

10 MHz Reference IN/OUT

10 MHz Reference Input When a 10 MHz external reference signal is detected at this port, it will be used as the instrument frequency reference instead of the internal frequency reference.

10 MHz Reference Output This BNC(f) connector outputs a frequency reference signal for use by other test equipment.

See SCPI command that detects an external reference signal at this connector.
See Specifications

VGA Connector Learn more
USB Hub

This USB hub contains two SuperSpeed USB ports to power your VNA peripherals. There is also one USB port below the LAN connector, on USB device port, and four USB ports on the front panel.

Limitation: The total power consumption for all eight USB ports is limited to 4.0 amps. If this limit is exceeded, all USB ports are disabled until a device is removed and power consumption falls below the limit. The total current limit for all rear panel USB ports is 2.3 amps. The total current limit for all front panel USB ports is 2 amps. When first connected, Keysight ECal modules 8509x and N4431 draw significantly more current than other modules. See Specifications.

See Important First-time USB connection note.

USB Device  Learn more

LAN Connector

This 10/100/1000 BaseT Ethernet connection has a standard 8-pin configuration and auto selects between the two data rates.

Mini DisplayPort Connector

The Mini DisplayPort is a miniature DisplayPort connector for connection to external displays.

Line Power

See Specifications

GPIB Controller and Talker/Listener Ports

The N523x can be a GPIB Controller and Talker/Listener. Learn more.

PCIe X4 Connector

The PCIe X4 connector is a 4-lane slot for future enhancements.

External and AUX Trigger I/O
MEAS TRIG IN - When enabled, VNA is triggered by signals on this connector. Learn more.

MEAS TRIG RDY When enabled, VNA outputs a 'READY' signal on this connector to other devices. Learn more.

AUX TRIG 1&2 IN When enabled, VNA accepts signals on these connectors which indicates that the external devices is ready to be triggered. Learn more.

AUX TRIG 1&2 OUT When enabled, VNA outputs signals on these connectors either before or after a measurement. Learn more.

Test Set I/O
See Details

Material Handler I/O
See details.

CPU
See CPU Speed / Performance

See Determine Your VNA's CPU Version (Internet connection required)

Solid State Drive (SSD)
See Service Guide to learn how to remove the SSD. (Internet connection required)

See Preventing VNA SSD Problems
This 15-pin female D-sub connector allows connection of external VGA monitors. With the following resolution settings, you can simultaneously view the VNA display and a remote monitor.

See Rear Panel

<table>
<thead>
<tr>
<th>External Device</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flat Panel (TFT)</td>
<td>1024x768, 800x600, 640x480</td>
</tr>
<tr>
<td>Flat Panel (DSTN)</td>
<td>800x600, 640x480</td>
</tr>
<tr>
<td>CRT Monitor</td>
<td>1280x1024, 1024x768, 800x600, 640x480</td>
</tr>
</tbody>
</table>

**How to turn ON an external monitor**

1. Minimize the VNA application.

2. Click Start > Control Panel > Hardware and Sound > Display.

3. Under Multiple displays select Duplicate desktop on 1 and 2.

4. Click OK, then OK again to confirm.

**How to view both the VNA screen and the external monitor**

The VNA must be set to the default resolution (640 x 480). While on any other resolution setting, only the external monitor can be viewed. To change resolution:

- Minimize the VNA application.
- Right-click on the VNA desktop, then click Screen resolution.
- Adjust display resolution.
Administrative Tasks Guide

Perform Administrative Tasks

- VNA Users and Passwords
- VNA Computer Properties
- Operating System Recovery
- Run Error-check and Disk Defragmenter

See Also

Windows Considerations
Quick Start Guide

The following topics can help you become familiar with your analyzer:

- Option Enable
- Quick Start Measurements
- Front Panel Tour
- Rear Panel Tour

- PNA-X and N522xB
- N523xB
- Front-panel Interface
- Powering the VNA ON and OFF
- Traces, Channels, and Windows
- Basic Measurement Sequence
- Internal Second Source
- Preferences
- Using Help
- Operating System Recovery
- Windows Considerations
- Networking and Connecting the Analyzer
- Perform Administrative Tasks

**Read:** VERY IMPORTANT NOTES:
Setup Guide
Set Up Measurements

Set up your measurement using the following information:

**Basic Settings**
- Preset the Analyzer
- Measurement Classes
- **Select a Measurement Parameter**
- Set Frequency Range
- Set Power Level
- Set Sweep Type
- Set Number of Points
- Set Triggering
- Set Data Format
- Set Scale

**Pre-configured Measurement Setups**
- Customize the Screen
- Undo/Redo Settings

**Advanced Settings**
- Source Phase Control (Opt S93088A/B)
- Phase Coherent Measurements
- Receiver Leveling
- ADC Measurements
- Copy Channels
- RF Path Configurator

**Caution:** Avoid expensive repairs to your analyzer. Read Electrostatic Discharge Protection.
Move App to Back

When running an external program on the VNA, such as a support program, the program may have a dialog that becomes blocked by the VNA application. When this happens, there are several ways to cause your program to again have focus on top of the VNA. An easy way to do this is to cause the VNA App to move to the back of the applications.

This operation must be performed each time the VNA application is selected, which again moves it to the top of another application.

<table>
<thead>
<tr>
<th>How to Move the VNA App to the Back</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Using Hardkey/SoftTab/Softkey</strong></td>
</tr>
<tr>
<td>Press <strong>System &gt; Main &gt; Move App to Back</strong>.</td>
</tr>
<tr>
<td><strong>Using a mouse</strong></td>
</tr>
<tr>
<td>Click File</td>
</tr>
<tr>
<td>Select Move App to Back</td>
</tr>
<tr>
<td>Keyboard shortcut: Alt-f-b</td>
</tr>
</tbody>
</table>

This setting is not programmable
Calibration

Cal Basics

Calibration Wizard
Select a Calibration
Using Cal Sets
Calibration Preferences

To learn about calibrating Application channels, refer to the help topic for the Application.

Cal Types

Using ECal
Perform a 4-Port Cal with a 2-Port ECal Module
TRL Cal
Power Cal
Guided Power Cal
Calibrate All Channels

Cal Concepts

Calibration Overview
About Calibration Standards
Error Correction and Interpolation
Calibration Thru Methods
Accurate Calibrations
Validity of a Calibration

Advanced Cal Topics

CalPod
CalPod as ECal
Cal Plane Manager
Dynamic Uncertainty
ECal User Characterization
Port Extensions
Fixture Simulator
Swap Adapters Method
Delta Match Calibration
Measurement Errors
Modify Cal Kits

To learn how to perform a full 4-port on-wafer calibrations manually using the 4-port 20 GHz PNA-L network analyzer, visit www.Keysight.com and search for 5989-2287EN.

Caution: Avoid expensive repairs to your analyzer. Read Electrostatic Discharge Protection.
Dynamic Uncertainty Plots
Plotting Data in Uncertainty Manager

The following selections appear in Uncertainty Manager under the Plot menu:

- Plot Data
- Show Uncertainty
- Format
- Frequency
- Scale
- Memory
- Marker
- Copy Plot to ClipBoard
- Labels and Colors

Note: Plot settings are NOT saved with a workspace (*.ml4) file.

Plot Parameters dialog box help
This choice can also be made by clicking Plot, then Plot Data, then select a parameter.

A parameter is available when the selected Uncertainty Item has data to plot. For example, Standard is available when a standard is selected on the Cal Kit Editor tab.

Click one of the selections, then click OK to plot the data.

Learn more about Standards, Port Noise, or Cable Repeatability data.

Show Uncertainty dialog box help
This dialog is available ONLY when plotting Cal Kit standard data.

**Trace Type**

- **Normal** - Display the trace without uncertainties.
- **UMax** – Display the trace as the uncertainty maximum (measured or memory data PLUS upper limit uncertainty values).
- **UMin** - Display the trace as the uncertainty minimum (measured or memory data MINUS lower limit uncertainty values).
- **UBars** – Display the uncertainties as “error bars” around the trace.
- **Shade** – Display the uncertainties as a shaded region around the trace.
- **Ellipse** – Display the uncertainties in ellipse form. Supported ONLY in Smith Chart or Polar display format.

**Coverage Factor** - Sets the level of confidence used in computing the specified measurement uncertainties. The higher the coverage factor, the higher the confidence that the computed uncertainty region includes the “true” measured data.

Allows you to plot the uncertainty associated with the selected standard data.

**Show Uncertainty dialog box help**

This dialog is available ONLY when plotting Cal Kit standard data.
This dialog is used to set the frequencies for plotting Cal Kit standards.

When a standard is selected, the response is plotted at these frequencies
When the standard is databased, the data is interpolated.
When the standard is polynomial, the model is computed at each of these frequencies.
When a standard response is exported as touchstone or DSD file, the data will contain these frequencies.

Simulation Frequencies - Shows the current list of frequencies.

**To add or remove a few frequencies:**

Insert frequencies - To add frequencies, enter a value in GHz, then click <<
Remove - To remove frequencies, select the value, then click remove.

**To quickly replace the values with a evenly-spaced step size/data points:**

Click Reset table to remove all frequencies.
Click Start/Stop to show the following dialog:

![Sweep Set Dialog](image)

Enter the Start and Stop frequencies.
Select, then enter either the (number of) Points or the Step size in GHz.
Click OK

Store - Saves the frequency list to a text file.
Retrieve - Loads the frequency list from a text file.

**Perform Math on Plotted Characterization Data**

Noise or Repeatability characterization data can be stored in Uncertainty Manager memory, then math can be performed on the Data vs. Memory.

To show this dialog, first plot data, then click Plot, then Memory.
Selection - Choose a memory location, then click Data => Memory. The current characterization data can be stored in up to 6 memory locations.

View - Select one or more of the following:

- Check a Memory location to view on the plot. Only the locations with memory saved to them are available for selection.
- Check Math to view an additional trace of the selected math operation.
- Check Data to view the original data trace

Math - Select one of the following math operations to perform on the Data and Memory location. Learn about Data Math operations.

Graphic Properties dialog box help
Scale / Format tab
To manually change scale on the X, Y, or 2nd Y axis:

Clear (uncheck) Auto.

Enter or use the +/- buttons to set the Min and Max scale of the plot. Each rectilinear plot has 10 vertical and 10 horizontal divisions.

**Draw**

Choose one or more of the following:

- **Line** - Data points are connected with a line.
- **Tag** - Each data point is highlighted.
- **Hold** - Traces remain visible after plotting new traces. The scale becomes fixed, although the plot can be rescaled manually. This is useful for comparing similar traces.

**Format**

For Noise and Repeatability, choose from Linear or Log.

For Standards, choose from all formats.

Learn about these formats.

**More tab**
Marker

- **Font** - Select the size and font for the marker readout.
- **Transparent** - Check to make the traces and grid show through the marker label.
- **Data Digits** - Select the resolution of the readout.

Trace

Select the trace for which the following settings will be made.

- **Set Lines** - Sets the trace width, color, and type.
- **Legend** - Edit the legend that appears in the upper left corner of the chart.
- **Font** - Select the size and font for the trace title.
- **2nd Axis** - Check to associate this trace with the 2nd (right) Y-axis.
- **Set as Default** - Check to make the settings apply to all traces.

Smoothing

- Drag the slider to achieve the desired amount of trace smoothing.
- **On** - Check to quickly turn ON / OFF the smoothing value.

Buttons

- **1st Axis Font** - Click, then set the size, color, and font for the primary Y-axis.
- **Grid Lines** - Click, then set the style, color, and thickness of the grid lines.
- **2nd Axis Font** - Click, then set the size, color, and font for the 2nd Y-axis.
To show this dialog, first plot data, then click **Plot**, then **Marker**.

You can add up to 4 markers for each trace.

To add a marker - Select a marker from the drop-down selector. The marker that is selected when this dialog is closed is the **ACTIVE** marker.

Move the marker using the slider. You can also move the **ACTIVE** marker by clicking on the plot with a mouse. The marker moves to the X-axis location of the mouse cursor when clicked.

Click **Delete All** or **Delete** to remove markers.

Label - Check to show a readout just above the selected marker.

Data Window - Check to show readouts for all four markers in a sizeable and movable window as shown below.

```
<table>
<thead>
<tr>
<th>Mk</th>
<th>Trk</th>
<th>GHz</th>
<th>dB</th>
<th>Deg</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>S11</td>
<td>15.00</td>
<td>-73.63 -36.89 -999.0k</td>
<td>-149.63 +/−3047.98</td>
</tr>
<tr>
<td>M2</td>
<td>S11</td>
<td>14.00</td>
<td>-63.26 -26.80 -999.0k</td>
<td>176.69 +/−1196.16</td>
</tr>
<tr>
<td>M3</td>
<td>S11</td>
<td>3.00</td>
<td>-77.29 -32.63 -999.0k</td>
<td>-77.41 +/−2295.90</td>
</tr>
<tr>
<td>M4</td>
<td>S11</td>
<td>14.00</td>
<td>-63.26 -26.80 -999.0k</td>
<td>176.69 +/−1196.16</td>
</tr>
</tbody>
</table>
```
Application Look - Select a color scheme for the entire Uncertainty Manager workspace.

Legend On - Check to show the legend on the current plot.

Titles On - Check to show Titles on the current plot.

Set Plot Titles - Starts a dialog that allows you to change all of the titles for the current plot.

Copy Plot to Clipboard - Copies an image of the current plot. You can then paste it into a word processing or image editing software.
Port Subset Correction

It is often convenient to calibrate all the ports of the instrument so that corrected data is available at every port. However, applying the resulting calset results in every port being swept. This level of correction is appropriate if every port is connected to the DUT. However, sweeping ports that are disconnected unnecessarily slows down measurement throughput. To remedy this situation, the user can tell the instrument to exclude selected ports from the correction process. This is called "port sub-setting" or "devolve calibration". This process does not modify the calset in any way. There are two settings associated with port sub-setting: an on/off state, and the list of ports that should be included in the correction for the channel.

Port sub-setting values are independent of calset selection. They are essentially a mask that is applied to the calset.

Restricting correction to Enabled Ports

For example, on a 4 port instrument, the user is measuring two DUTs. Device #1 is connected to ports 1 and 2. Device #2 is connected to ports 3 and 4. The two devices are not interconnected in any way. Channel 1 is used to measure device #1. Channel 2 is used to measure device #2. If you apply a 4 port calset to each of these channels, both channels will sweep all 4 ports. Port sub-setting can be used to reduce the level of the correction for each channel.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Port subset values</th>
<th>Measurements</th>
<th>Correction applied</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel 1</td>
<td>ON, ports 1 and 2 enabled</td>
<td>S11, S21, S12, S22</td>
<td>Full 2 Port (1,2)</td>
</tr>
<tr>
<td>Channel 2</td>
<td>ON, ports 3 and 4 enabled</td>
<td>S33, S43, S34, S44</td>
<td>Full 2 Port (3,4)</td>
</tr>
</tbody>
</table>

In this condition, when the user performs a 4 port calibration and applies the same user calset to both channels. Channel 1 sweeps ports 1 and 2. Channel 2 sweeps ports 3 and 4.

Best Effort on Disabled Ports

If measurements are added to the channel that utilize ports that are disabled in port sub-setting, those measurements will be corrected on a "best effort" basis: some correction may be applied depending on the contents of the calset. The level of correction is limited to enhanced response calibration or simple response calibration.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Port subset values</th>
<th>Measurements</th>
<th>Correction applied</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel 1</td>
<td>ON, ports 1 and 2 enabled</td>
<td>S11, S21, S12, S22</td>
<td>Full 2 Port (1,2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>S43</td>
<td>Enhanced Response(4,3)</td>
</tr>
<tr>
<td>Channel 2</td>
<td>ON, ports 3 and 4 enabled</td>
<td>S33, S43, S34, S44</td>
<td>Full 2P(3,4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>S11</td>
<td>1 Port (1)</td>
</tr>
</tbody>
</table>

This "best effort" correction cannot be turned off using the dialog. But there are SCPI commands for disabling this feature.

The correction pop up pane, accessed from the status bar, indicates port by port correction methods for VNA with 12 or less test ports. This table is updated when the port subset correction is turned on to reflect the correction methods being applied. In the image below, the pane indicates a full 4 Port calibration. On the right, the table indicates the methods after the correction was devolved to ports 1,2,3.
Power Cal for Apps

This dialog appears when the Source Cal Settings button is clicked from Swept IMD and GCA calibrations.

**Source Calibration Settings - dialog box help**

![Source Calibration Settings dialog box](image)

**Note:** Be sure that the frequency range of your power sensor covers the frequency range of your measurement. This does NOT occur automatically.

*See Important First-time USB connection note.*

**Power**

- **Port Selection**: Select the port for which the Source Cal settings will apply.
- **Power Offset**: Allows you to specify a gain or loss (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier in the path to your DUT. Following the calibration, the VNA power readouts are adjusted to this value.
- **Power Level** of the calibration is set within the Application. This can be from the Cal Wizard page, or from the power level setting of the source or port.

**Accuracy**

- **Tolerance**: Sets the maximum desired deviation from the specified Cal Power level.
- **Max Number of Readings**: Sets the maximum number of readings to take at each data point for iterating the source power.

**Use Reference Receiver for Fast Iteration**  When checked, the first reading at each data point uses a power meter to calibrate the reference receiver. Subsequent readings, if necessary to meet your accuracy requirement, are measured using the reference receiver. This technique is much faster than using the power meter with almost no degradation in accuracy.

*Note: Do NOT use the Reference Receiver for Fast Iteration feature if there is a component before the power sensor that exhibits non-linear behavior, such as a power amplifier in compression.*
Power Meter Config  Invokes the Power Meter Settings dialog box. From this dialog, you can configure two power sensors to cover the frequency range of interest.

Learn more about standard Source Power Cal.
Caution: Avoid expensive repairs to your analyzer. Read Electrostatic Discharge Protection.
Equation Editor and MatLab

MATLAB can be used with Equation Editor in two different ways:

1. When you install a full version of MATLAB on the VNA, MATLAB functions can be called directly from Equation Editor.

   Install the full 32 bit version of MATLAB.
   Press Math > Analysis > Equation Editor...
   In the Equation Editor dialog, click the Enable MATLAB button.
   You can then start calling MATLAB from within your equation editor entry field.
   Here are a few example of how you would do this:
   Matlab("S11.*S22") -> produces an array of multiplying S11*S22
   Matlab("phase(S11)") -> produces an array of unwrapped phase of the S11 trace.

2. Compile a MATLAB *.dll using the full version on your PC. Then import the *.dll into Equation Editor

   Learn how to Import Functions into Equation Editor

   The MATLAB Compile Runtime on the VNA is currently 7.14, which is shipped with R2010b (7.11).
   If you compile your *.dll using that version of MATLAB, then you don't need to change the version of MCR on the VNA.
   Determine the version of MATLAB you will use.
   You will need this version of MATLAB installed on your development machine (Step 2). You will also need a C++ compiler (Step 3).
   You will need the appropriate version of MATLAB Compiler Runtime (MCR) installed on your target machines (Step 5).

   To see installed version of MCR, check the following locations:

   On a 32-bit Windows system:
   
   C:\Program Files\Matlab\Matlab Compiler Runtime\
   C:\Program Files\Matlab\Matlab Component Runtime\

   On a 64-bit Windows system:
Use the chart below to compare MATLAB, MCR, and Compiler versions.

Install MATLAB (32-bit) on your development machine.

If your development system is 64-bit, manually navigate to \bin\win32\ on your install disk and run setup.exe to use the 32-bit installer. This requirement will be removed in future releases.

You must install with the MATLAB Compiler toolbox.

Install a compatible C++ compiler on your development machine.

For R2011b through R2013b, Windows SDK 7.1 with .NET 4.0 is sufficient.

Due to issues with the SDK installer, it is recommended to separately install .NET first, if not already installed.

The current SDK installer will also fail if Visual Studio redistributables are installed.

For a list of compatible compilers, see http://www.mathworks.com/support/compilers/

Note that LCC is only a C compiler, not C++, and is not an acceptable option.

In MATLAB, select your installed compiler by running the command: >>> mbuild -setup

Install the correct MCR on all target machines. This requires a reboot even if not prompted to do so by the installer.

Creating a MATLAB DLL

Open the MATLAB Compiler Deployment Tool, either through the user interface or with >>> deploytool, and choose to make a "C++ Shared Library" project.

Add any .m files to your project that you need in the DLL through the Deployment Tool window. This includes any helpers for functions need to execute.

Functions must take in 0 to 32 arguments and return 1 value to be used on traces.
The return value can be an array the size of a trace or a single value.
Functions not conforming aren't directly accessible from Equation Editor but must be included if other functions rely on them.

Optional: Add function descriptors. These are separate functions which provide default arguments and tooltips for the function.

Descriptors must have the same name as the function they describe with the postfix “desc”. For example, “MyFunc” could have a descriptor “MyFuncdesc”.

Descriptors take no input arguments.
Descriptors return a single string with the format “defaultArgs;tooltip”. For example, “S11,S22;Performs an operation.”

Build the project. This may take a few minutes and, if there are no errors, will generate project, src and distrib folders. The DLL will be under the distrib folder.
Some of the other generated files may be useful but are not needed for Equation Editor. Optional: Package all the distributable files.
This package can also include the appropriate MCR installer (as large as 0.5 GB).
In general this step isn't needed, only the generated DLL is required.

Notes about Writing Scripts

The MATLAB functions that will be accessed directly by the Equation Editor must follow a specific format, as noted briefly above.

Inputs: 0 to 32 vectors with dimensions [1, Sweep Size].

Constants (0, e, channel(), etc.) are expanded and passed as [1, Sweep Size] vectors.
Outputs: 1 vector with dimensions [1, Sweep Size] or [1, 1].

Outputs size [1, 1] are expanded to [1, Sweep Size] vectors automatically.
Functions with other input/output sets are not directly accessible from Equation Editor but may be included in your DLL and used by other functions, meaning helpers can be used without risk.
Existing functions that do not fit these parameters can be included along with wrapper functions which convert the parameters and outputs within the MATLAB environment.
Outputting Data Guide
Print, Save, and Recall Data

- Save and Recall a File
- Print a Displayed Measurement
- Drive Mapping

**Caution:** Avoid expensive repairs to your analyzer. Read *Electrostatic Discharge Protection.*
This topic contains the measurement uncertainty equations used to generate the uncertainty curves in Specifications documents.

It also contains general information about determining system measurement uncertainties.

Note: RSS Computations are included along with worst case computations.

Learn about the following subjects in this topic:

- Measurement Uncertainty Equations
- Forward Reflection Uncertainty
- Forward Transmission Uncertainty
- Reverse Transmission Uncertainty
- Reverse Reflection Uncertainty
- Sources of Systematic Errors
- Sources of Random Errors
- Determining Expected System Performance
- Determining Cable Stability Terms (CR1, CR2, CTM1, CTM2, CTP1, CTP2)

See Also

- Measurement Errors
- What is Measurement Calibration?
- Why is Calibration Necessary?

**Measurement Uncertainty Equations**

Any measurement result is the vector sum of the actual test device response plus all error terms. The precise effect of each error term depends on its magnitude and phase relationship to the actual test device response. When the phase of an error response is not known, phase is assumed to be worst case (-180° to +180°).

Note: The uncertainty equations are derived for two-port measurement uncertainties. However, uncertainties for a one-port device can be derived by setting S21=0 then computing the reflection uncertainties.

View the abbreviations for residual systematic errors used in the equations.

View the abbreviations for random errors used in the error models and equations.

**Forward Reflection Uncertainty**

**Equation 1: Forward Reflection Magnitude Uncertainty (Worst Case Computation)**
Equation 2: Forward Reflection Phase Uncertainty (Worst Case Computation)

\[ \Delta S_{11(\text{worst})} = \sin^{-1} \left( \frac{\sqrt{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2}}{S_{11}} \right) + 2C_{\theta 1} \]

where:

\[ \text{Systematic} = R_{EF} + B_{EF} S_{11} + B_{EF} S_{11}^2 + S_{11} S_{11} \left( B_{EF} + 2 B_{EF} B_{EF} S_{11} + B_{EF} S_{12} \right) + A_{EF} S_{11} \]

\[ \text{Stability}^2 = C^2 + R^2 \]

\[ C^2 = C_{\text{Max}}^2 \left( 1 + S_{11}^2 \right) + 4 C_{\text{Max}}^2 S_{11}^2 + C_{\text{Max}}^2 S_{12}^2 S_{12}^2 \]

\[ R^2 = \left( R_{EF} \left( 1 + S_{11}^2 \right) + 2 R_{EF} S_{11} \right)^2 + \left( R_{EF} S_{12} \right)^2 \]

\[ \text{Noise}^2 = \left( N_{F} S_{11} \right)^2 + N_{F}^2 \]

Equation 3: Forward Reflection Magnitude Uncertainty (RSS Computation)

\[ \Delta S_{11(\text{worst})} = \sqrt{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2} \]

where:

\[ \text{Systematic}^3 = E_{EF}^2 + E_{EF}^2 S_{11}^2 + E_{EF}^2 S_{11}^4 + S_{11}^2 S_{11} \left( E_{EF} + 2 E_{EF} E_{EF} S_{11} + E_{EF} S_{12} \right) + A_{EF}^2 S_{11} \]

\[ \text{Stability}^3 = C^2 + R^2 \]

\[ C^2 = C_{\text{Max}}^2 \left( 1 + S_{11}^4 \right) + 4 C_{\text{Max}}^2 S_{11}^2 + C_{\text{Max}}^2 S_{12}^2 S_{12}^2 \]

\[ R^2 = R_{EF}^2 \left( 1 + S_{11}^4 \right) + 4 R_{EF}^2 S_{11}^2 + R_{EF}^2 S_{12}^2 S_{12}^2 \]

\[ \text{Noise}^2 = \left( N_{F} S_{11} \right)^2 + N_{F}^2 \]

Equation 4: Forward Reflection Phase Uncertainty (RSS Computation)
\[ \Delta S_{\text{mag}}(\text{Worst}) = \sqrt{\sin^{-1}\left( \frac{\sqrt{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2}}{S_{11}} \right)}^2 + 4C_{22}^2 \]

\[ \text{where:} \]

- \( \text{Systematic}^2 = B_{\text{eff}}^2 + B_{\text{sp}}^2 S_{11}^2 + B_{\text{sp}}^2 S_{21}^2 + B_{\text{sp}}^2 S_{12}^2 + B_{\text{sp}}^2 S_{22}^2 + B_{\text{sp}}^2 S_{12}^2 + B_{\text{sp}}^2 S_{22}^2 \times \sin^2(A_p) \times C_{11}^2 \)
- \( \text{Stability}^2 = C^2 + R^2 \)
- \( C^2 = C_{\text{inv}}^2 + 4C_{\text{inv}}^2 S_{11}^2 + C_{\text{inv}}^2 S_{11}^2 S_{12}^2 \)
- \( R^2 = R_{11}^2 (1 + S_{11}^2) + 4R_{11}^2 S_{11}^2 + R_{12}^2 S_{21}^2 S_{11}^2 \)
- \( \text{Noise}^2 = (N_p S_{11})^2 + N_p^2 \)

**Forward Transmission Uncertainty**

Equation 5: Forward Transmission Magnitude Uncertainty (Worst Case Computation)

\[ \Delta S_{\text{mag}}(\text{Worst}) = \sqrt{\sin^{-1}\left( \frac{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2}{S_{11}} \right)}^2 + 4C_{22}^2 \]

\[ \text{where:} \]

- \( \text{Systematic} = E_{\text{eff}} + S_{21}(E_{\text{sp}} + E_{\text{sp}} S_{11} + E_{\text{sp}} S_{21} + E_{\text{sp}} E_{\text{sp}} (S_{21} S_{12} + S_{11} S_{22}) + A_{sp}) \)
- \( \text{Stability} = \sqrt{C^2 + R^2} \)
- \( C^2 = C_{\text{inv}}^2 + 4C_{\text{inv}}^2 S_{11}^2 + C_{\text{inv}}^2 S_{11}^2 S_{12}^2 \)
- \( R^2 = R_{11}^2 (1 + S_{11}^2) + 4R_{11}^2 S_{11}^2 + R_{12}^2 S_{21}^2 S_{11}^2 \)
- \( \text{Noise}^2 = (N_p S_{11})^2 + N_p^2 \)

Equation 6: Forward Transmission Phase Uncertainty (Worst Case Computation)

\[ \Delta S_{\text{phase}} = \sin^{-1}\left( \frac{\sqrt{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2}}{S_{11}} \right) + C_{22}^2 + C_{22}^2 \]

\[ \text{where:} \]

- \( \text{Systematic} = E_{\text{eff}} + S_{21}(E_{\text{sp}} + E_{\text{sp}} S_{11} + E_{\text{sp}} S_{21} + E_{\text{sp}} E_{\text{sp}} (S_{21} S_{12} + S_{11} S_{22}) + \sin(A_p)) \)
- \( \text{Stability} = \sqrt{C^2 + R^2} \)
- \( C^2 = C_{\text{inv}}^2 + 4C_{\text{inv}}^2 S_{11}^2 + C_{\text{inv}}^2 S_{11}^2 S_{12}^2 \)
- \( R^2 = R_{11}^2 (1 + S_{11}^2) + 4R_{11}^2 S_{11}^2 + R_{12}^2 S_{21}^2 S_{11}^2 \)
- \( \text{Noise}^2 = (N_p S_{11})^2 + N_p^2 \)

Equation 7: Forward Transmission Magnitude Uncertainty (RSS Computation)
\[
\Delta S_{21(\text{med})} = \sqrt{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2}
\]

where:

\[
\text{Systematic}^2 = E_{\text{ff}}^2 + S_{21}^2 \left( B_{21}^2 + E_{\text{ff}}^2 S_{11}^2 + E_{\text{ff}}^2 S_{22}^2 + E_{\text{ff}}^2 E_{\text{ff}}^2 \left( S_{11}^2 S_{22}^2 + S_{11}^2 S_{22}^2 \right) + A_{\text{ff}}^2 \right)
\]

\[
\text{Stability}^2 = C^2 + R^2
\]

\[
C^2 = S_{21}^2 \left( C_{\text{ff}}^2 + C_{\text{ff}}^2 + C_{\text{ff}}^2 S_{11}^2 + C_{\text{ff}}^2 S_{22}^2 \right)
\]

\[
R^2 = S_{21}^2 \left( R_{\text{ff}}^2 + R_{\text{ff}}^2 S_{11}^2 + R_{\text{ff}}^2 S_{22}^2 \right)
\]

\[
\text{Noise}^2 = (N_T S_{21})^2 + N_T^2
\]

Equation 8: Forward Transmission Phase Uncertainty (RSS Computation)

\[
\Delta S_{21(\text{med})} = \sqrt{\sin^2 \left( \sqrt{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2} \right) + \frac{C^2}{S_{21}} + \frac{R^2}{S_{21}}}
\]

where:

\[
\text{Systematic}^2 = E_{\text{ff}}^2 + S_{21}^2 \left( B_{21}^2 + E_{\text{ff}}^2 S_{11}^2 + E_{\text{ff}}^2 S_{22}^2 + E_{\text{ff}}^2 E_{\text{ff}}^2 \left( S_{11}^2 S_{22}^2 + S_{11}^2 S_{22}^2 \right) + \sin^2 (A_{\text{ff}}) \right)
\]

\[
\text{Stability}^2 = C^2 + R^2
\]

\[
C^2 = S_{21}^2 \left( C_{\text{ff}}^2 + C_{\text{ff}}^2 + C_{\text{ff}}^2 S_{11}^2 + C_{\text{ff}}^2 S_{22}^2 \right)
\]

\[
R^2 = S_{21}^2 \left( R_{\text{ff}}^2 + R_{\text{ff}}^2 S_{11}^2 + R_{\text{ff}}^2 S_{22}^2 \right)
\]

\[
\text{Noise}^2 = (N_T S_{21})^2 + N_T^2
\]

Reverse Transmission Uncertainty

Equation 9: Reverse Transmission Magnitude Uncertainty (Worst Case Computation)

\[
\Delta S_{21(\text{wsc})} = \sqrt{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2}
\]

where:

\[
\text{Systematic} = E_{\text{rr}}^2 + S_{11}^2 \left( B_{21}^2 + E_{\text{rr}}^2 S_{22}^2 + E_{\text{rr}}^2 S_{12}^2 + E_{\text{rr}}^2 E_{\text{rr}}^2 \left( S_{21}^2 S_{22}^2 + S_{11}^2 S_{22}^2 \right) + A_{\text{rr}}^2 \right)
\]

\[
\text{Stability} = \sqrt{C^2 + R^2}
\]

\[
C^2 = S_{11}^2 \left( C_{\text{rr}}^2 + C_{\text{rr}}^2 + C_{\text{rr}}^2 S_{22}^2 + C_{\text{rr}}^2 S_{12}^2 \right)
\]

\[
R^2 = S_{11}^2 \left( R_{\text{rr}}^2 + R_{\text{rr}}^2 S_{22}^2 + R_{\text{rr}}^2 S_{12}^2 \right)
\]

\[
\text{Noise}^2 = (N_T S_{11})^2 + N_T^2
\]

Equation 10: Reverse Transmission Phase Uncertainty (Worst Case Computation)
\[ \Delta \xi_{\text{pop}} = \sin^{-1} \left( \frac{\sqrt{\text{Systematic} + \text{Stability}^2 + \text{Noise}^2}}{S_{12}} \right) + C_{\tau_1} + C_{\tau_2} \]

where:

\[ \text{Systematic} = E_{p}S_{12} + R_{12} + E_{s}S_{22} + E_{s}E_{p}S_{11} + E_{s}S_{12} \left( S_{21}S_{12} + S_{11}S_{22} \right) + \sin \left( A_{p} \right) \]

\[ \text{Stability} = \sqrt{C^2 + R^2} \]

\[ C^2 = S_{21}^2 \left( C_{\tau_1}^2 + C_{\tau_2}^2 + C_{\eta_1}^2 + C_{\eta_2}^2 \right) \]

\[ R^2 = S_{21}^2 \left( R_{12}^2 + R_{21}^2 \right) \]

\[ \text{Noise}^2 = \left( N_r S_{12} \right)^2 + N_f^2 \]

Equation 11: Reverse Transmission Magnitude Uncertainty (RSS Computation)

\[ \Delta \xi_{\text{pop}} = \sqrt{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2} \]

where:

\[ \text{Systematic}^2 = E_{p}^2S_{12}^2 + S_{12}^2 \left( E_{s}^2 + E_{s}^2 + S_{11}^2 + E_{s}^2 \right) \left( S_{21}^2 + S_{22}^2 \right) + \sin^2 \left( A_{p} \right) \]

\[ \text{Stability}^2 = C^2 + R^2 \]

\[ C^2 = S_{21}^2 \left( C_{\tau_1}^2 + C_{\eta_1}^2 + C_{\eta_2}^2 \right) \]

\[ R^2 = S_{21}^2 \left( R_{12}^2 + R_{21}^2 \right) \]

\[ \text{Noise}^2 = \left( N_r S_{12} \right)^2 + N_f^2 \]

Equation 12: Reverse Transmission Phase Uncertainty (RSS Computation)

\[ \Delta \xi_{\text{pop}} = \sin^{-1} \left( \frac{\sqrt{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2}}{S_{12}} \right) + C_{\tau_1} + C_{\tau_2} \]

where:

\[ \text{Systematic}^2 = E_{p}^2S_{12}^2 + S_{12}^2 \left( E_{s}^2 + E_{s}^2 + S_{11}^2 + E_{s}^2 \right) \left( S_{21}^2 + S_{22}^2 \right) + \sin^2 \left( A_{p} \right) \]

\[ \text{Stability}^2 = C^2 + R^2 \]

\[ C^2 = S_{21}^2 \left( C_{\tau_1}^2 + C_{\eta_1}^2 + C_{\eta_2}^2 \right) \]

\[ R^2 = S_{21}^2 \left( R_{12}^2 + R_{21}^2 \right) \]

\[ \text{Noise}^2 = \left( N_r S_{12} \right)^2 + N_f^2 \]

Reverse Reflection Uncertainty

Equation 13: Reverse Reflection Magnitude Uncertainty (Worst Case Computation)
Equation 14: Reverse Reflection Phase Uncertainty (Worst Case Computation)

$$\Delta S_{21(\text{wac})} = \sin^{-1}\left(\frac{\sqrt{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2}}{S_{22}}\right) + 2C_{\pi/2}$$

where:

- **Systematic** = $E_{23} + E_{32}S_{22} + E_{32}^2S_{22}^2 + E_{23}^2S_{22} + R_{21}S_{22} (E_{23} + 2E_{32}E_{12}S_{22} + E_{32}^2S_{11}) + A_{34}S_{22}$
- **Stability** = $\sqrt{C^2 + R^2}$
- $C^2 = C_{\text{kkz}} (1 + S_{22}^2) + 4C_{\text{mc}}^2S_{22} + C_{\text{mc}}^2S_{21}^2S_{12}^2$
- $R^2 = (R_{23} (1 + S_{22}^2) + 2R_{22}S_{22})^2 + (R_{21}S_{21}S_{12})^2$
- **Noise** = $(N_xS_{22})^2 + N_y^2$

Equation 15: Reverse Reflection Magnitude Uncertainty (RSS Computation)

$$\Delta S_{21(\text{rss})} = \sqrt{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2}$$

where:

- **Systematic** = $E_{23} + E_{32}S_{22} + E_{32}^2S_{22}^2 + E_{23}S_{22} + R_{21}S_{22} (E_{23} + 2E_{32}E_{12}S_{22} + E_{32}^2S_{11}) + \sin(A_{3})S_{22}$
- **Stability** = $\sqrt{C^2 + R^2}$
- $C^2 = C_{\text{kkz}} (1 + S_{22}^2) + 4C_{\text{mc}}^2S_{22} + C_{\text{mc}}^2S_{21}^2S_{12}^2$
- $R^2 = (R_{23} (1 + S_{22}^2) + 2R_{22}S_{22})^2 + (R_{21}S_{21}S_{12})^2$
- **Noise** = $(N_xS_{22})^2 + N_y^2$

Equation 16: Reverse Reflection Phase Uncertainty (RSS Computation)
Sources of Systematic Errors

The residual (after measurement calibration) systematic errors result from imperfections in the calibration standards. For reflection measurements, the associated residual errors are:

- residual directivity
- residual source match
- residual load match
- residual reflection tracking

For transmission measurements, the additional residual errors are:

- residual crosstalk
- residual load match
- residual transmission tracking

The listing below shows the abbreviations used for residual systematic errors that are in the uncertainty equations.

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDF</td>
<td>forward residual directivity</td>
</tr>
<tr>
<td>ESR</td>
<td>reverse residual source match</td>
</tr>
<tr>
<td>ESF</td>
<td>forward residual source match</td>
</tr>
<tr>
<td>ERR</td>
<td>reverse residual reflection tracking</td>
</tr>
<tr>
<td>ERF</td>
<td>forward residual reflection tracking</td>
</tr>
<tr>
<td>EXR</td>
<td>reverse crosstalk</td>
</tr>
<tr>
<td>EXF</td>
<td>forward crosstalk</td>
</tr>
<tr>
<td>ELM</td>
<td>reverse load match</td>
</tr>
<tr>
<td>ELF</td>
<td>forward load match</td>
</tr>
<tr>
<td>ETR</td>
<td>reverse transmission tracking</td>
</tr>
<tr>
<td>ETF</td>
<td>forward transmission tracking</td>
</tr>
<tr>
<td>AM</td>
<td>magnitude dynamic accuracy</td>
</tr>
<tr>
<td>AP</td>
<td>phase dynamic accuracy</td>
</tr>
<tr>
<td>EDR</td>
<td>reverse residual directivity</td>
</tr>
</tbody>
</table>

All measurements are affected by dynamic accuracy. Dynamic accuracy includes: errors during internal self-calibration routines, gain compression in the microwave frequency converter (sampler) at high signal levels, errors generated in the synchronous detectors, localized non-linearities in the IF filter system, and from LO leakage into the IF signal paths.

Sources of Random Errors

The random error sources are:

- noise
- connector repeatability
- interconnecting cable stability

There are two types of noise in any measurement system:

- low level noise (noise floor)
- high level noise (trace noise)

\[
\Delta S_{\text{sys}}(\text{phase}) = \sqrt{\sin^2 \left( \frac{\text{Systematic}^2 + \text{Stability}^2 + \text{Noise}^2}{S_{22}} \right)^2 + 4 \frac{C^2}{\eta_0}}
\]

where:

\[
\text{Systematic}^2 = E_{D1}^2 + E_{D2}^2 + E_{S1}^2 + E_{S2}^2 + E_{R1}^2 + E_{R2}^2 + E_{L1}^2 + E_{L2}^2 + E_{T1}^2 + E_{T2}^2 + \sin^2 \left( A_p \right) S_{22}^2
\]

\[
\text{Stability}^2 = C^2 + R^2
\]

\[
C^2 = C_{\text{erb}}^2 \left( 1 + S_{22}^4 \right) + 4 C_{\text{erb}}^2 S_{22}^2 + C_{\text{erb}}^2 S_{21}^4 S_{12}^2
\]

\[
R^2 = \left( R_{\text{sr}} \left( 1 + S_{22}^2 \right) + 2 R_{\text{pr}} S_{22} \right) + \left( R_{\text{rl}} S_{21} S_{12} \right) \cdot 2
\]

\[
\text{Noise}^2 = \left( N_{S} S_{22} \right) + N_{f}^2
\]
Low level noise is the broadband noise floor of the receiver which can be reduced through averaging or by changing the IF bandwidth.

High level noise or trace noise is due to the noise floor of the receiver, and the phase noise of the LO source inside the test set. It is worsened by reducing the IF bandwidth. Using a high stability 10 MHz time base can reduce high level noise.

Connector repeatability is the random variation encountered when connecting a pair of RF connectors. Variations in both reflection and transmission can be observed.

Cable stability is dependent on the cable used and the amount of cable movement between calibration and measurement. The listing below shows the abbreviations used for random errors in the error models and uncertainty equations.

\[
\begin{align*}
NF &= \text{noise floor} \\
NT &= \text{trace noise} \\
CR1 &= \text{port 1 cable reflection stability} \\
CTM1 &= \text{port 1 cable magnitude transmission stability} \\
CTP1 &= \text{port 1 cable phase transmission stability} \\
CR2 &= \text{port 2 cable reflection stability} \\
CTM2 &= \text{port 2 cable magnitude transmission stability} \\
CTP2 &= \text{port 2 cable phase transmission stability} \\
RR1 &= \text{port 1 connector reflection repeatability} \\
RT1 &= \text{port 1 connector transmission repeatability} \\
RR2 &= \text{port 2 connector reflection repeatability} \\
RT2 &= \text{port 2 connector transmission repeatability}
\end{align*}
\]

Determining Expected System Performance

Improper connection techniques and contact surfaces can degrade measurement accuracy.

Proper connection techniques include using a torque wrench with proper torque limits, ensuring that the connector pin depths meet specifications, ensuring that the center conductor of sliding loads is properly set, and observing proper handling procedures for beadless airlines.

Contact surface errors are caused by improper cleaning procedures, scratches, worn plating, and rough seating. View more information on connector care.

If proper connection techniques and connector care is observed, the following table provides an indication of connector repeatability.

<table>
<thead>
<tr>
<th>Connector Type</th>
<th>Frequency Range</th>
<th>Repeatability</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.4-mm</td>
<td>0 to 2 GHz</td>
<td>0.0002</td>
</tr>
<tr>
<td></td>
<td>2 to 8 GHz</td>
<td>0.0003</td>
</tr>
<tr>
<td></td>
<td>20 to 26.5 GHz</td>
<td>0.0010</td>
</tr>
<tr>
<td>Type-N</td>
<td>0 to 2 GHz</td>
<td>0.0001</td>
</tr>
<tr>
<td></td>
<td>2 to 8 GHz</td>
<td>0.0006</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Connector Type</th>
<th>Frequency Range</th>
<th>Repeatability</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.5-mm</td>
<td>0 to 2 GHz</td>
<td>0.0001</td>
</tr>
<tr>
<td></td>
<td>2 to 8 GHz</td>
<td>0.0006</td>
</tr>
</tbody>
</table>

8550
Determining Cable Stability Terms
(CR1, CR2, CTM1, CTM2, CTP1, CTP2)

Cable stability is dependent on the cable used and the amount of cable movement between calibration and measurement. Values for cable reflection stability are determined by connecting a fixed load to the free end of the cable and measuring the change in reflection coefficient after flexing the cable through the normal range of cable movement for a particular setup. Cable transmission stability is determined by connecting a short to the free end of the cable and measuring the change in reflection coefficient due to changes in cable position.

Graphics 1-3 demonstrate concepts useful in determining cable stability. In each case, a cable (part number 8120-4779) was connected to port 1, with a fixed load connected to the free end. A reference trace is obtained by measuring S11 with the free end held close to port 2 and storing the results in memory. Two additional S11 measurements are made; one with the cable flexed out to its straight position and the other with the cable positioned back to the same location as reference trace. As shown in Graphic 1, the flexed position demonstrates the effect of moving the cable after calibration. The repeatability trace demonstrates the stability of the cable when moved to its original position.

Graphic 1

Graphic 1 demonstrates the concepts useful in determining cable reflection stability. A fixed load is connected to the free end. The DATA-MEM feature provides an indication of the cable reflection stability. A 60-dB peak on the chart yields a reflection stability estimated as 10(−60/20) or 0.001.

Graphic 2
Graphic 2 and Graphic 3 demonstrate the concepts useful in determining cable transmission stability. A short is connected to the free end. The DATA/MEM feature provides an indication of the two-way cable transmission stability. The one-way transmission magnitude stability is determined by dividing the two-way magnitude measurement by two before it is converted to linear. A $0.013\text{ dB}$ peak on the chart yields transmission magnitude stability estimated as $10(0.013/40) - 1$ or $0.00075$. The one-way transmission phase stability is determined by dividing the two-way phase measurement by two.

Cable movement often has a much larger effect on phase measurements than magnitude measurements.

See Also

Measurement Errors

What is Measurement Calibration?

Why is Calibration Necessary?
Default EE Data

EE Default Adjustment

This program must be run whenever an assembly is replaced. This program sets correction DAC values for each assembly for each band so that the assembly outputs the proper power level at all frequencies.

Equipment needed

Power Meter and Sensor, GPIB cable (or LAN, or USB cable depending upon connection method to Power Meter), Adapters as needed.

Procedure

Note: You must be logged onto the VNA as an Administrator to perform an adjustment. Learn more.

Press System > Service > Adjustment Routines....

At the Adjustments selection, click EE Default Adjustment

This program is a combination of up to 9 different adjustments. The number of applicable adjustments varies with model number and options. The above image shows all available adjustments for a 50GHz PNA-X with Option 029. Assemblies with “NF” or “Noise” are only applicable to VNAs with Option 029 or H29 (Noise Figure). Only valid assemblies for each particular VNA will be available.

Select the appropriate assembly, then click Initialize. This presets all values to some known (default) condition. This only takes a few seconds. In general, Initializing is usually the only process that needs to be performed for all Synthesizers.

For other assemblies, click Adjustment / Verify. This will improve upon the default initialization and will result in better performance. Performing an adjustment or verification will require one or more power sensors to measure the actual output power. Connect the power sensor at the appropriate measurement point. Follow the prompts in the program. For assemblies where the connection point is not obvious, click Setup to see a picture of the actual connection point.

Selecting Verify will plot the difference between its current output power level and the specified level.
Note: The maximum measured frequency for these adjustments is only 26.5GHz, even for VNA models with a maximum frequency of 50GHz and higher.

Data Storage

The correction data for the Synth LO, Synth Src1, and Synth Src2 adjustments are stored in the EEPROM on the selected assembly.

The correction data for the remaining adjustments are stored in flash memory on the Test Set Motherboard.
System Default Calibration

The Default System Calibration creates a system cal for TDR, GDM, and other uses. This adjustment is for service only; not for measurement calibration.

Required Equipment

DC ECal or Mechanical Calibration Kit
See VNA Accessories

Notes

You must be logged onto the VNA as an Administrator to perform an adjustment. Learn more.

Procedure

Click Utility, then System, then Service, then Adjustment Routines...

At the Adjustments selection, click System Default Calibration. The System Default Calibration dialog is displayed as shown below.

Select the calibration method (ECal or Mechanical Standards).

Click on the Begin Cal button to begin the calibration.

Follow the instructions displayed in the program.

Data Storage

The correction data is stored in the hard drive.
IF Gain Adj for mm Mode

This program adjusts the gain of the IF mux in the N5290A/91A system.

This adjustment is for service only; not for measurement calibration.

**Required Equipment**

See list of supported power meters and sensors.

See VNA Accessories

**Notes**

You must be logged onto the VNA as an Administrator to perform an adjustment. [Learn more.](#)

**Procedure**

Click **Utility**, then **System**, then **Service**, then **Adjustment Routines...**

At the Adjustments selection, click **IF Gain Adj for mm Mode**. The PNA IF Gain Adjustment for Wideband dialog is displayed as shown below.

Select the first port to adjust. Each port with a frequency extender connected must be adjusted.

Click on the **Begin** button and follow the instructions displayed in the program.

Set up the power sensors then click **OK**. Up to three different power sensor configurations can be saved and recalled.
Follow the instructions displayed in the program.

**Data Storage**

The correction data is stored in the flash memory.
IF Response Adj for mm Mode

This program adjusts IF correction for the frequency extenders from the N5292A, N5261A, or N5262A test set. This adjustment is for service only; not for measurement calibration.

Required Equipment

See VNA Accessories

Notes

You must be logged onto the VNA as an Administrator to perform an adjustment. Learn more.

Procedure

Click Utility, then System, then Service, then Adjustment Routines...

At the Adjustments selection, click IF Response Adj. for mm Mode. The PNA IF Response for Millimeter Systems dialog is displayed as shown below.

Select the ports to adjust that have frequency extenders (mm heads) attached.

Connect either a short or open to the ports.

Click on the Begin button.

Follow the instructions displayed in the program.

Data Storage

8559
The correction data is stored in the flash memory.
IF Response Adjustment

This adjustment is required when upgrading. Adjustments are made to the IF amplitude, phase, and offsets.

Note: For phase adjustment, the U9391C 10 MHz to 26.5 GHz comb generator is recommended for best results instead of the higher frequency comb generator models.

When to perform

IF response adjustment is required when upgrading to Option S93090x A/B, S93093/A/B, or S93094/A/B. In addition, this adjustment must be performed each time a new configuration is set up or if cables are changed.

Requirements

Latest firmware
For Broadband systems (Option S93093/A/B):

- Leave ports open
- Connect a thru

For Banded systems (Option S93094/A/B):

- Apply a short to the waveguide ports
- U9391x comb generator (phase adjustment only)
- U9391-60009 sine to square wave converter (phase adjustment only)
- +15 VDC power supply (phase adjustment only)
- USB cables (2) (phase adjustment only)
- RF cable
- Adapters as needed

Procedure

Click Utility, then System, then Service, then Adjustment Routines....

At the Adjustments selection, click IF Response Adjustment.
Select the desired adjustments then click Begin.
Follow the instructions displayed in the program.

Data Storage

The correction data is stored in the flash memory on the Test Set Mother Board.
IF Response Adjustment_6021 and above

Note: This procedure applies ONLY to instruments with serial prefix 6021 and above with A.14.00.xx firmware and above (new DDS source).

This adjustment is required when upgrading. Adjustments are made to the IF amplitude, phase, and offsets.

When to perform

IF response adjustment is required when upgrading to Option S93090xA/B, S93093A/B, or S93094A/B. In addition, this adjustment must be performed each time a new configuration is set up or if cables are changed.

Requirements

Latest firmware
For Broadband systems (Option S93093A/B):

- Leave ports open
- Connect a thru

For Banded systems (Option S93094A/B):

- Apply a short to the waveguide ports
- RF cable
- Adapters as needed

Procedure

Click Utility, then System, then Service, then Adjustment Routines....

At the Adjustments selection, click IF Response Adjustment.
Select the desired adjustments then click Begin.
Follow the instructions displayed in the program.

Data Storage

The correction data is stored in the flash memory on the Test Set Mother Board.
LFE Receiver Adjustment mm

This program adjusts the Option 205 (2-port) or Option 425 (4-port) Low Frequency Extension (LFE) receivers in the N5293AX03/N5295AX03 Frequency Extenders for a flat response across the full frequency range. The range of this adjustment is 500 Hz to 210 MHz. Learn more about LFE.

This adjustment is for service only; not for measurement calibration.

Required Equipment

See list of supported power meters and sensors.
See VNA Accessories

Notes

- The recommended power sensor is the U8485A with Option 200 (DC - 33 GHz).
- The ECal module must function down to DC.
- You must be logged onto the VNA as an Administrator to perform an adjustment. Learn more.

Procedure

Click Utility, then System, then Service, then Adjustment Routines...

At the Adjustments selection, click LFE Receiver Adj. The LFE Head Mixer Cal Adjustment dialog is displayed as shown below.

Select the first port to adjust. Each port with a frequency extender connected must be adjusted.

Select the calibration method (ECal or Mechanical Standards).
Click on the **Begin** button and follow the instructions displayed in the program.

Follow the instructions displayed in the program.

**Data Storage**

The correction data is stored in the flash memory.
LFE Receiver Adjustment - Standard

This program adjusts the Option 205 (2-port) or Option 425 (4-port) Low Frequency Extension (LFE) receivers for a flat response across the full frequency range. The range of this adjustment is 500 Hz to 210 MHz. Learn more about LFE.

This adjustment is for service only; not for measurement calibration.

Required Equipment

See list of supported power meters and sensors.
See VNA Accessories

Notes

The recommended power sensor is the U8485A with Option 200 (DC - 33 GHz).
The ECal module must function down to DC.
You must be logged onto the VNA as an Administrator to perform an adjustment. Learn more.

Procedure

Click Utility, then System, then Service, then Adjustment Routines...

At the Adjustments selection, click LFE Receiver Adjustment. The LFE Mixer Cal Adjustment dialog is displayed as shown below.

Select the calibration method (ECal or Mechanical Standards).
Select the power sensor.
Click on the Begin button and follow the instructions displayed in the program.
Follow the instructions displayed in the program.

**Data Storage**

The correction data is stored in the flash memory.
LO Adjustment for mm Mode

This program adjusts the LO power to the frequency extenders from the N5292A, N5261A, or N5262A test set.

This adjustment is for service only; not for measurement calibration.

Required Equipment

See list of supported power meters and sensors.
See VNA Accessories

Notes

The recommended power sensor is the U8485A with Option 200 (DC - 33 GHz).

You must be logged onto the VNA as an Administrator to perform an adjustment. Learn more.

Procedure

Click Utility, then System, then Service, then Adjustment Routines...

At the Adjustments selection, click LO Adjustment for mm Mode. The PNA LO Adjustment for Wideband Systems dialog is displayed as shown below.

Select the port to adjust. Only one port needs to be adjusted since the same data is used for all ports.

Click on the Begin button.
Set up the power sensors then click **OK**. Up to three different power sensor configurations can be saved and recalled.

Follow the instructions displayed in the program.

**Data Storage**

The correction data is stored in the flash memory.
N522xB Block Diagrams

2-Port Models with Option 219 and the following:

- Opt 020 - Rear-panel IF Inputs
- Opt 021 - Pulse Modulator on Source 1
- Opt S93025A/B - Four internal pulse generators

See all options

Legend

- Bias Tee
- Source and Receiver Attenuators
- Front-panel Jumper
- Receiver

4-Port Models with Option 419 and the following:

- Opt 020 - Rear-panel IF Inputs
- Opt 021 - Pulse Modulator on Source 1
- Opt 022 - Pulse Modulator on Source 2
- Opt 025 - Four internal pulse generators
Noise Figure Adjustment

This adjustment is ONLY for the following 13.5 or 26.5 GHz models with Noise Figure:

- N5241B Opt 029
- N5242B Opt 029

See Adjustment procedure for 43.5 GHz, 50 GHz, and 67 GHz models.

This procedure does the following:

- Sets warning level for overpower conditions at the noise receiver. (Noise compression test)
- Sets IF Gain flatness for the noise receiver. (IF Gain Cal)
- Overwrites the factory noise receiver calibration. (Noise correction test).

Perform this procedure after replacing any component in the noise figure circuitry, or if the VNA fails the noise IF gain portion of the OP check.

To launch the adjustment program:

- Click Utility, then System, then Service, then Adjustment Routines....
- Click Noise Adjustment. Available only with Opt 029 or Opt H29 installed.

Equipment Required

- Compatible ECal module (N4691A recommended)
Compatible Noise source (346C recommended)
High quality RF Cable

Note: On a 50 GHz Option H29 VNA, noise figure adjustments and measurements are limited to 26.5 GHz.

Advanced  For Service personnel only.

Select Adjustments  Leave all three checked, unless otherwise instructed.

Procedure

This adjustment is made without the Noise Tuner connected. Connect the original jumper cable to the VNA front panel.

This adjustment requires a compatible ECal module. Make sure the ECal USB cable is connected to the VNA before beginning. Also required is a 26.5 GHz noise source such as the 346C.

Ignore any beeps or warnings as the adjustment program is running.

The adjustment program prompts you through the following steps:

Select the ENR file for the Noise Source you will be using. This adjustment will automatically remember, and use, the last file selected. These files should normally reside in the C:\Program Files(x86)\Keysight\Network Analyzer\Noise directory, but they can also exist elsewhere. If the proper file does not already exist, click Create New to enter data for the noise source you will be using.

Click Begin

Connect a high-quality cable between port 1 and port 2.

When prompted, connect the Noise Source to port 2, either directly or with a short, low-loss adapter.

You will be prompted to reconnect the high-quality cable between ports 1 and 2.

When prompted, disconnect the cable at port 2 and insert the ECal module between port 2 and the free end of the cable. Connect the ECal module directly to port 2 without using any adapters if possible.

The entire procedure takes about 15-30 minutes.

Noise Figure Adjustment

This adjustment is ONLY for the following 43.5, 50, and 67 GHz models with Noise Figure:

N5244A Opt 029
N5245 Opt 029
N5247A Opt 029

This procedure does the following:

Sets warning level for overpower conditions at the noise receiver. (Noise compression test)
Sets IF Gain flatness for the noise receiver. (IF Gain Cal)
Sets the proper offset to compensate for Noise Bandwidth offset variations
Overwrites the factory noise receiver calibration. (Noise correction test).
Perform this procedure after replacing any component in the noise figure circuitry, or if the VNA fails the noise IF gain portion of the OP check.

To launch the adjustment program:

Click Utility, then System, then Service, then Adjustment Routines....
Click Noise Adjustment  Available only with Opt 029 installed.

Equipment Required

Compatible ECal module (N4693A or N4694A recommended)
Power Meter with compatible Sensor (8487A, N8487A, or N8488A required)
High quality RF Cable

Note: On a 67 GHz VNA, noise figure adjustments and measurements are limited to 50 GHz.
Advanced  For Service personnel only.

Select Adjustments  Leave all four checked, unless otherwise instructed.
This adjustment requires a compatible 50 or 67 GHz ECal module. Make sure the ECal USB cable is connected to the VNA before beginning. Ignore any beeps or warnings as the adjustment program is running.

The adjustment program prompts you through the following steps:
Select the GPIB address of the power meter you will be using. If using a dual power meter, select the input channel that the sensor is connected to.

Click Begin

Follow all prompts.

When prompted for the ECal module, disconnect the cable at port 2 and insert the ECal module between port 2 and the free end of the cable. Connect the ECal module directly to port 2 without using any adapters if possible.

The entire procedure takes about 30-45 minutes.

**Noise Figure Adjustment Data Storage for all VNA models**

The correction data is stored in the flash memory on the Test Set Mother Board.
VNAX Block Diagrams

2-Port Models with Option 224 and the following:

- Opt 021 - Pulse Modulator on Source 1
- Opt 022 - Pulse Modulator on Source 2
- Opt S93025A/B - Four internal pulse generators

See all options

Block Diagrams for other configurations are available in the last pages of each specifications document.

4-Port Models with Option 423 and the following:

- Opt 021 - Pulse Modulator on Source 1
- Opt 022 - Pulse Modulator on Source 2
Opt S93025A/B - Four internal pulse generators
See all options
Preventing VNA SSD Problems

The leading cause of PNA failures is problems with the VNA Solid State Drive (SSD). These problems are usually preventable, and in many cases, recoverable (see Recovering from VNA SSD Problems). The following could save you weeks of downtime and the cost of replacing your VNA SSD.

Do NOT Modify or Reconfigure the Operating System

The Microsoft Windows operating system has been modified and optimized by Keysight to improve the performance of the VNA.

- Do NOT install a standard version of the Windows operating system on the VNA.
- Do NOT change advanced performance settings or group policies.
- Do NOT add or delete any hard disk drive partitions on the VNA.
- Do NOT modify any of the Keysight software registry entries.

Install Antivirus Protection

The VNA does NOT have antivirus protection when shipped. Use of an antivirus program is strongly recommended if you connect the VNA to the Internet.

In addition, the use of a firewall could help to protect the VNA from viruses. However, some firewalls could limit DCOM connectivity of the VNA.

Install Windows Critical Updates

The VNA is always shipped with the latest service packs and critical updates that were available at the time that the VNA is produced. We recommend that you maintain the latest available protection for your VNA by automatically accepting and installing the latest critical security patches from the Microsoft Windows Update website: http://windowsupdate.microsoft.com

Use Firmware Update

If your VNA is connected to the Internet, use the Firmware Update to obtain the latest firmware. On the VNA, click System, Service, Firmware Update to automatically update your firmware to the current revision.

If your VNA is NOT connected to the Internet, you must update firmware using a CDROM or pen drive. When updating, NEVER skip a major VNA firmware revision. For example, to update from A.02.xx to A.04.xx, first install any version of A.03.xx, then update to A.04.xx.

Remove Power from the VNA ONLY when the Power Button is Yellow

Unplugging the power cord, or otherwise removing power when the power button is green, can damage the VNA SSD. Learn more about powering the VNA ON and OFF.

Having VNA SSD Problems?

If you are experiencing problems which may be caused by a faulty VNA SSD, please see Recovering from VNA SSD Problems.
Receiver Characterization

**Equipment needed:**

The cables and adapters listed in the dialog box.

**Procedure**

*Note: You must be logged onto the VNA as an Administrator to perform an adjustment.* Learn more.

Click **Utility, then System, then Service, then Adjustment Routines....**

At the **Adjustments selection**, click **Receiver Characterization**.

Follow prompts in the program.

![Receiver Characterization](image)

Select the receivers to be characterized, or click **Select All**.

**Data Storage**

The correction data is stored in the flash memory on the Test Set Mother Board.
Receiver Characterization mm Mode

This program adjusts the high level amplitude response of the N5290A/91A system.

**Required Equipment**

See list of supported power meters and sensors.
See VNA Accessories

**Notes**

You must be logged onto the VNA as an Administrator to perform an adjustment. Learn more.

**Procedure**

Click **Utility**, then **System**, then **Service**, then **Adjustment Routines**...

At the Adjustments selection, click **Receiver Characterization**. The **Receiver Characterization** dialog is displayed as shown below.

Select the receivers to be characterized, or click **Select All**.

Follow the instructions displayed in the program.

**Data Storage**

The correction data is stored in the flash memory.
Source and Receiver Adj

This program adjusts the source and receiver of the N5293AX03/N5295AX03 Frequency Extenders.

This adjustment is for service only; not for measurement calibration.

Required Equipment

See list of supported power meters and sensors.
See VNA Accessories

Notes

You must be logged onto the VNA as an Administrator to perform an adjustment. Learn more.

Procedure

Click Utility, then System, then Service, then Adjustment Routines...

At the Adjustments selection, click Source and Receiver Adj. The PNA Src/Rcvr Adjustment dialog is displayed as shown below.

Select the first port to adjust. Each port with a frequency extender connected must be adjusted.

Click on the Begin button and follow the instructions displayed in the program.

Follow the instructions displayed in the program.
Data Storage

The correction data is stored in the flash memory.
Support Overview

Learn about your Analyzer
- Specifications
- PNA Configurations and Options
- Analyzer Accessories

Update your Analyzer
- Firmware Update (Agile Update)
- Option Enable
- Instrument Calibration

Problems with your Analyzer
- Diagnostic Tools, Utilities, and Adjustments
- Troubleshoot the Analyzer
- About Error Messages

Resources for your Analyzer
- Technical Support
- Other Resources

Tip: Use Move App to Back to cause the PNA application to move behind these support applications on the screen.

Caution: Avoid expensive repairs to your analyzer. Read Electrostatic Discharge Protection.
Synthesizer Bandwidth Adjustment

The Synth Bandwidth adjusts the loop bandwidth of the synthesizer by measuring the sideband amplitude difference of the IF signal from the main signal. The program aims for a difference of 7dB.

**Equipment needed:**

Compatible Spectrum Analyzer, GPIB cable, RF cable. An Keysight spectrum analyzer (SA) that covers 3.7 GHz is required. The SA must be controlled via GPIB.

**Procedure**

*Note: You must be logged onto the VNA as an Administrator to perform an adjustment. [Learn more.]*

Click Utility, then System, then Service, then Adjustment Routines....

At the **Adjustments selection**, click **Synthesizer Bandwidth Adjustment**.

Follow prompts in the program.

Connect the SA to the VNA.

The adjustment program will display the proper connection point. In most cases the connection point is either on the front or rear panel of the VNA, so no internal connection is needed.

For PNA-L models (N5231A-N5239A) the connection point is internal to the chassis, so the VNA must be opened.
There are either one, two, or three synthesizers depending upon model and option. Only the synthesizers valid for the particular instrument will be enabled and selectable. All should be adjusted since the actual adjustment takes less than 1 minute.

**Data Storage**

The correction data is stored in the EEPROM on the selected Synthesizer assembly.
Synthesizer DAC Adj.

The synthesizer DACs are adjusted for best phase noise. This adjustment is only required on option UNY instruments.

**Note:** Performing this adjustment on instruments that do not have option UNY can be done but will not improve phase noise. That is, it will have no effect.

**Equipment needed:**

Compatible Spectrum Analyzer, GPIB cable (or LAN cable), RF cable. An Keysight spectrum analyzer (SA) that covers 6 GHz is required. The Spectrum Analyzer must be controlled via GPIB or LAN.

**Procedure**

**Note:** You must be logged onto the VNA as an Administrator to perform an adjustment. [Learn more.]

Click **Utility**, then **System**, then **Service**, then **Adjustment Routines**...

At the **Adjustments selection**, click **Synthesizer DAC Adj**.

Follow prompts in the program.

The program will prompt for the proper connections.

Ensure that the Spectrum Analyzer is connected to the VNA via GPIB or LAN.
Click on the Begin Cal button.

**Data Storage**

The correction data is temporarily stored on the hard drive but it is then immediately uploaded to the synthesizer flash memory.
Install VDI PM5 Driver

VDI’s ‘PM5’ power meter MAY work with the VNA. However, because there are many factors out of our control, we can not guarantee the functionality or performance.

**Note:** The minimum required firmware version of the VDI PM5 power sensor is VC1501.

**Note:** VNA firmware between A.11.00.xx and A.13.20.xx do NOT include this capability. Ideally, the VNA should have the most recent VNA Firmware that is supported by your VNA.

To enable the use of the PM5 power meter with the VNA Source Power Cal and Power Meter-As-Receiver (PMAR) features, you must first install on your VNA the software that VDI supplies on a USB stick with the PM5. The procedure for installing that software on the VNA is the same as installing it on a PC, which is detailed in the manual for the PM5 which can be found here: http://www.vadiodes.com/images/Products/PowerMeter/PM5manual/VDI-724-PM5-Manual.pdf

Once that software is installed on the VNA, the PM5 is accessed via Source Power Cal and PMAR by the VISA resource string “ASRL3::INSTR”. The following dialogs illustrate how to specify that in the VNA’s Power Meter Settings dialog for Source Power Cal and the VNA’s External Devices dialog for PMAR.

![Power Meter Settings dialog](image)

For more information, refer to the Power Meter Settings dialog description.
For more information, refer to the External Device Configuration dialog description.
System Topics

System Settings and Configuration

System Settings

- Dialog Transparency
- Display Colors
- Frequency Blanking (For security purposes)
- Mechanical Devices (View channel conflicts)
- IF Path Configuration
- RF Path Configuration
- Preferences
- Power Limit and Power Offset (Prevents overpowering DUTs)
- Preset the Analyzer
- System Impedance
- Receiver Temperature
- About Error Messages

External Hardware Setup Topics

- Configure an External Device
  - Configure an External RF Source
  - Configure a Power Meter As Receiver (PMAR)
  - Configure a DC Source and DC Meter
  - Configure an External Pulse Generator
  - N5251A mmWave Test Set Configuration
  - N5290A-91A mmWave Configuration
  - External Test Set Control (Multiport)
  - E5092 Test Set Control
  - Interface Control

See Also

Administrative Tasks Guide
Connectivity Guide
Rear-Panel Tour

Caution: Avoid expensive repairs to your analyzer. Read Electrostatic Discharge Protection.
A 'Hot' S22 measurement is a normal S22 measurement of a power amplifier with drive power applied to the amplifier input. This situation more closely resembles the output match characteristics of an amplifier under normal operating conditions.

**Note:** Refer also to Active (Hot) Parameters for the latest Hot S22 measurement technique.

**Procedure**

On the PNA-X:

1. Press **Meas > S-Param > Meas Class...**, select Standard, then click OK.
2. Press **Meas > S-Param > S22**.
3. Press **Freq**, then set the Start and Stop frequencies for the measurement.
4. Press **Freq > Main > Frequency Offset...**. Enable Frequency Offset and set the Source(2) frequencies with a positive or negative offset of at least (100 * IFBW). This will be the Hot (Input) signal and must be far enough away from the S-parameter frequencies to avoid interference.
5. Press **Sweep > Sweep Timing > Sweep Mode** to STEPPED.
6. Press **Setup > Internal Hardware > RF Path Config...**.

For PNA-X models N5241B, N5242B, and N5249B: Under Configuration select Hot S-Parameters. This is necessary because these models have different path loss between J9 to Port1 and J10 to Port1 due to the bridge used as a combiner. The ‘Hot’ (input) signal coming from Source 2 is routed through the coupled arm of the combiner to port 1 and the DUT input. Configure the rear-panel cables as shown in the above Path Configuration diagram:

On the rear panel, connect 'SRC2 SW SRC OUT (J8)' to Port 1 'COMB THRU IN (J10)'
Connect Port 1 'SW SRC OUT (J11)' to Port 1 'COMB THRU IN (J9)'.
Connect DUT input port to VNA port 1
Connect DUT output port to VNA port 2.
For PNA-X models N5244B, N5245B, and N5247B: These models use a splitter as a combiner. Set both the Port 1 and Port 3 Bypass Switch to Combiner Path.

Press **Power > Main > Power and Attenuators...** See [Power and Attenuator](#) for more information.

For Port 2 (DUT output) select at least 10 dB of source attenuation on the Source 1 path.
For Port 1 (DUT input) select the S-parameter port power level.
Click Receiver Attenuator... Considering the gain of the DUT and the Source 2 input power, select the appropriate attenuation for the B receiver.

Perform a 2-port Calibration.
Press Power, then Power and Attenuator, and set Port 1 Src2 to ON.
Connect the DUT and measure S22. To ensure the DUT is being driven and the desired power level, perform a Source Power cal for Port 1 Src2.

In the Power and Attenuators dialog, adjust the Port 1 Src 2 power level and notice the change in S22.
The following image shows a Hot S22 measurement of a Hittite HMC452ST89 Power Amp tuned for 900 MHz.

The Red circled measurement is Hot S22.
The other is with the Hot signal OFF.
If the Hot signal is too close to the measurement signal, ripples appear on the out-of-band region of the trace.
Tutorials Guide

Videos
Application Notes
Connector Care
Electrostatic Discharge (ESD) Protection
Network Analyzer Basics (video)

Measurement Tutorials
Absolute Output Power
Active Probing
AM-PM Conversion
Amplifier Parameters
Antenna Measurements
Balanced Measurements
Complex Impedance
Comparing the VNA Delay Functions
Deviation from Linear Phase
Gain and Flatness
Gain Compression
Group Delay
High-Gain Amplifier Measurements
High-Power Amplifier Measurements
Hot S22 Measurement
Phase Measurements
Reverse Isolation
Reflection Measurements
SA Amplifier Harmonics Measurement
SA Converter Spurious Measurement
Synchronize an External PSG Source
Time Domain Measurements
Wide Power Sweep with Receiver Leveling

Caution: Avoid expensive repairs to your PNA. Read Electrostatic Discharge Protection.
Videos

Watch and subscribe to the latest **Keysight Network Analyzer videos** at:

https://www.youtube.com/user/KeysightNetworkAnalyzers

(Internet connection required)
Wide Power Sweep with Receiver Leveling

The following procedure illustrates how to use the Receiver Leveling feature to extend the power sweep range of the VNA, sometimes to 60 dB. Your actual sweep range may vary depending on the power offset setting and maximum available power at each frequency.

With VNA internal leveling, the power sweep range is limited to the range of the internal VNA source which is about 30 to 40 dB depending on the frequency range. See Power Sweep Range in the specs.

Note: You may not be able to achieve 60 dB of power sweep on your VNA. This is NOT specified.

With the use of source attenuation, the low power setting can be adjusted down to -95 dBm.

Without source attenuation, receiver leveling extends the low power setting to about -60 dBm. This is done by measuring and adjusting the source power at every data point. Because power levels are very low, noise must be reduced using IFBW or Averaging.

Press Sweep > Main > Sweep Setup....
In the Sweep Setup dialog select Power Sweep.
Set Start Power to -60 dBm.
Set Stop Power to 0 dBm.
Set CW Freq of the measurement. Click OK.
Press Power > Main > Power and Attenuators....
Click Leveling Mode then Receiver-R1.
Click Receiver Leveling... button.
Clear Enable source ALC hardware circuit checkbox.
When the measured power level falls below some level, the message appears: Power set to min power level.
Press Avg BW > Main > IF Bandwidth to reduce the amount of noise on the measurement.
Adjust the scale.
For troubleshooting purposes, display R1 trace to view the power level at test port 1.